

Overview of AI



Objectives

- ◆ After completing this course, you will be able to:
 - Understand the development of Artificial Intelligence.
 - Master AI technologies and related concepts.
 - Understand the justice and equity in the era of AI.
 - Understand the man-machine relationship and AI governance in the era of AI.





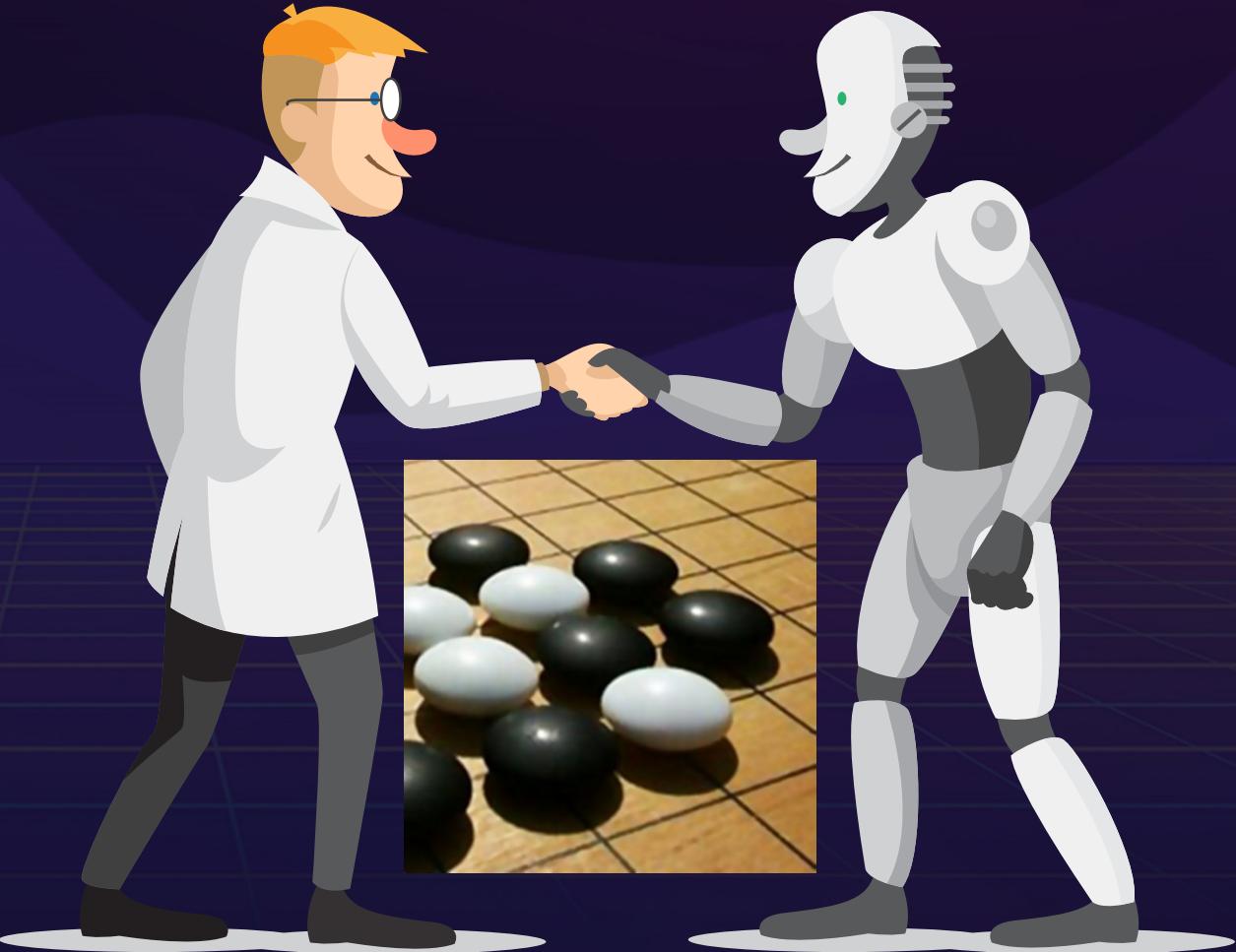
Contents

- 
- 1. The Past of AI**
 - 2. What Is AI?**
 - 3. The Present Of AI**
 - 4. Development and Strategic Planning**
 - 5. Justice and Equity**
 - 6. Man-Machine Relationship and AI Governance**
 - 7. AI Society in the Future**



The Rise of AI

- ◆ In March 2016, AlphaGo defeated Lee Sedol, a South Korean 9-dan professional Go player, by 4-1.





Dartmouth Workshop: Birth of AI



- ◆ In August 1956, some scientists and mathematicians gathered at Dartmouth College, discussing about how to make machines simulate human learning and any other feature of intelligence.
- ◆ The workshop ran for two months. No consensus was reached, but they picked the name artificial intelligence for the field they discussed about. Then, the year 1956 marked the birth of AI.



Scientists reunited



Moore

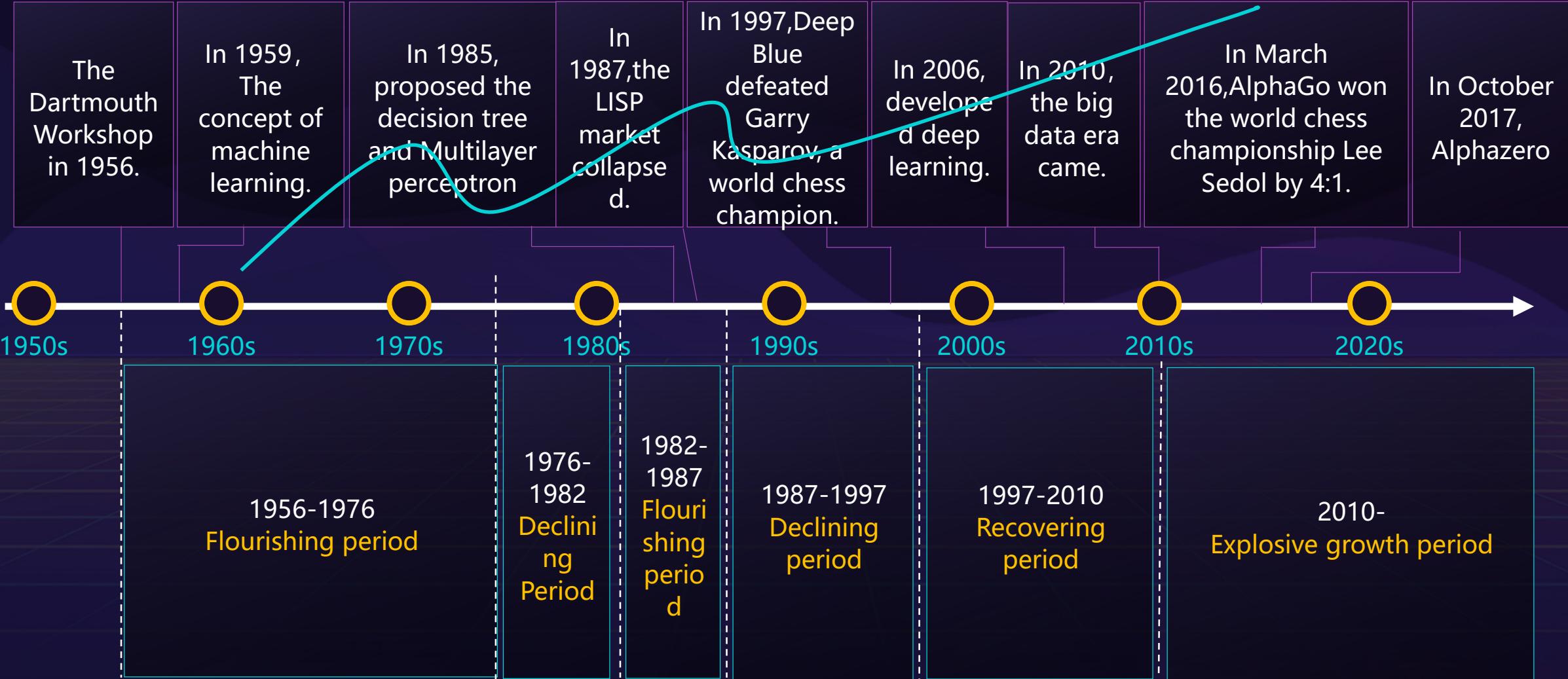
McCarthy

Minsky

Selfridge

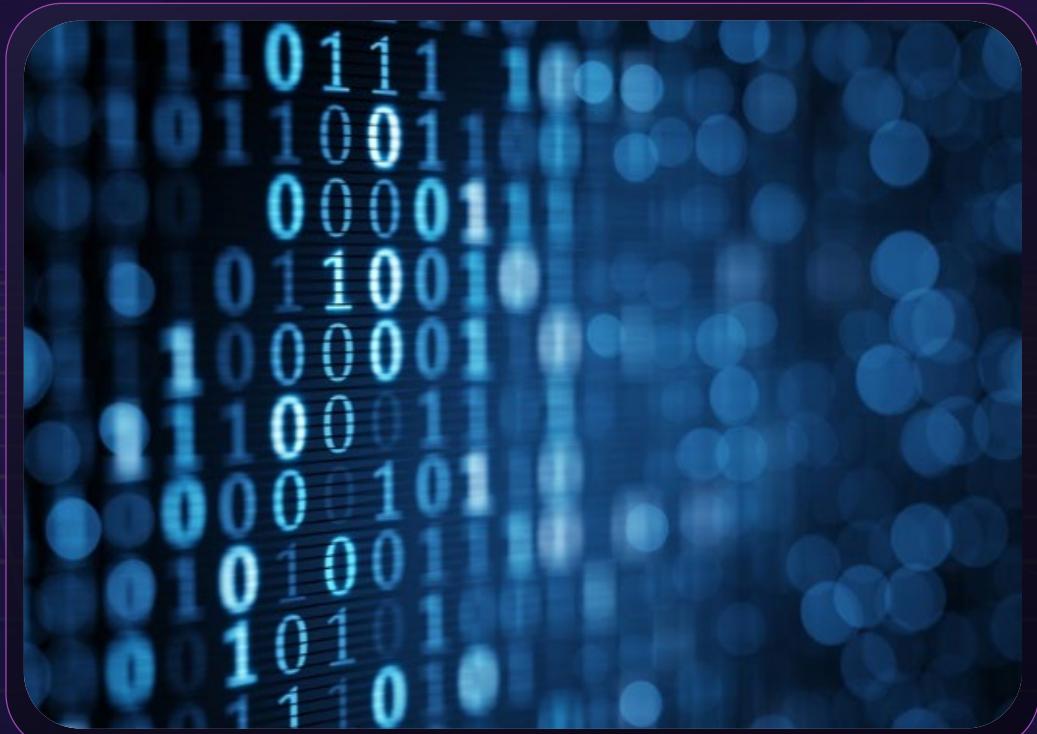
Solomonoff

AI Development History



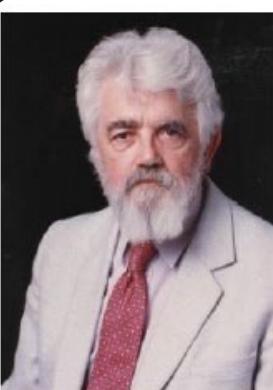
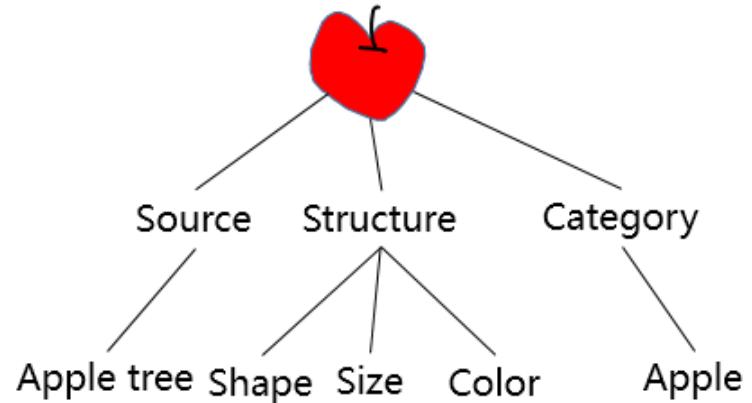
Symbolicism

- ◆ Principle: physical symbol system hypothesis and finite reasonableness principle
- ◆ Origin: mathematical logic
- ◆ Concept:
 - Symbol is the human cognition unit, and the cognition process is a symbol operation process.
 - People are regarded as a physical symbol system, so are computers. Therefore, computers can be used to simulate human behavior.
 - Knowledge is a form of information and is the basis of intelligence. The critical issues of AI are knowledge representation and knowledge inference.



Symbolism

Apple in eyes of symbolists



John McCarthy
(1927-2011)



Allen Newell
(1927-1992)



Herbert Simon
(1916-2001)



Edward Feigenbaum
(1936-)

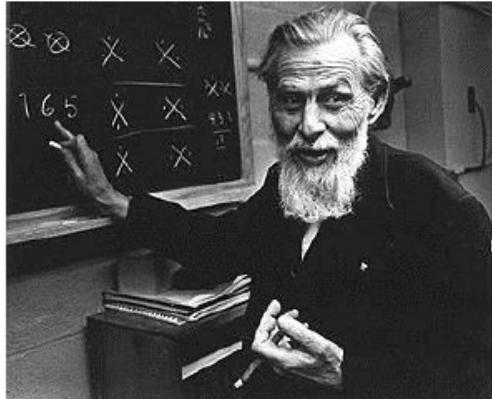
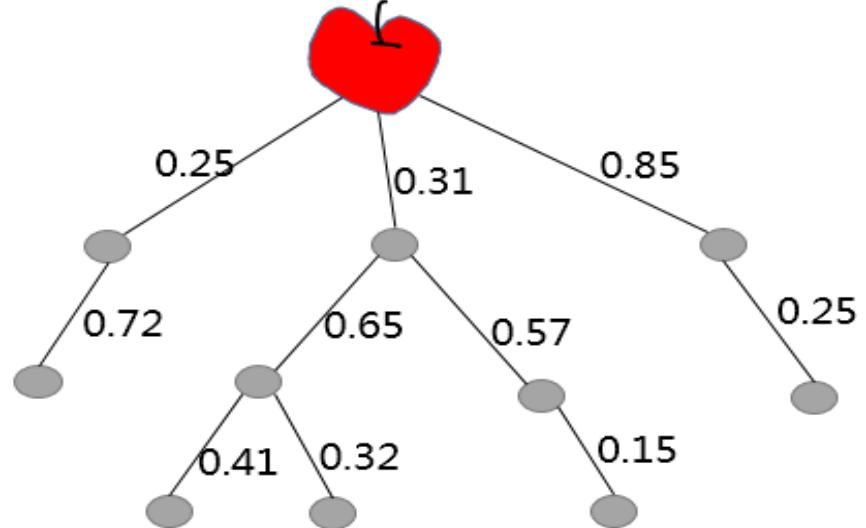
Connectionism

- ◆ **Principle:** neural network, connection mechanism and learning algorithm between neural networks
- ◆ **Origin:** bionics, especially the study of the human brain model
- ◆ **Concept:**
 - Neuron, instead of the symbol operation process, is the basic thinking unit.
 - Human brain differs from computers, and the human brain pattern can be used to replace the computer pattern.



Connectionism

Apple in eyes of connectionists



Warren S. McCulloch
(1898-1969)

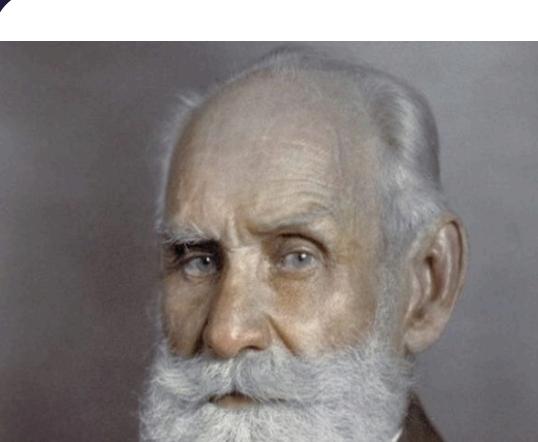


Walter H. Pitts
(1923-1969)



Marvin Minsky
(1927-2016)

Actionism



Ivan Petrovich Pavlov
(1849-1936)



John Broadus Watson
(1878-1958)



Burrhus Frederic Skinner
(1904-1990)



Actionism

- ◆ Principle: cybernetics and perception-action control system
- ◆ Origin: cybernetics
- ◆ Concept:
 - Intelligence depends on perception and actions.
 - Intelligence requires no knowledge, representation, and inference.





Mainstream AI Theories(1)

Mainstream AI Theories	Knowledge Representation	Black Box	Feature Learning	Interpretability	Requiring Large Samples
Symbolicism (logicism)	Strong	No	No	Strong	No
Connectionism (bionicsm)	Weak	Yes	Yes	Weak	Yes
Actionism (decision-making control)	Strong	No	No	Strong	No



Mainstream AI Theories(2)

Mainstream AI Theories	Computational Complexity	Combinatorial Explosion	Interaction With Environment	Overfitting
Symbolicism (logicism)	High	Many	No	No
Connectionism (bionicsm)	High	Few	No	Yes
Actionism (decision-making control)	Ordinary	Ordinary	Yes	No



History of AI Chess Games

Stochastic simulation method

Darmouth Conference
1956: the birth of AI

Mac Hack
1967: chess AI beats person in tournament

Kaissa
1974: first world computer chess champion

Backprop
1986: multi-layer neural net approach widely known

TD- Grammon
1992: RL and neural net based backgammon AI shown

NeuroGo
1996: ConvNet with RL for Go, 13kyu (amateur)

MCTS Go
2006: French researchers advance Go AI with MCTS

Crazy Stone
2008: MCTS based Go AI beats 4 dan player

Zen 19
2012: MCTS based Go AI reaches 5-dan rank



Samuel's Checkers AI
1956: IBM Checkers AI first demonstrated

Bernstein's Chess AI
1958: first fully functional chess AI developed

Checkers AI Wins
1962: Samuel's program wins game against person

Zobrist's AI
1968: First Go AI, beats human amateur

CNN
1989: convolutional nets first demonstrated

Monte Carol Go
1993: first research on Go with stochastic search

CHINOOK
1994: checkers AI draws with world champion



Deep Blue
1997: IBM chess AI beats world champion



DeepMind
2014: Google buys deep-RL AI company for 400Mil

AlphaGo
2016: Deep Learning +MCST Go AI beats top human



Quiz

1. Neural networks follow the connectionism? ()

- A. True
- B. False

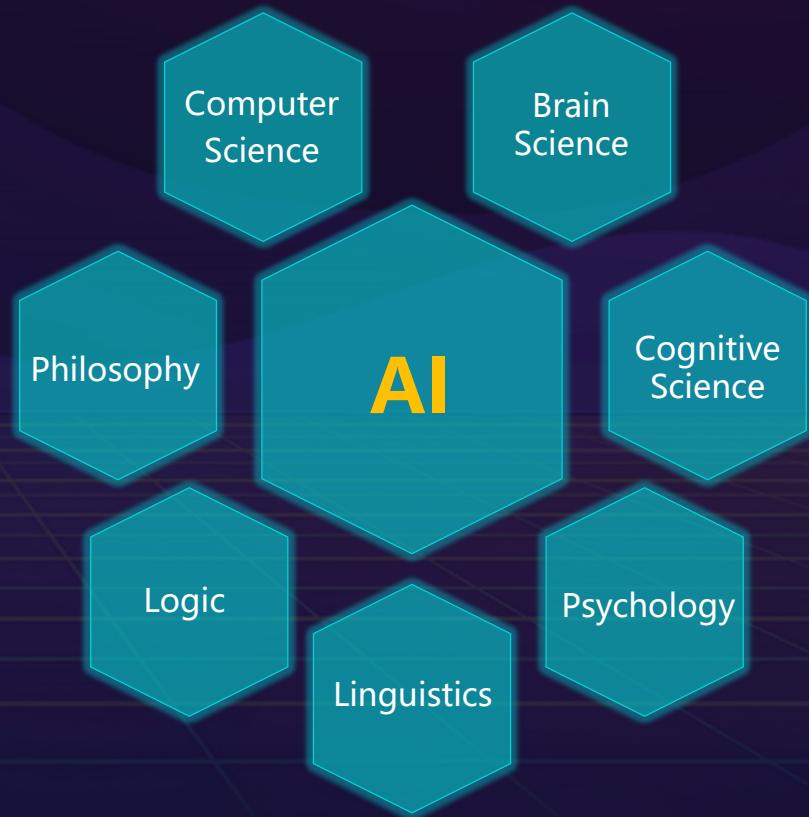


Contents

1. The Past of AI
2. What Is AI?
3. The Present Of AI
4. Development and Strategic Planning
5. Justice and Equity
6. Man-Machine Relationship and AI Governance
7. AI Society in the Future

What Is AI?

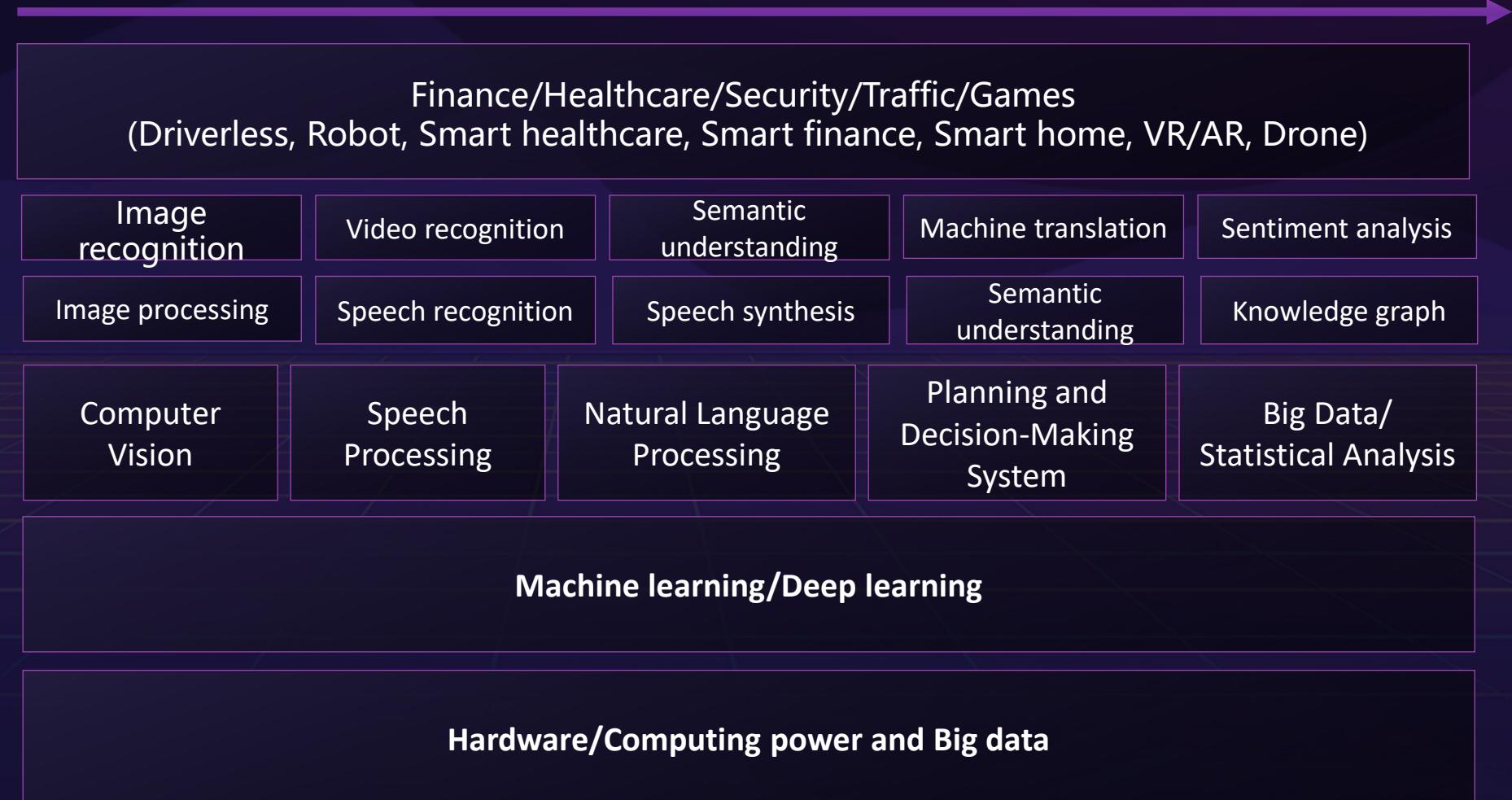
Artificial Intelligence (AI) is a technical science that studies and develops theories, methods, technologies, and applications for simulating and extending human intelligence. The purpose of AI is to enable machines to think like people and to make machines intelligent. Today, AI has become an interdisciplinary course that involves various fields.



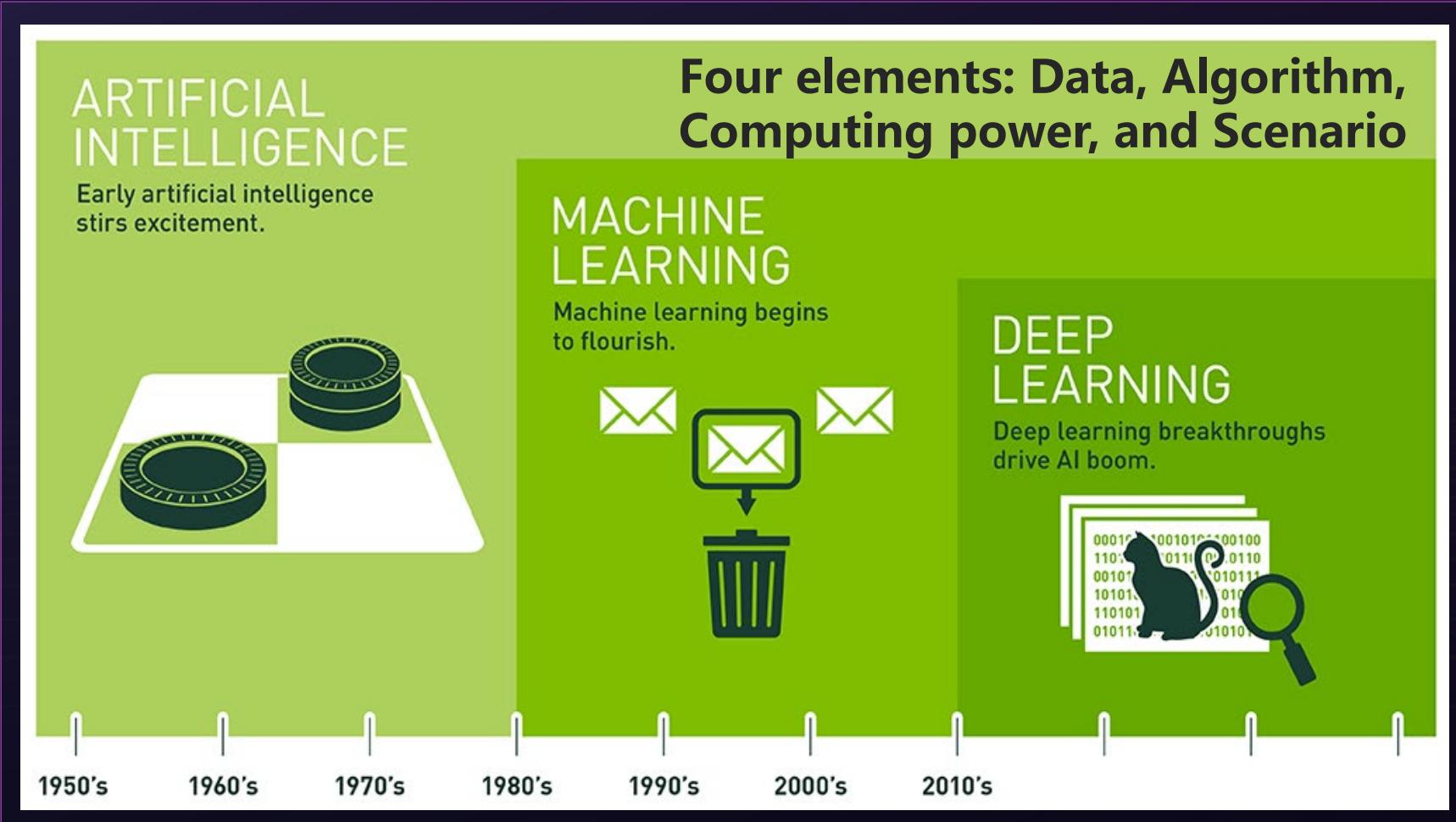


Hierarchy of AI

Awareness and analysis Understanding and Thinking Decision Making and Interaction



Relationship of AI, ML and DL





Relationship Between AI, Machine Learning, and Deep Learning

◆AI

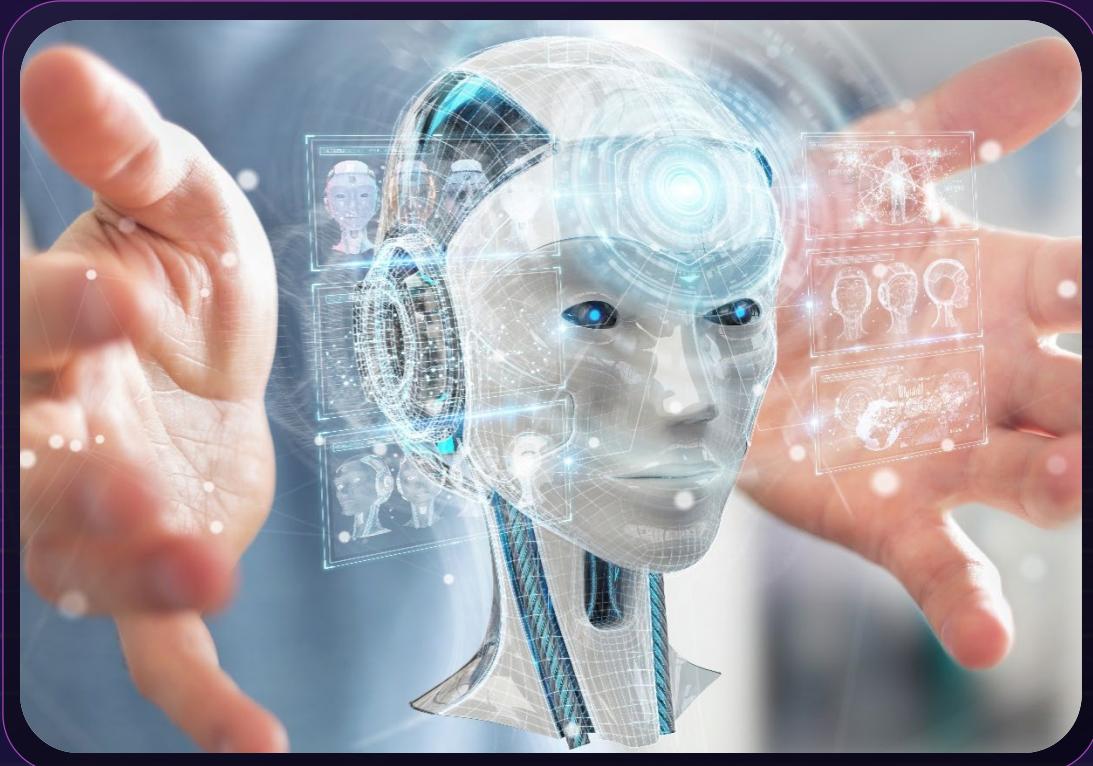
◆ AI is a technical science that studies and develops theories, methods, and applications for simulating and extending human intelligence. Besides Machine learning technology, expert system also belongs to AI.

◆Machine learning

◆ Machine learning finds patterns in dataset. Besides Neural Network, other models like decision tree, SVM are all belong to machine learning.

◆Deep learning

◆ Deep learning is the subset of machine learning, the models are Neural Networks.





Quiz

1. Deep learning is a subset of machine learning.()

- A. True
- B. False



Contents

- 
1. The Past of AI
 2. What Is AI?
 - 3. The Present Of AI**
 4. Development and Strategic Planning
 5. Justice and Equity
 6. Man-Machine Relationship and AI Governance
 7. AI Society in the Future



AI Application Scenarios



Driverless car



Smart home



Virtual reality



Intelligent robot



Smart investment adviser



Intelligent healthcare

Speech signal processing

◆ Main Technologies:

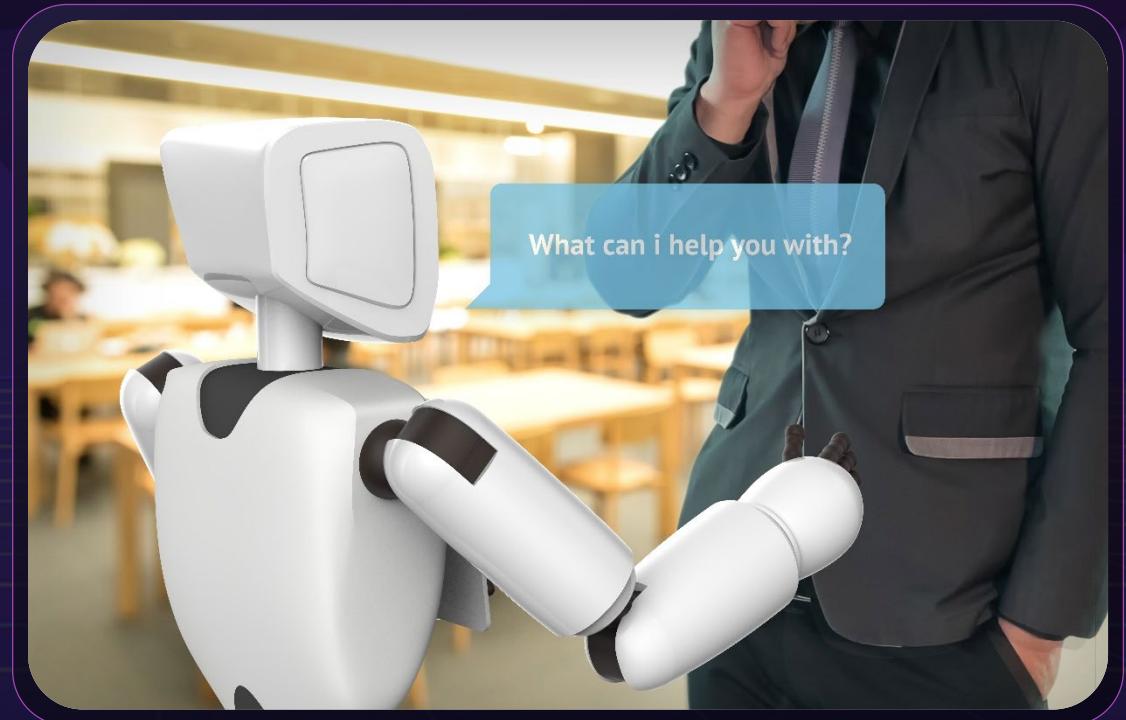
- Signal processing ➤ Semantic recognition
- Speech recognition ➤ Speech synthesis

◆ Application:

Medical dictation, speech dictation, voice operated computer system, phone customer service, etc.

◆ Future:

There is a long way to go before machines can communicate naturally with people like human beings.



Computer Vision

◆ Main technologies:

- Image processing
- Image recognition
- Image understanding

◆ Applications:

- Medical image analysis
- Shopping

◆ Future:

Computer vision is expected to enter an advanced stage of independent understanding, and analysis and decision making, truly endow machines with the ability to watch, and play a bigger role in scenarios such as driverless cars and smart home.



Natural Language Processing

◆ Translation:

- Natural language processing is one of the technical direction of artificial intelligence.

معالجة اللغة الطبيعية هي أحد التوجيهات التقنية للذكاء الاصطناعي.

◆ Content moderation:





Machine Learning

◆ **Machine learning** studies how computers simulate or implement human learning behavior to acquire new knowledge or skills, and reorganize existing knowledge structures to improve their performance continuously.

◆ **Research directions:**

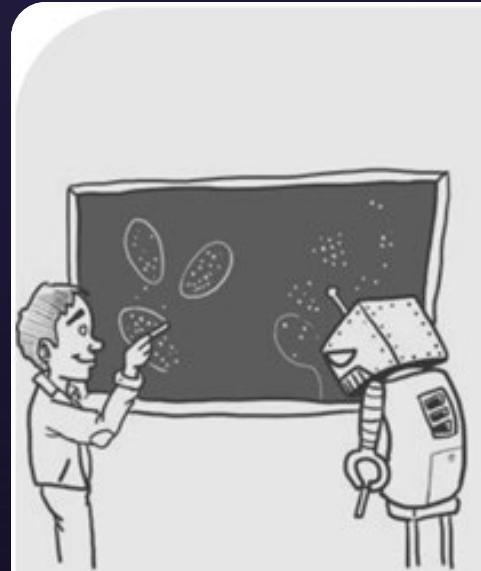
- Widely used in vertical fields, such as the finance, law, and healthcare fields
- From convex optimization to non-convex optimization
- From supervised learning to unsupervised learning and reinforcement learning

◆ **Future:** Reinforcement learning and Transfer Learning.

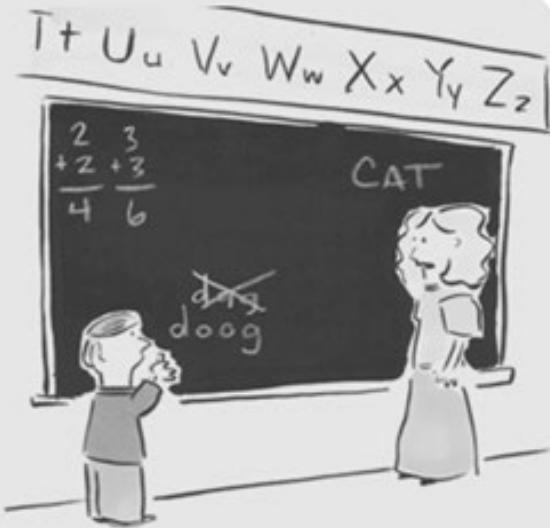




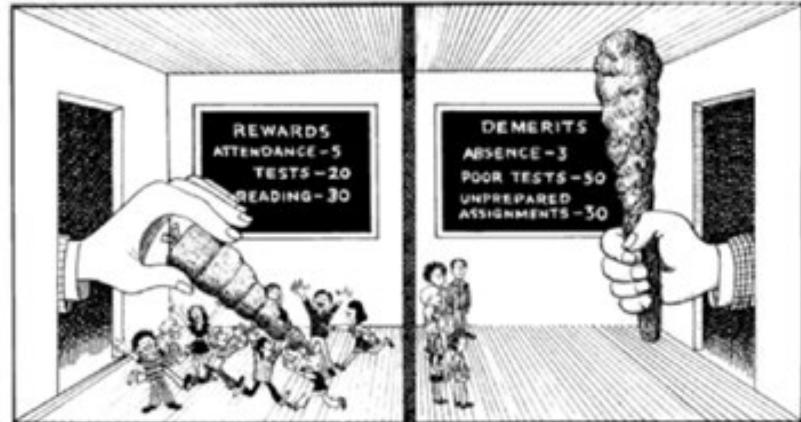
Machine Learning



Unsupervised learning:
no human interference



Supervised learning:
model answer



Reward

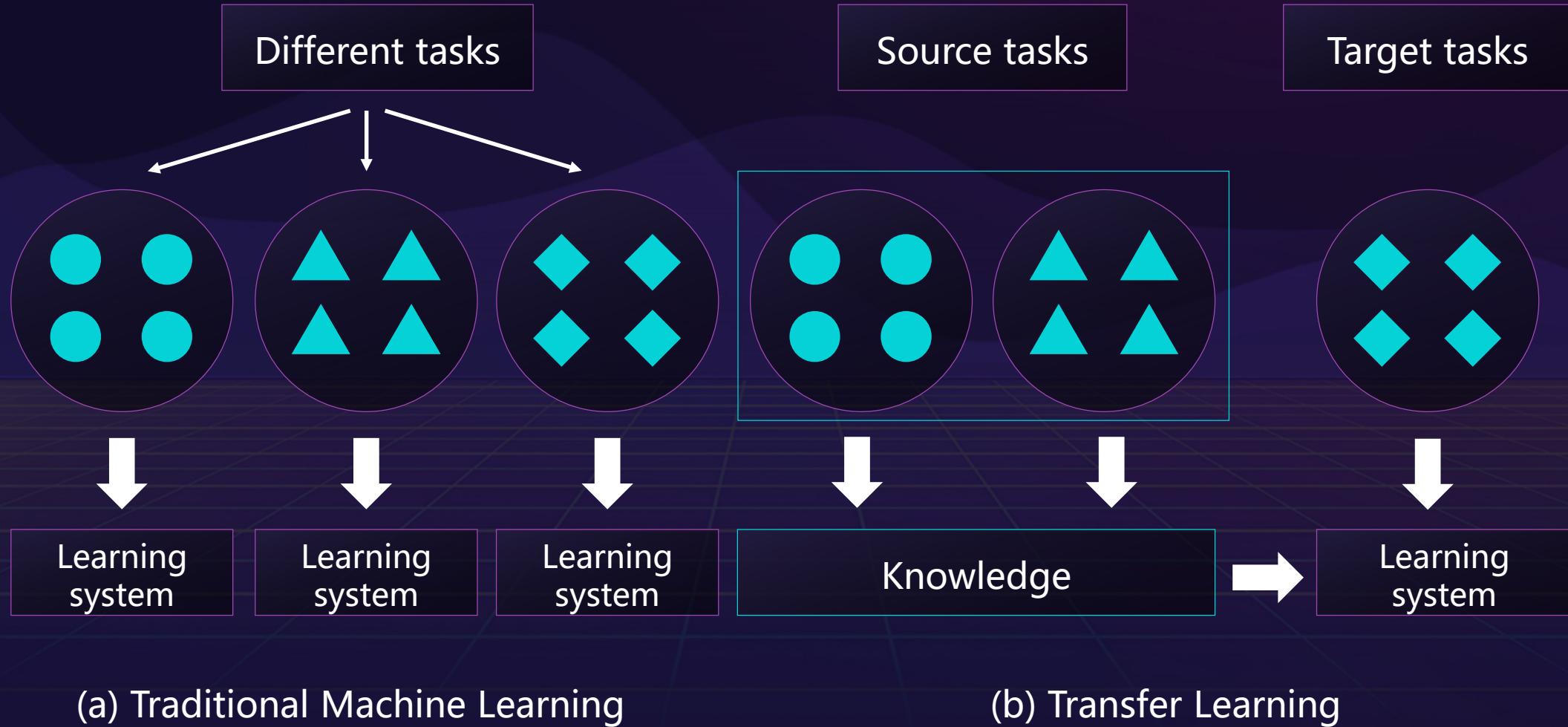
Penalty

Traditional Machine Learning

Reinforcement Learning



Machine Learning

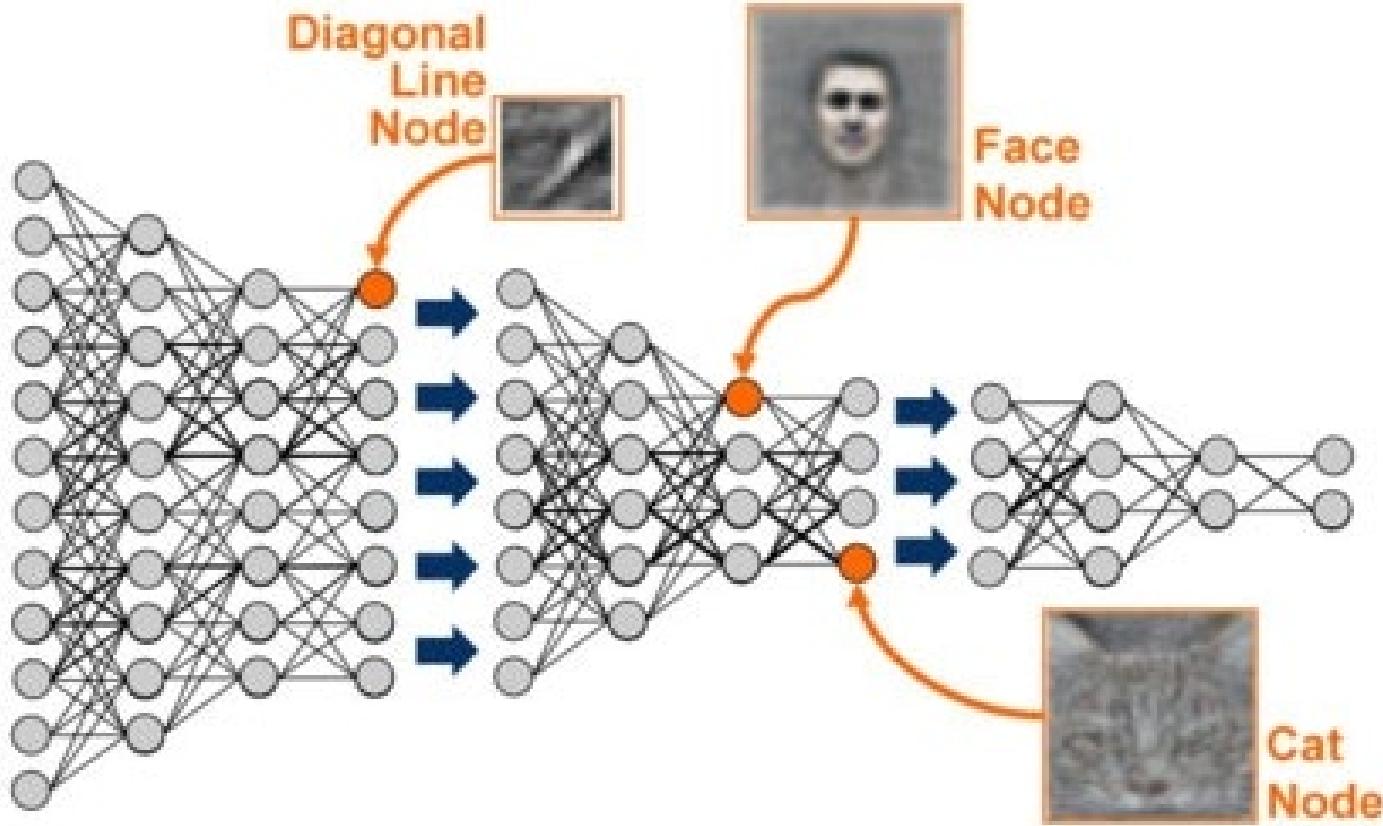


(a) Traditional Machine Learning

(b) Transfer Learning



Deep Learning





The Initial Stage

Three Phases of AI	Performance	Example	Benefits	As-is of AI: initial stage of perceptual intelligence
 Computational intelligence	Capable of storage and computation: Machines can compute and transfer information as human beings do.	Distributed computing and neural network	Help human beings store and quickly process massive data, laying a foundation for perception and cognition.	
 Perceptual intelligence	Capable of listening and seeing: Machines can listen and see, make judgments, and take simple actions.	Cameras capable of facial recognition and speakers able to understand speeches	Help human beings efficiently finish work related to listening and seeing.	
 Cognitive intelligence	Capable of understanding and thinking: Machines can understand, think, and make decisions like human beings.	Unmanned vehicles enabling autonomous driving and robots acting autonomously	Fully assist in or replace part work of human beings.	

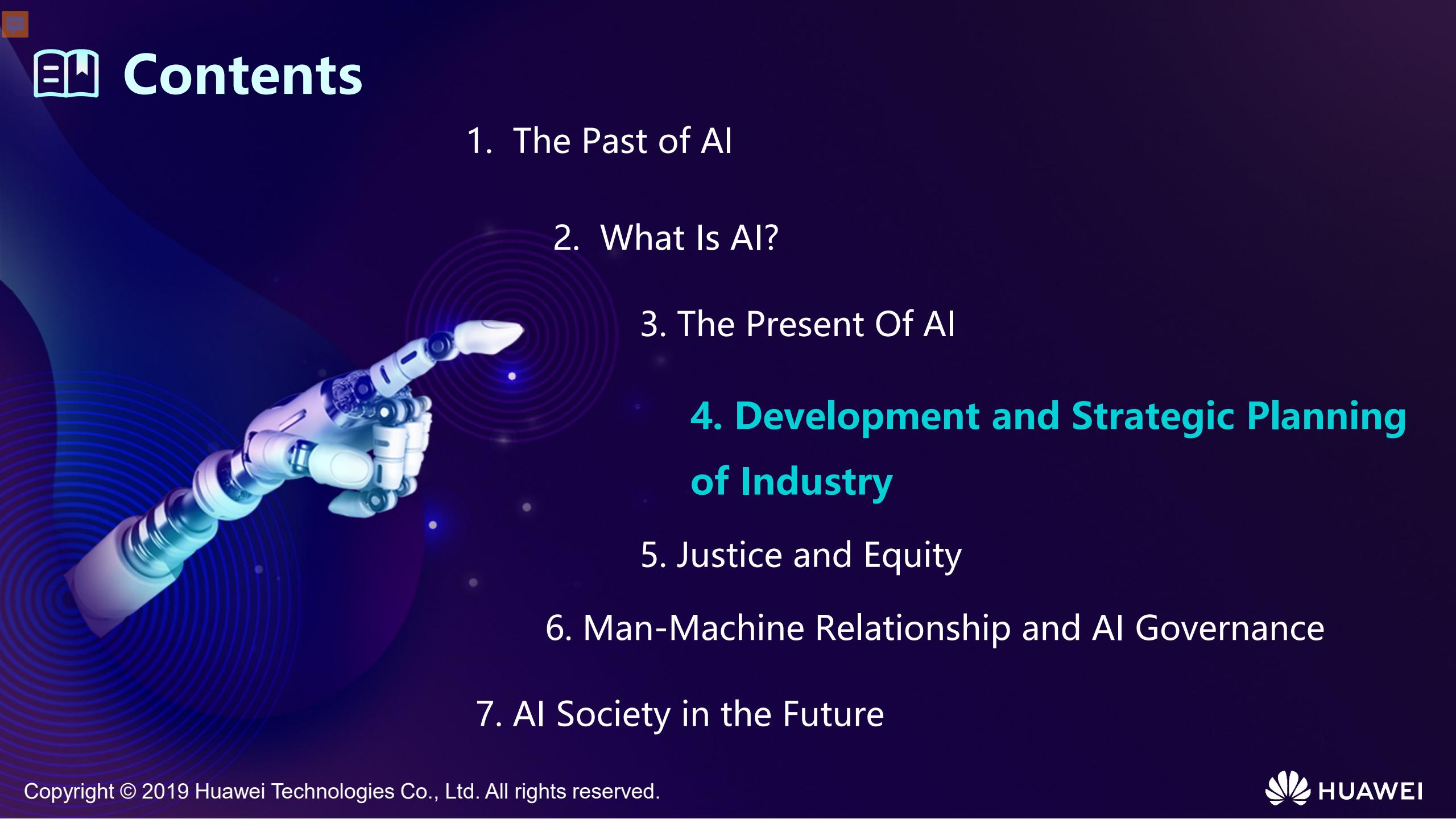


Quiz

1. At present, Natural language processing, Computer vision and Speech signal processing are the main directions of AI. ()
 - A. True
 - B. False



Contents

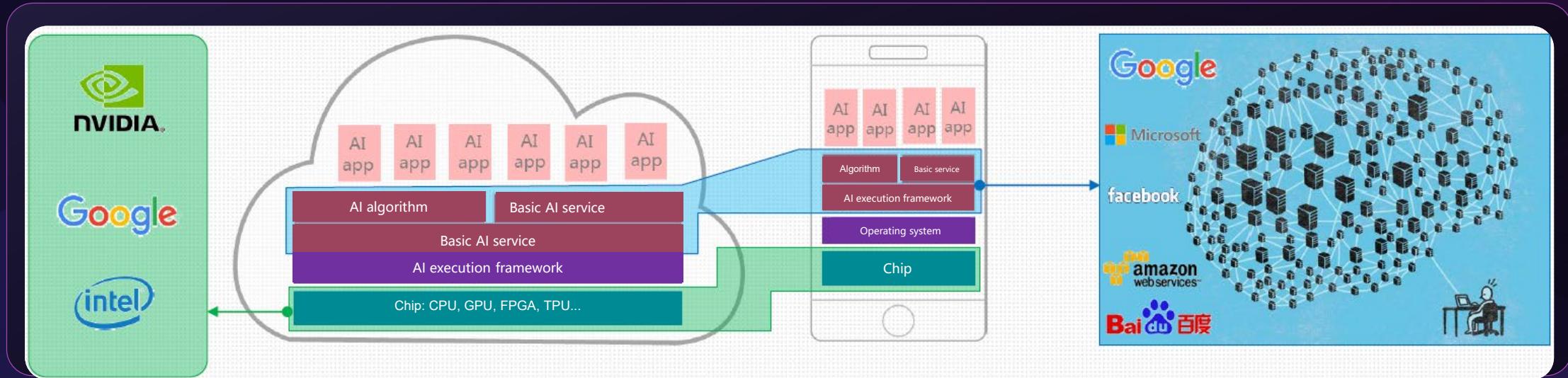
- 
1. The Past of AI
 2. What Is AI?
 3. The Present Of AI
 - 4. Development and Strategic Planning of Industry**
 5. Justice and Equity
 6. Man-Machine Relationship and AI Governance
 7. AI Society in the Future

Brain-like Research in the World

 Brain Initiative	 Human Brain Project	 Brain/MINDS
The US	EU	Japan
Brain Initiative: Exploration on how human brain works (initiated in 2013, US\$4.5 billion) SYNAPSE: Development of large-scale electronic neuromorphic computer prototypes (2008–2016)	Human Brain Project: Study on information communication technologies and healthcare in the future (initiated in 2013, EUR1 billion)	Brain/Minds: Study on a marmoset's brain to look into the brain functions and diseases (initiated in 2014, US\$270 million)



The Industry Landscape



- ◆ AI might lead to a change in chip architectures, which will further reshape the industry landscape. NVIDIA, Google, and Intel are competing for the dominant place in the future.
- ◆ Striving to be a leader of digital brains in the future becomes a strategic vision of information giants. Cloud services in the future might integrate cloud computing, big data, and AI.



Contents

- 
1. The Past of AI
 2. What Is AI?
 3. The Present Of AI
 4. Development and Strategic Planning
 - 5. Justice and Equity**
 6. Man-Machine Relationship and AI Governance
 7. AI Society in the Future

Who Takes Responsibilities?

At 22:00 on Sunday (March 19, 2018, local time) in Tempe, Arizona, an Uber's self-driving test car struck a 49-year old woman, Elaine Herzberg.



Self-driving Legislation



- ◆ In 2013, the U.S. National Highway Traffic Safety Administration (NHTSA) issued the Federal Automated Vehicles Policy.
- ◆ In August 2016, the United Nations Education, Scientific and Cultural Organization (UNESCO) and World Commission on the Ethics of Scientific Knowledge and Technology (COMEST) explored the possibility of robots in the Preliminary Draft Report of COMEST on Robotics Ethics.



How to Protect Privacy?

Two researchers from the University of Texas at Austin successfully identified two people out of the nearly half million anonymized users whose movie ratings were released by online rental company Netflix, which forced the company to cancel the movie-recommendation engine competition.



Data Protection

◆ Legislation:

- Swedish Data Protection

◆ Technical application:

Data anonymization:

It is the process of removing personally identifiable information from personal data, so that the people whom the data describes remain anonymous.



Is the Algorithm Fair?

- ◆ The algorithm model and input data, which determine the prediction results, are two main sources of algorithm discrimination. The following is a example:

Microsoft's AI chatbot, Tay, was taught to be an anti-semite, sexist, and racist after it was launched.





Issues To Be Resolved



Copyright ?



Who assign ?



What rights?



.....



Quiz

1. Now that self-driving technology is very mature, there is nothing to worry about too much. ()
 - A. True
 - B. False



Contents

- 
1. The Past of AI
 2. What Is AI?
 3. The Present Of AI
 4. Development and Strategic Planning of Industry
 5. Justice and Equity
 - 6. Man-Machine Relationship and AI Governance**
 7. AI Society in the Future



Three Generations of Robots



Generation 1:
Playback robot



Generation 2:
Robot with feelings



Generation 3:
Intelligent robot



Man-Machine Relationship



- There are worries that robots might pose threats to human beings. However, machines and human beings can coexist by controlling AI.
- AI becomes the agent of human consciousness. Human beings extend themselves through AI.
- The virtual reality will come true in the future.



Three Laws of Robotics

- ◆ In 1942, Isaac Asimov, a well-known American science fiction author, proposed the Three Laws of Robotics.
 - A robot may not injure a human being or, through inaction, allow a human being to come to harm.
 - A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.
 - A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.





AI Governance

- AI governance should be based on technological and industrial innovation.
- Regulators are advised to give more freedom to the market for innovation.
- Do not set too many constraints on the grounds of security.
- Strive to facilitate development and innovation.
- Encourage different entities to participate in the AI governance.



Contents

- 
1. The Past of AI
 2. What Is AI?
 3. The Present Of AI
 4. Development and Strategic Planning of Industry
 5. Justice and Equity
 6. Man-Machine Relationship and AI Governance
 - 7. AI Society in the Future**

Robot Colleagues



ASIMO

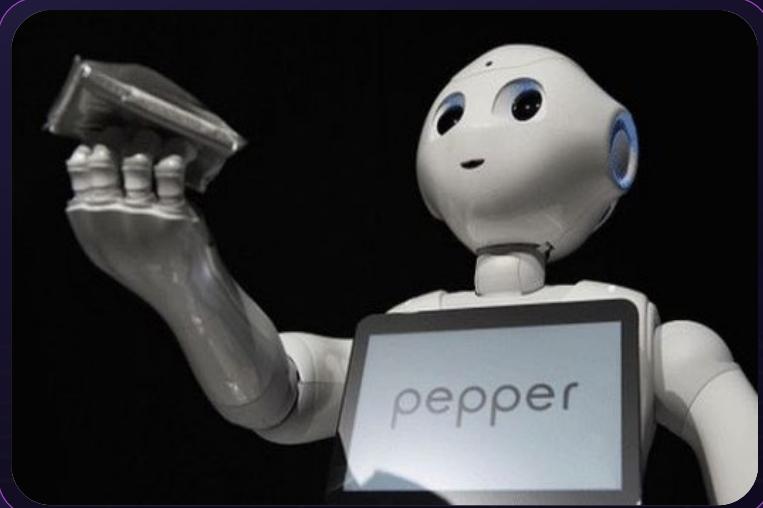


Waiter

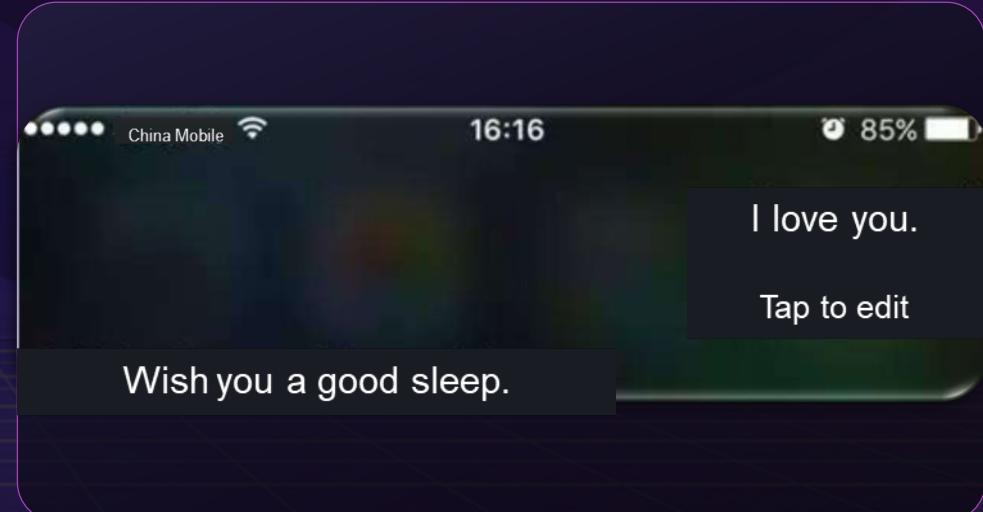
Soul Mate



Baymax

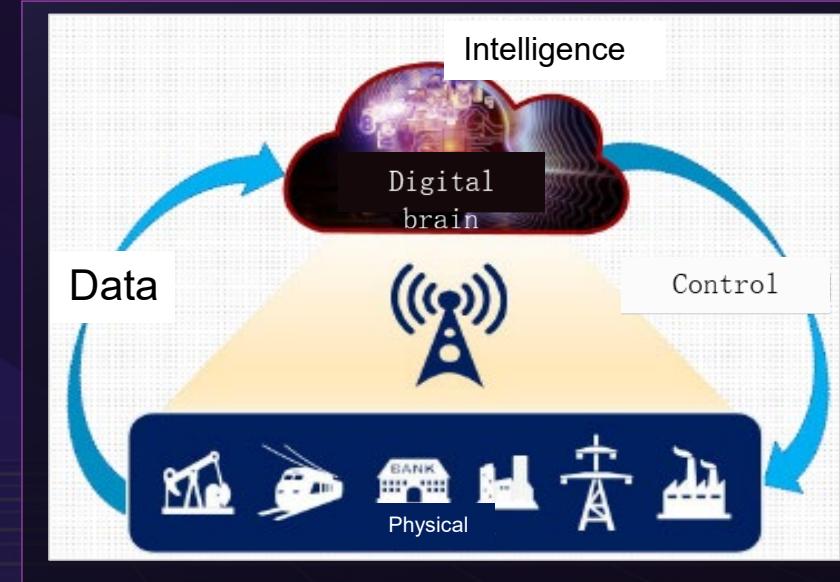


Pepper



Siri

Opportunities and Challenges of AI



Opportunities:

From efficiency to intelligence, AI will create a market larger than today's IT market (US\$2 trillion), which sparks a competition in the information industry.

Challenges:

In the entire industry chain, people who master intelligence will have greater say and gain more value. This is why traditional enterprises, such as GE, set up their own digital departments.



Quiz

1. What does AI stand for? ()

- A** Automatic Intelligence
- B** Artificial Intelligence
- C** Automatic Information
- D** Artificial Information

2. Which of the following theories does neural network research belong to? ()

- A** Symbolicism
- B** Connectionism
- C** Actionism
- D** None of the above



Quiz

3. In May 1997, a computer defeated Garry Kasparov, a former world chess champion, by 3.5:2.5. What's the name of this computer? ()

A Deep Blue

B Dark Green

C Deep Thinking

D Blue Sky

4. Who was the first to put forward the concept of AI in 1950 while proposing a machine intelligence test model? ()

A Marvin Minsky

B Zadeh

C Alan Turing

D John von Neumann



Summary

Now we've covered the past, present, and future of AI, AI technologies and development, as well as questions and problems to be thought in the AI era, such as justice and equity, man-machine relationship, and AI governance.

Thanks

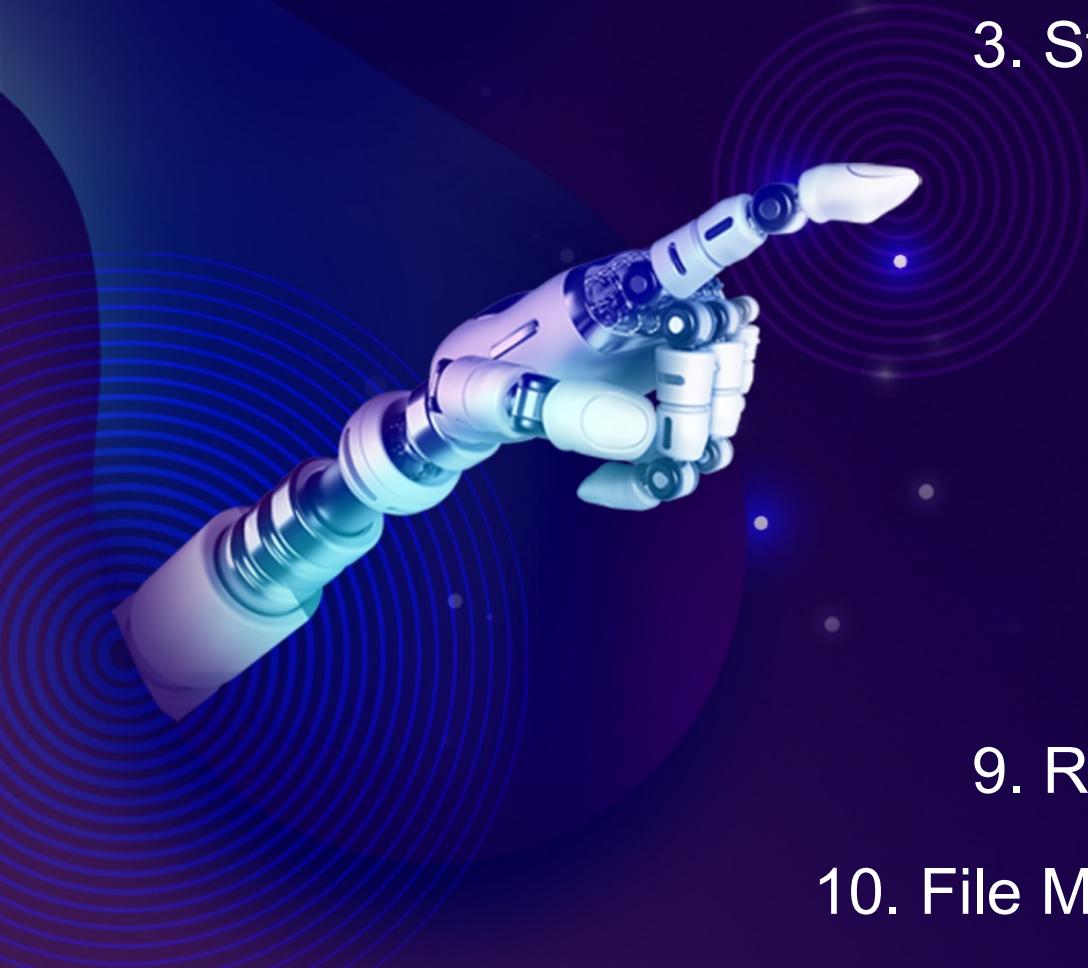
www.huawei.com





Python Programming Basics

Contents

- 
- 1. Introduction to Python**
 2. Lists and Tuples
 3. Strings
 4. Dictionaries
 5. Conditional and Looping Statements
 6. Functions
 7. Object-Oriented Programming
 8. Date and Time
 9. Regular Expressions
 10. File Manipulation

History of Python



Founder	Guido van Rossum
Degree	Master of Mathematics, Master of Computer Science
When and Where	Created in Amsterdam during Christmas in 1989.
Meaning of Name	A big fan of Monty Python's Flying Circus
Origin	Influenced by Modula-3, Python is a descendant of ABC that would appeal to Unix/C hackers.

Origin of Python

- ◆ Python is one of the achievements of free software.
- ◆ Python is purely free software and both its source code and interpreter comply with the GNU General Public License (GPL) protocol.

Philosophy of Python:

- ◆ Python is engineering but not art.
- ◆ There should be one, and preferably only one, obvious way to do it.
- ◆ Simple is better than complex, and explicit is better than implicit.

What is Python?

- ◆ Python is a programming language.
- ◆ Python is a general-purpose and advanced programming language.
- ◆ Python applies to programming in many fields:



Data
science



Writing
system tools



Developing applications
with graphical UIs



Writing
network-based software



Interacting
with databases



Features of Python Statements

- ◆ Dynamic: Objects (like attributes and methods) can be changed during execution.
- ◆ Python uses indentation instead of a pair of curly braces { } to divide a block of statements.
- ◆ Multiple statements on one line are separated by “;”.
- ◆ The symbol used for commenting out a line is #, and doc string ("... ") is used to comment out multiple lines.
- ◆ Variables do not need type definitions.
- ◆ Functional Programming (FP) is available.

```
    "taxes" : >>>
    $taxes = $taxes;
    $tax_order = array();
    $sort_order = $this.$model_extension_extension->getExtensions('total');
    foreach ($results as $key => $value) {
        if (!isset($value['code'])) {
            $code = $value['code'];
        } else {
            $code = $value['key'];
        }
        $sort_order[$key] = $this->config->get($code . '_sort_order');
    }
    array_multisort($sort_order, SORT_ASC, $results);
    foreach ($results as $result) {
        if (!isset($result['code'])) {
            $code = $result['code'];
        } else {
            $code = $result['key'];
        }
        if ($this->config->get($code . '_status')) {
            $this->load->model('extension/total/' . $code);
            // We have to put the totals in an array so that they pass
            // by reference.
            $this->model_extension_total->getTotal($total_data);
            if (!empty($total_data[count($total_data) - 1]) && !isset($total_data[count($total_data) - 1]['code'])) {
                $total_data[count($total_data) - 1]['code'] = $code;
            }
            $tax_difference = 0;
            foreach ($taxes as $tax_id => $value) {
                if ($tax_id == $code) {
                    $tax_difference = $value;
                }
            }
            $this->model_extension_total->getTotal($total_data);
            if (!empty($total_data[count($total_data) - 1]) && !isset($total_data[count($total_data) - 1]['code'])) {
                $total_data[count($total_data) - 1]['code'] = $code;
            }
            $total_data[count($total_data) - 1]['difference'] = $tax_difference;
        }
    }
}

Carousel.prototype.getSlideForDirection = function (direction, active) {
    var delta = direction == 'prev' ? -1 : 1;
    var activeIndex = this.getItemIndex(active);
    var itemIndex = (activeIndex + delta) % this.$items.length;
    return this.$items.eq(itemIndex)
}

Carousel.prototype.to = function (pos) {
    var that = this;
    var activeIndex = this.getItemIndex(this.$active);
    this.$element.find('.item').eq(pos).addClass('active');
    if (pos > (this.$items.length - 1) || pos < 0) return;
    if (this.sliding) return this.$element.one('slid.bs.carousel', function () {
        if (activeIndex == pos) return this.pause().cycle();
    });
    return this.slide(pos > activeIndex ? 'next' : 'prev', this.$items.eq(pos));
}

Carousel.prototype.pause = function (e) {
    e || (this.paused = true);
    if (this.$element.find('.next, .prev').length && $.support.transition) {
        this.$element.trigger($.support.transition.end);
        this.cycle(true);
    }
}

this.interval = clearInterval(this.interval)
```

Python VS C and Shell

◆ Python VS C :

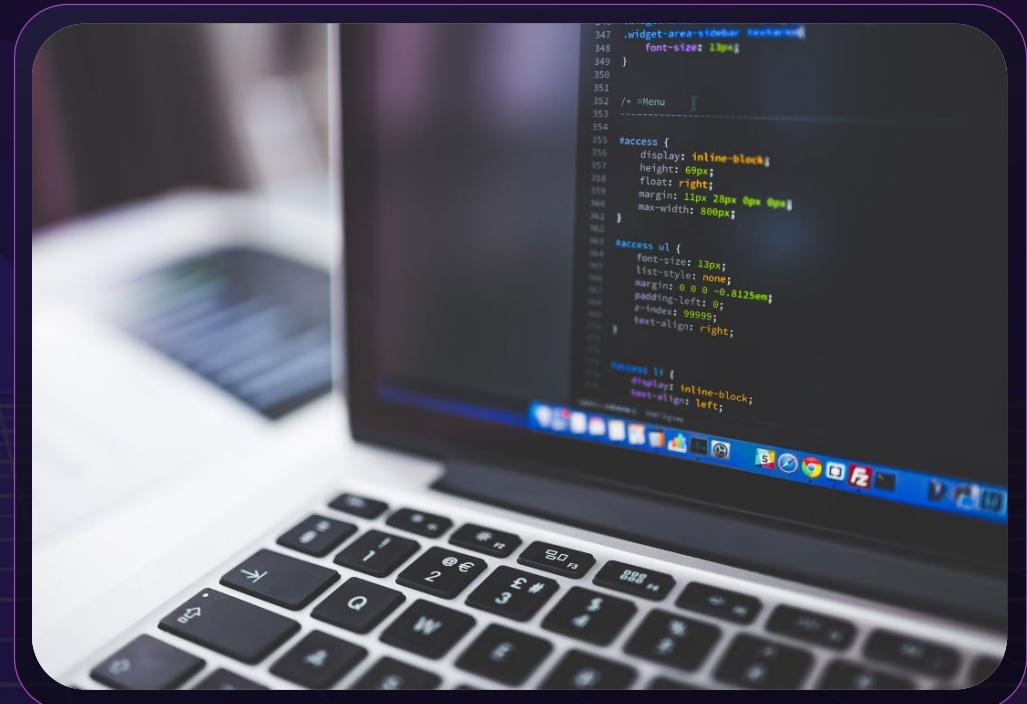
- Python is dynamic while C is static.
- Memory is managed by the developer in C but by the interpreter in Python.
- Python has many libraries but C has no standard library for a hybrid tuple (Python list) and a hash table (Python dictionary).
- Python cannot be used to write a kernel but C can.
- C or C++ extends Python functions based on the Python APIs.

◆ Python VS Shell :

- Python has simple syntax and is easy to transplant.
- Shell has a longer script.
- Python can reuse code and embraces simple code design, advanced data structure, and modular components.

Python VS Java

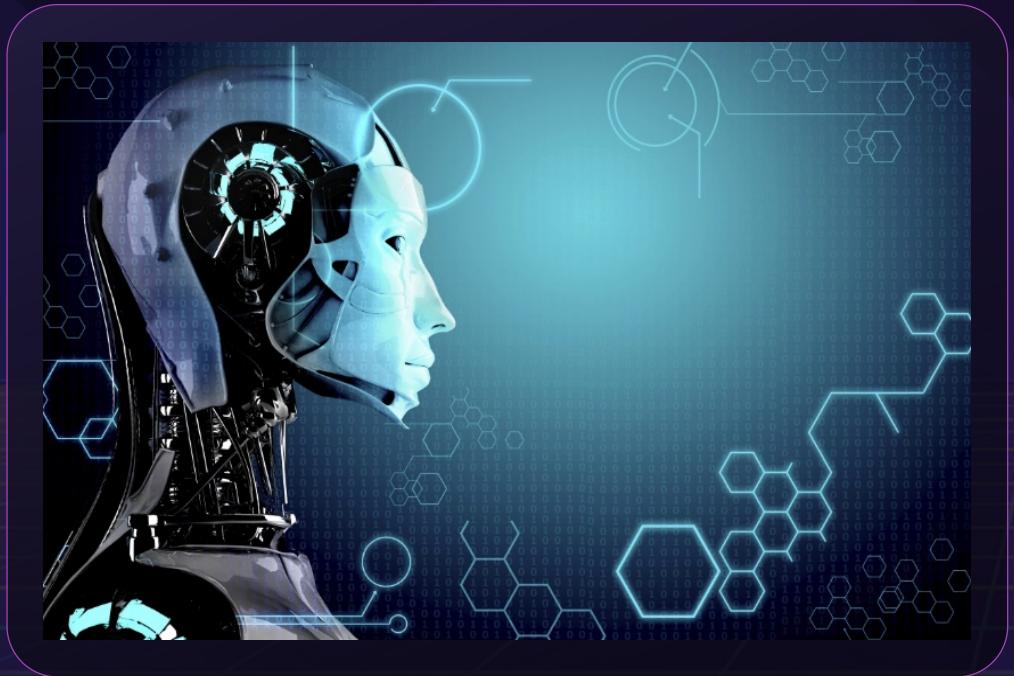
- Python is dynamic while Java is static.
- Python supports object-oriented and function-based programming while Java supports object-oriented programming only.
- Python is simpler than Java, and typically applies to quick prototyping.
- Python and Java enable multiple programmers to develop a large project together step by step.



Advantages of Python

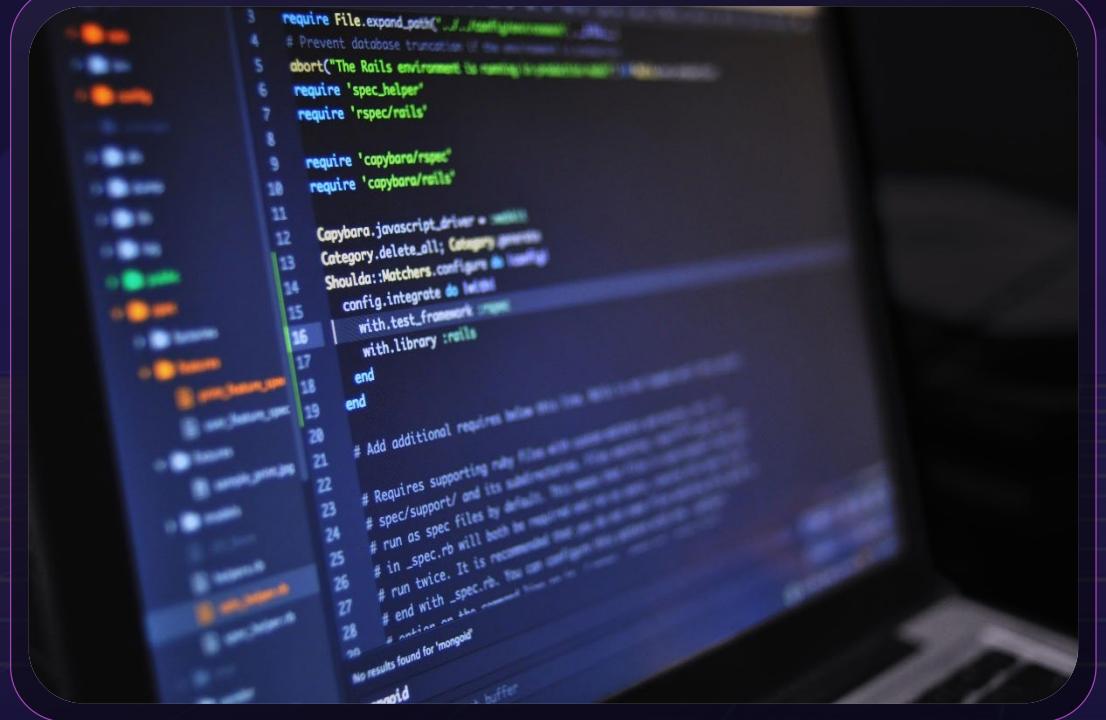


Python Application Domains



- ◆ Python has rich third-party libraries and advantages. Therefore, Python can be used in many fields as follows:
 - AI
 - Data science
 - System tool development
 - Apps
 - Automatic O&M scripts
 - Web development
 - ...

Development Environment of Python



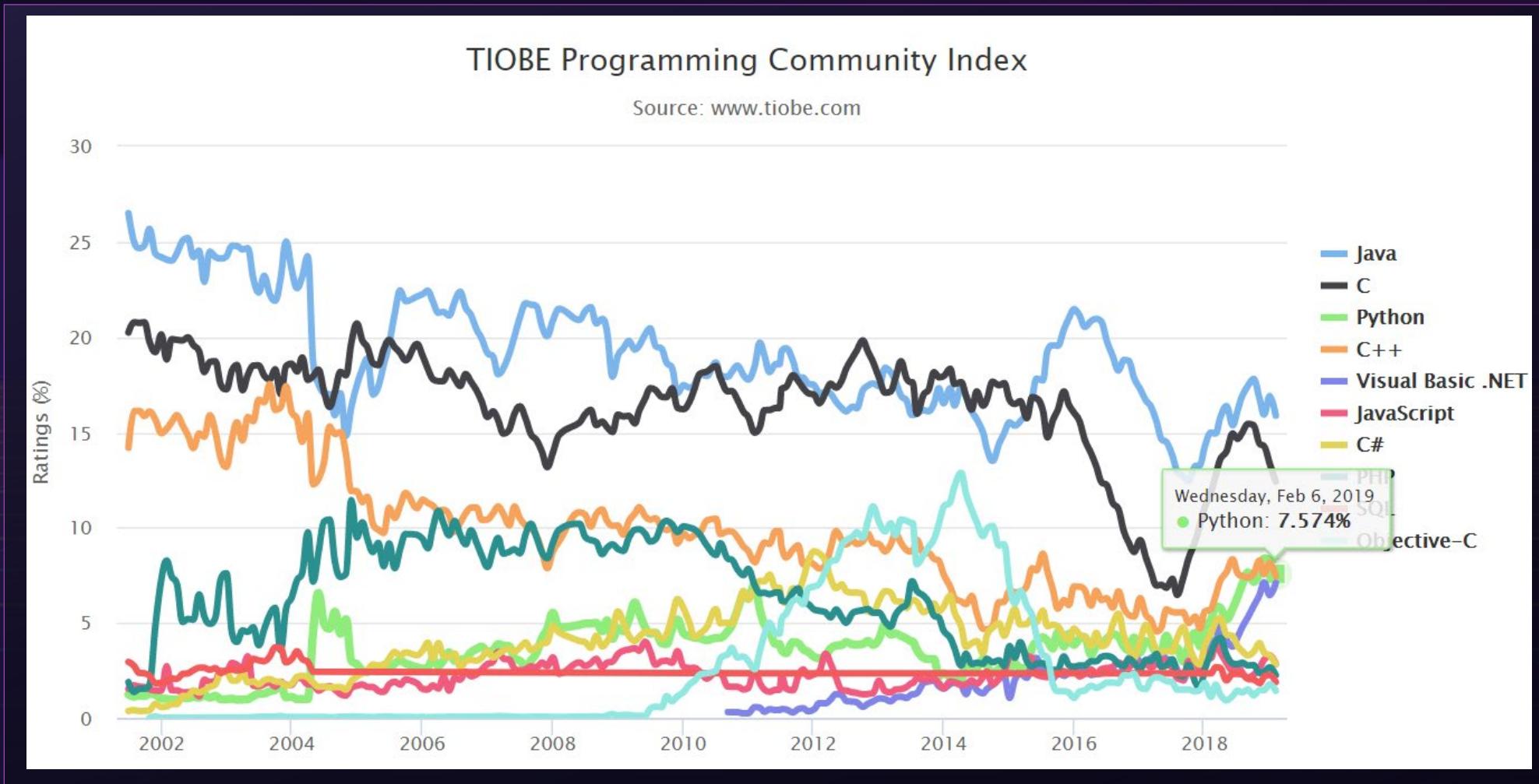
- ◆ VIM
- ◆ IDLE
- ◆ Sublime Text
- ◆ Eclipse
- ◆ Eric4
- ◆ Boa
- ◆ WingIDE
- ◆ Other editors:
notepad++,
editplus



- ◆ Vim is a lightweight IDE.
- ◆ If you do not install too many plug-ins or plug-in performance is good, using VIM for development has a low requirement on hardware.
- ◆ Vim can achieve a consistent programming experience on local and remote servers.
- ◆ Vim has an editing speed of "what you think is what you have".



Popularity Ranking



Python 2 VS Python 3 (1)

- ◆ Python 3 cannot be backwards compatible with Python 2, which requires people to decide which version of the language is to be used.
- ◆ Many libraries are only for Python 2, but the development team behind Python 3 has reaffirmed the end of support for Python 2, prompting more libraries to be ported to Python 3.
- ◆ Judging from the number of Python packages that are supported by Python 3, Python 3 has become increasingly popular.



Python2 VS python3 (2)



Installing Python (1)

For Linux users:

- ◆ Download the Python package and install it.

- ◆ Create soft connections.

- ◆ Verify installation.

```
$tar -zxf python3.6.4.tar.gz
```

```
$cd Python3.6.4
```

```
$./configure
```

```
$make && make install
```

```
$mv /usr/bin/python /usr/bin/python.bak
```

```
$ln -s /usr/local/bin/python3.6.4
```

```
/usr/bin/python
```

```
$python -V
```

Installing Python (2)

For Windows users:

- ◆ Download the official Python setup program.
- ◆ Select the latest Python Windows installer to download the .exe installation file.
- ◆ Double-click the setup program, Python-3.x.exe.
- ◆ Add environment variables: Choose My Computer > Properties > Advanced > Environment Variables, and enter your Python installation location in path.
- ◆ Verify installation: Start > Program > Python 3.x > Start Python command line, and then enter:
`print("Hello World")`. If the output is "Hello World", it indicates that the installation was successful.

Starting Python

Linux:

```
[root@yaokaiqiangzjhw ~]# python3
Python 3.6.4 (default, Apr  9 2018, 13:51:42)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-16)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>>
```

Windows:

```
C:\Users\ywx516714>python
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>>
```

Executing a Python Program

Using command lines:

- ◆ Linux:
 - Enter a Python command in the Linux command line.
- ◆ Window :
 - Enter a Python command on the DOS prompt.

Using scripts:

- ◆ Store Python Statements in a script file and execute it in the command line.

Input: `python hello.py`

Output: `hello world !`





Quiz

1. Python is a dynamic interpretive language.()

A: True

B: False

Contents

- 
1. Introduction to Python
 - 2. Lists and Tuples**
 3. Strings
 4. Dictionaries
 5. Conditional and Looping Statements
 6. Functions
 7. Object-Oriented Programming
 8. Date and Time
 9. Regular Expressions
 10. File Manipulation

Lists

- ◆ List is expressed using [].
- ◆ A list is an ordered set of elements that you can add and delete at any time.
- ◆ The element types in the list can be different. You can access each element in the list by index. The first digit of the index starts from 0, and the reverse index starts from -1.



Common Operations on Lists

Access

Update (append and insert)



Delete elements (del)

Operator on list script (+/*)

List interception



Functions of Python Lists



`cmp(list1,list2)`



`len(list)`



`max(list)`



`min(list)`



`list(seq)`



Methods of Python Listing

1

`list.append(obj)`

2

`list.count(obj)`

3

`list.extend(seq)`

4

`list.index(obj)`

5

`list.insert(index, obj)`

6

`list.pop(obj=list[-1])`

7

`list.remove(obj)`

8

`list.sort([func])`

9

`list.reverse()`

Tuples

- ◆ Tuple is expressed using ().
- ◆ A tuple is simple to create. You only need to add elements to parentheses and separate them with commas.
- ◆ Like a list, a tuple cannot be modified once initialized, and the elements need to be identified when a tuple is defined.
- ◆ A tuple does not have the append () or insert () method, nor can it be assigned to another element. It has the same fetching methods as a list.
- ◆ Because a tuple is unchangeable, the code is more secure. Therefore, if possible, use a tuple instead of a list.

Common Operations on Tuples

Access

Modify
(tuple calculation)

Delete
(del tuple)

Tuple operators
 $(+, *)$

Tuple index and
interception

No-close
separator



Embedded Functions of Tuples



`cmp(tuple1, tuple2)`



`len(tuple)`



`max(tuple)`



`min(tuple)`



`tuple(seq)`



Quiz

1. List is changeable, but Tuple is unchangeable. ()

A: True

B: False

Contents

- 
1. Introduction to Python
 2. Lists and Tuples
 - 3. Dictionaries**
 4. Strings
 5. Conditional and Looping Statements
 6. Functions
 7. Object-Oriented Programming
 8. Date and Time
 9. Regular Expressions
 10. File Manipulation



Dictionaries

- ◆ A dictionary is another variable container model and can store any type of object.
- ◆ Each key value of the dictionary is separated with a colon ":" key value pairs are separated by a comma "," and the entire dictionary is included in the curly braces "{}".
- ◆ The key is generally unique, and the type of the key is unchangeable. If the key repeats, the last key-value pair replaces the previous one. Key values do not need to be unique, and can take any data type.
- ◆ A dictionary has the following format
 - $d = \{key1 : value1, key2 : value2\}$



Python Dictionary Operations



Access



Modify



Delete

Built-in Functions of Dictionaries



`cmp(dict1, dict2)`



`len(dict)`



`str(dict)`



`type(variable)`



Built-in Methods of Dictionaries

01

`has_key(x)`

02

`keys()`

03

`values()`

04

`items()`

05

`clear()`

06

`copy()`

07

`update(x)`

08

`get(x[,y])`

09

`pop()`

10

`popitem()`



Quiz

1. There can be duplicate keys in the dictionary, but there can be no duplicate values. ()

A: True

B: False

Contents

- 
1. Introduction to Python
 2. Lists and Tuples
 3. Dictionaries
 - 4. Strings**
 5. Conditional and Looping Statements
 6. Functions
 7. Object-Oriented Programming
 8. Date and Time
 9. Regular Expressions
 10. File Manipulation



Definition of a String

```
252     document.getElementById(bigImageDesc).innerHTML = descriptions[page * 9 + i - 1];
253 }
254
255 function updatePhotoDescription() {
256     if (descriptions.length > (page * 9) + (currentImage - 1)) {
257         document.getElementById(bigImageDesc).innerHTML = descriptions[page * 9 + currentImage - 1];
258     }
259 }
260
261 function updateAllImages() {
262     var i = 1;
263     while (i < 10) {
264         var elementId = 'foto' + i;
265         var elementIdBig = 'bigImage' + i;
266         if (page * 9 + i - 1 < photos.length) {
267             document.getElementById(elementId).src = 'images/' + photos[page * 9 + i - 1];
268             document.getElementById(elementIdBig).src = 'images/' + photos[page * 9 + i - 1];
269         } else {
270             document.getElementById(elementId).src = '';
```

- ◆ In Python, a string is a sequence of 0 or more characters, and a string is one of several sequences built in Python.
- ◆ In Python, strings are unchangeable, and are similar to string constants in the C and C++ languages.
- ◆ Python strings may be expressed using single quotes, double quotes and triple quotes, as well as escape characters, raw strings, and so on.
 - name='JohnSmith'
 - name="Alice"
 - name="""Bob"""

String Formatting (1)

- ◆ Python supports the output of formatted strings. Although a complex expression may be used, the most basic use is to insert a value into a string.
- ◆ String formatting in Python is accomplished by the string formatting operator (%), and its string conversion type table and its formatting operator auxiliary instructions are shown in the following tables.

```
Input: print("My name is %s and age is %d !" %(‘AI’, 63))
```

```
Output: My name is AI and age is 63 !
```

String Formatting (2)

Format	Description
%c	Character and its ASCII code
%s	String
%d	Signed integer (decimal)
%u	Unsigned integer (decimal)
%o	Unsigned integer (octal)
%x	Unsigned integer (hexadecimal)
%X	Unsigned integer (hexadecimal upper-case letters)
%e	Floating number (scientific counting method)
%E	Floating number (scientific counting method, E replacing e)
%f	Floating number (decimal point sign)
%g	Floating number (%e or %f, depending on a value)
%G	Floating number (similar to %g)
%p	Pointer (print memory address of a value using hexadecimal)

String Formatting (2)

Symbol	Function
*	Defines width or precision of the decimal point.
-	Aligns to the left.
+	Displays + before a positive value.
<sp>	Displays space before a positive value.
#	Displays 0 before an octal number or 0x or 0X before a hexadecimal value (depending on whether x or X is used).
0	Adds 0 instead of default space before numbers.
%	%% outputs a single %.
(var)	Maps variables (dictionary arguments).
m.n	m means the minimum width and n means the number of digits after the decimal point.



String Operators

- ◆ Python does not have dedicated Char type, a string is the sequence of a character and one character is a string with a length of 1. Python strings are not changeable and do not end with '\0'. and is stored in memory as follows:

	P	y	t	h	o	n
Superscript	0	1	2	3	4	5
Subscript	-6	-5	-4	-3	-2	-1



String Methods

```
>>> dir("")  
['__add__', '__class__', '__contains__', '__delattr__', '__doc__', '__eq__', '__ge__',  
'__getattribute__', '__getitem__', '__getnewargs__', '__getslice__', '__gt__', '__hash__',  
'__init__', '__le__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__',  
'__reduce__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__',  
'__str__', 'capitalize', 'center', 'count', 'decode', 'encode', 'endswith', 'expandtabs', 'find',  
'index', 'isalnum', 'isalpha', 'isdigit', 'islower', 'isspace', 'istitle', 'isupper', 'join', 'ljust',  
'lower', 'lstrip', 'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split',  
'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']  
>>>
```



String Modules

```
>>> import string
>>> dir(string)
['Template', '_TemplateMetaclass', '__builtins__', '__doc__', '__file__', '__name__',
 '_float', '_idmap', '_idmapL', '_int', '_long', '_multimap', '_re', 'ascii_letters',
 'ascii_lowercase', 'ascii_uppercase', 'atof', 'atof_error', 'atoi', 'atoi_error', 'atol',
 'atol_error', 'capitalize', 'capwords', 'center', 'count', 'digits', 'expandtabs', 'find',
 'hexdigits', 'index', 'index_error', 'join', 'joinfields', 'letters', 'ljust', 'lower', 'lowercase',
 'lstrip', 'maketrans', 'octdigits', 'printable', 'punctuation',
 'replace', 'rfind', 'rindex', 'rjust', 'rsplit', 'rstrip', 'split', 'splitfields', 'strip', 'swapcase', 'translate',
 'upper', 'uppercase', 'whitespace', 'zfill']
>>>
```



Quiz

1. What format of data does “%s” represent? ()

- A: Signed integer
- B: Unsigned integer
- C: Floating number
- D: String

Contents

- 
1. Introduction to Python
 2. Lists and Tuples
 3. Dictionaries
 4. Strings
 5. **Conditional and Looping Statements**
 6. Functions
 7. Object-Oriented Programming
 8. Date and Time
 9. Regular Expressions
 10. File Manipulation

if Statements

- ◆ Python supports three control structures: if, for, and while, but it does not support switch statements in the C language.
- ◆ In Python programming, if statements are used to control execution of control programs, and the basic form is:

```
if Judging condition 1:  
    Statement 1...  
  
elif Judging condition 2:  
    Statement 2...  
  
elif Judging condition 3:  
    Statement 3...  
  
else:  
    Statement 4...
```



while Statements

- ◆ The while statement in the Python language is used to execute a loop program that, under certain conditions, loops through a program to handle the same tasks that need to be repeated.
- ◆ When the condition of a while statement is never a Boolean false, the loop will never end, forming an infinite loop, also known as a dead loop. You can use the break statement in a loop to force a dead loop to end.
- ◆ How to use a while statement:

```
age = 0
while (age < 100):
    print("The age is:", age)
    age = age + 1
print("Good bye!")
```

for Statements

- ◆ In the Python language, the for loop can traverse any items of a sequence, such as a list, a dictionary, or a string.
- ◆ The for statement is different from a traditional for statement. The former accepts an iterative object (such as a sequence or iterator) as its argument, and one element is iterated each time.

```
ages=[6,1,60]
for num in ages:
    if num == 6:
        print("He is a boy, his age is %d" % (num))
    elif num == 60:
        print("He is a oldman, his age is %d" % (num))
    else:
        print("He is a baby, his age is %d" % (num))
```



Loop Nesting

```
for iterating_var in sequence:  
    for iterating_var in sequence:  
        statements(s)  
    statements(s)
```

```
while expression:  
    while expression:  
        statement(s)  
    statement(s)
```



break and continue

- ◆ A break statement ends the entire loop, and if a break statement is triggered, the loop else is not executed.
- ◆ A continue statement ends the ongoing iteration of the loop, and begins the next iteration.
- ◆ If you use a nested loop, the break statement stops executing the deepest loop and starts executing the next line of code.
- ◆ The continue statement tells Python to skip the remaining statements of the current loop to proceed to the next round of loops.
- ◆ Both the break and continue statements are available in the while and for loops.



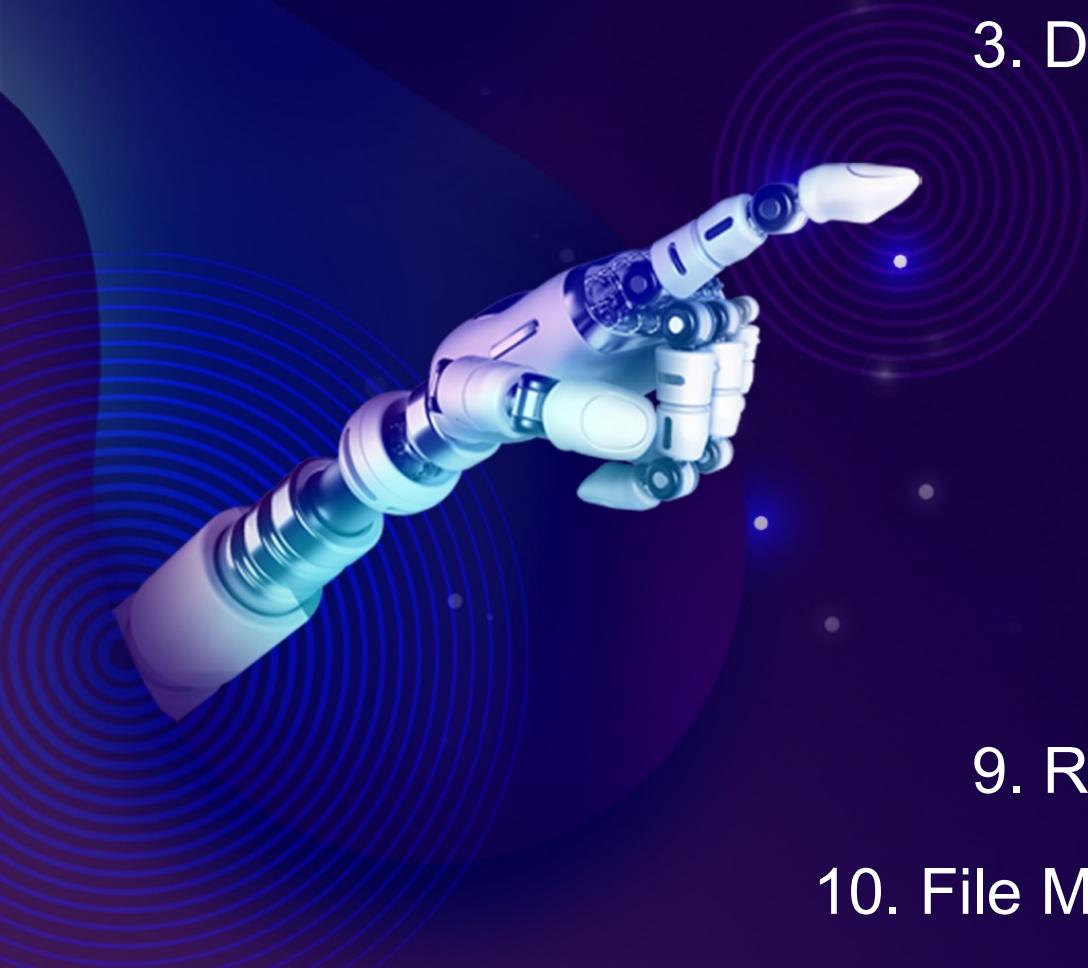
Quiz

1. Both “while” and “for” statements belong to looping statements. ()

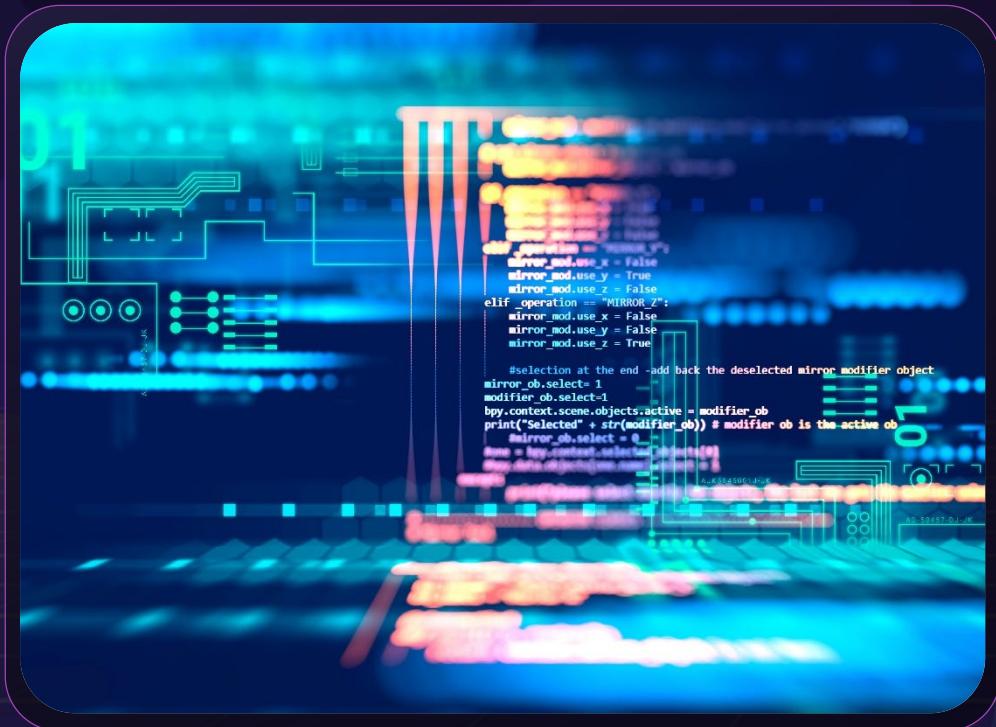
A: True

B: False

Contents

- 
1. Introduction to Python
 2. Lists and Tuples
 3. Dictionaries
 4. Strings
 5. Conditional and Looping Statements
 - 6. Functions**
 7. Object-Oriented Programming
 8. Date and Time
 9. Regular Expressions
 10. File Manipulation

Python Functions



- ◆ A function is a code segment that is organized, reusable, and used to implement a single function or associated functions.
- ◆ Functions can improve the modularity of applications and reuse of code.
- ◆ Python provides a number of built-in functions, such as `print()`. You can also create your own functions, which are called user-defined functions.

Defining a Function

- ◆ Define a function with the following rules:
- ◆ The function code block begins with a def keyword, followed by the function name and parentheses ().
- ◆ Any incoming arguments and independent variables must be placed in the middle of the parentheses. Parentheses can be used to define arguments.
- ◆ The first line of the function can selectively use the document string to hold the description of the function.
- ◆ The function content starts with a colon and indents.
- ◆ return[expression] ends a function, and selectively returns a value to the caller. Returning without an expression is equivalent to returning none.



Calling a Function

```
# Define a function
def test(str):
    "print any incoming string"
    return str

# Call a function
test("I want to call a user-defined function!")
test("call the same function again")
```

Transferring Arguments

- ◆ In Python, a type belongs to an object, and a variable is of no type.

```
a = [1,2,3]  
a = "Huawei"
```

- ◆ In the above code, [1,2,3] is the list type, "Huawei" is a string type, and the variable a is of no type, which is only a reference (a pointer) to an object, and can be a list type object, or a string type object.

Argument Types



Essential argument



Keyword argument



Default argument



Indefinite length
argument

Anonymous Functions

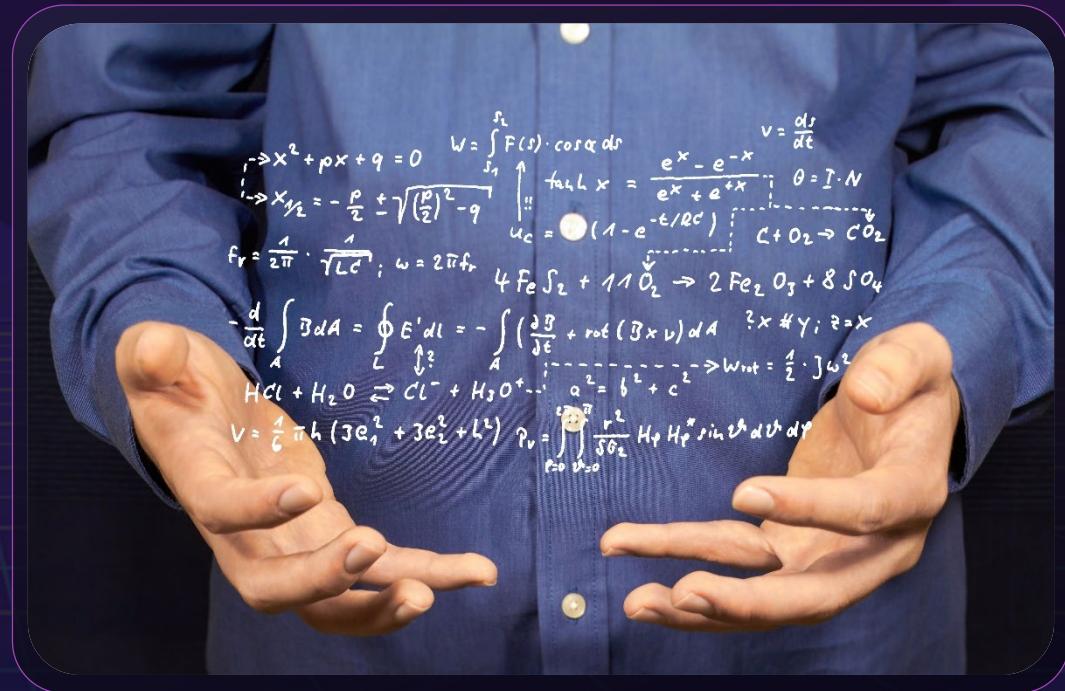


```
$default_language = "en";  
/* Try to figure out which language to use.  
 */  
function negotiate_language($lang) {  
    global $supported_languages, $HTTP_ACCEPT_LANGUAGE;  
  
    if (isset($supported_languages[$lang])) {  
        return $lang;  
    }  
  
    /* If the client's language header  
     * see if it is supported.  
     */  
    if ($HTTP_ACCEPT_LANGUAGE) {  
        $accepted_languages = explode(',', $HTTP_ACCEPT_LANGUAGE);  
        for ($i = 0; $i < count($accepted_languages); $i++) {  
            if ($accepted_languages[$i] == $lang) {  
                return $lang;  
            }  
        }  
    }  
  
    /* One last desperate try: check for valid language code in the  
     * top-level domain of the client's IP address.  
     */  
    if (eregi( "\.\[^\.]+\\"", $REMOTE_HOST, $arr)) {  
        $lang = strtolower($arr[1]);  
        if ($supported_languages[$lang]) {  
            return $lang;  
        }  
    }  
  
    global $default_language;  
    return $default_language;
```

- ◆ lambda is only an expression
- ◆ The body of lambda is an expression
- ◆ A lambda function has its own namespace
- ◆ Although a lambda function may seem to write only one line

Global Variables and Local Variables

- ◆ A variable defined within a function has a local scope and is called a local variable.
- ◆ a variable defined beyond a function has a global scope and is called a global variable.





OS-Related Calling and Functions - sys



sys.argv



sys.stdout
sys.stdin
sys.stderr



sys.stdin.readline()



sys.stdout.write("a")



sys.exit()



sys.modules



sys.platform



sys.path



OS-Related Calling and Functions - os



`os.environ`



`os.environ
["HOME"]`



`os.chdir(dir)`



`os.chdir
(‘d:\\\\outlook’)`



`os.getcwd()`



`os.getegid()`



`os.getgid()`



`os.getuid()`



`os.geteuid()`



`os.setegid()
os.seteuid()`



`os.getgruops()
os.getlogin()`



`os.getenv()`



`os.putenv()`



`os.umask()`



`os.system(cmd)`



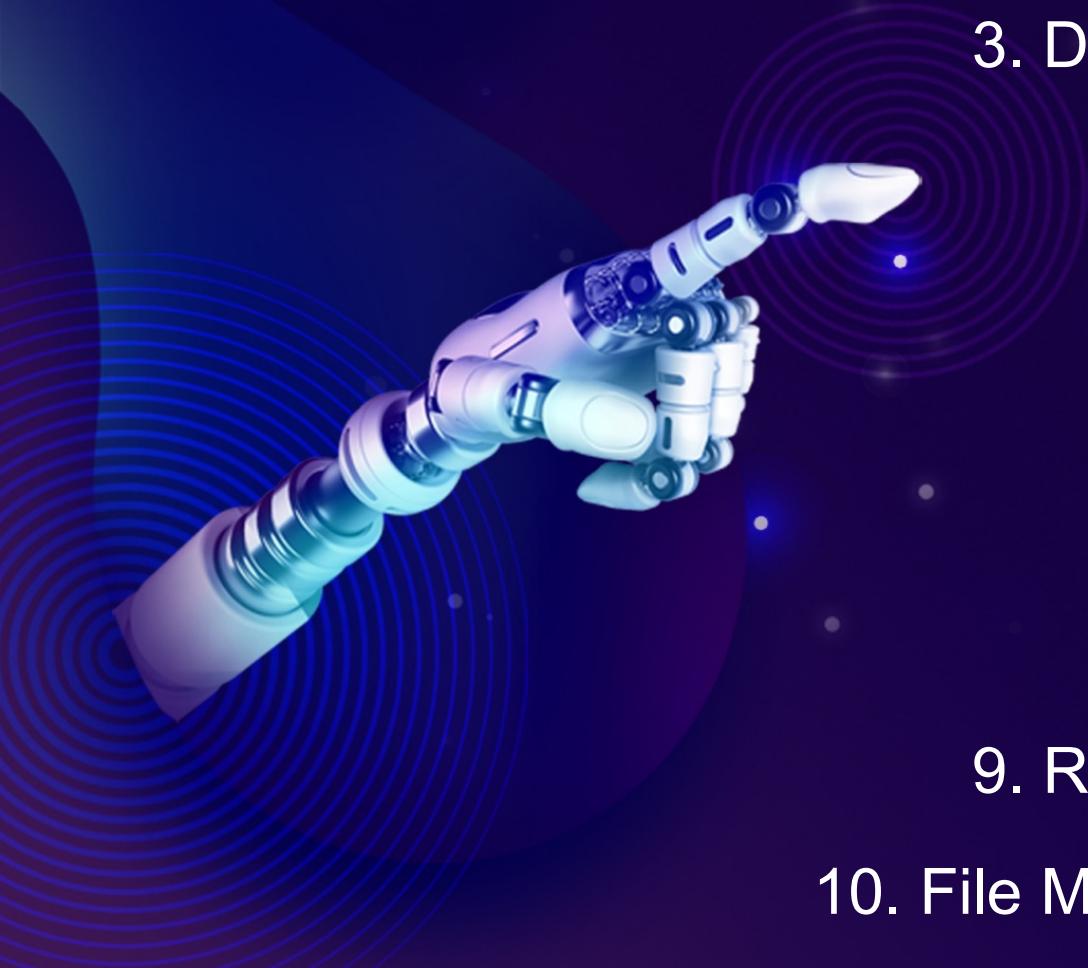
Quiz

1. All functions in Python must need the return value. ()

A: True

B: False

Contents

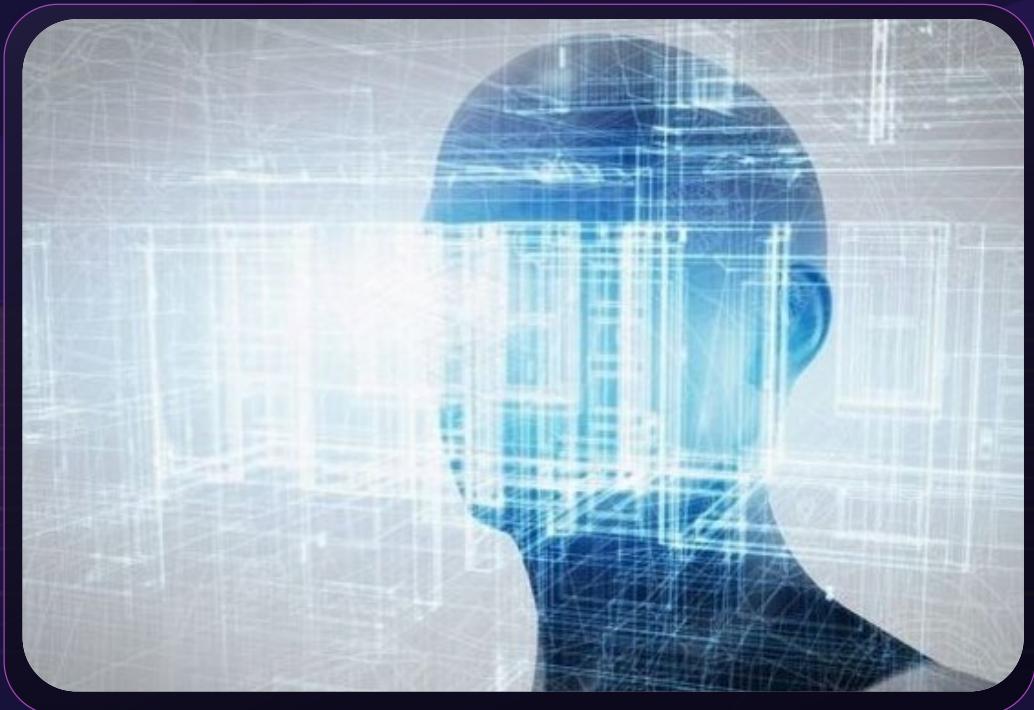
- 
1. Introduction to Python
 2. Lists and Tuples
 3. Dictionaries
 4. Strings
 5. Conditional and Looping Statements
 6. Functions
 - 7. Object-Oriented Programming**
 8. Date and Time
 9. Regular Expressions
 10. File Manipulation



Object-Oriented Programming

- ◆ OOP takes objects as the basic units of a program, and an object contains data and functions that manipulate data.
- ◆ Process-oriented programming treats a computer program as a series of command sets, that is, the sequential execution of a set of functions.
- ◆ OOP treats computer programs as a collection of objects, and each object can receive messages from other objects and process them.

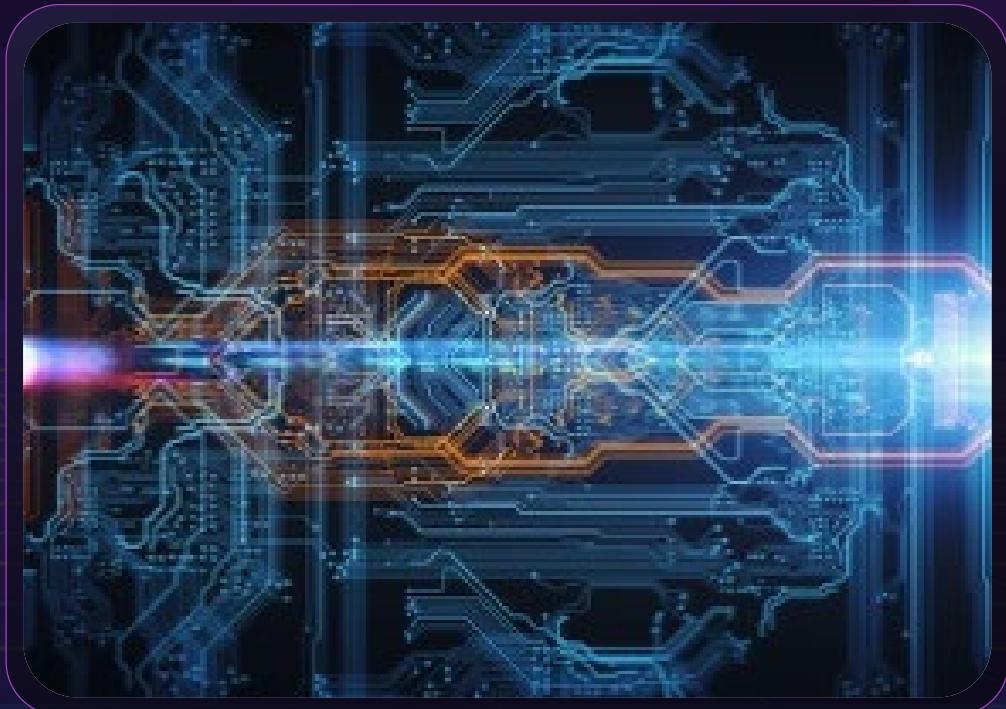
Object-Oriented Design Philosophy



- ◆ The idea of object-oriented design (OOD) derives from nature, because the concepts of class and instance are natural.
- ◆ Class is an abstract concept.
- ◆ OOD has a higher abstraction than function because a class contains data and methods of manipulating data.

Relationship Between OOD and OOP

- ◆ OOD does not specifically require OOP languages.
- ◆ An OOP language does not necessarily force you to write OO-related programs.
- ◆ In Python, classes and OOP are not necessary for daily programming.



Common Python OOP Terms

Abstract
Implementation

Encapsulation
Interface

Composition

Derivation
Inheritance
Inheritance Structure

Generalization
Specialization

Polymorphism

Introspection
Reflection



Classes

- ◆ A class is a data structure that can be used to define objects.
- ◆ In Python, a class statement is similar to a function statement, as follows:

```
def functionName(args):
    'function documentation string'
    function_suite

class ClassName(object):
    'Click class documentation string'
    class_suite
```

Class methods

Inheritance

Composition

Derivation

Subclasses

Privatization



Quiz

1. Python is an object-oriented programming language, everything in python is an object. ()

A: True

B: False

Contents

- 
1. Introduction to Python
 2. Lists and Tuples
 3. Dictionaries
 4. Strings
 5. Conditional and Looping Statements
 6. Functions
 7. Object-Oriented Programming
 - 8. Date and Time**
 9. Regular Expressions
 10. File Manipulation



Getting the Current Date and Time

```
>>> from datetime import datetime  
>>> now = datetime.now() # Get the current datetime  
>>> print(now)  
2015-05-18 16:28:07.198690  
>>> print(type(now))  
<class 'datetime.datetime'>
```



Converting datetime to timestamp

- ◆ We refer to the January 1, 1970 00:00:00 utc+00:00 time zone as epoch time, which is recorded as 0 (a negative timestamp for before 1970), and the current time is the number of seconds relative to epoch time, called timestamp. You can take that timestamp = 0 = 1970-1-1 00:00:00 utc+0:00. The corresponding Beijing time is: timestamp = 0 = 1970-1-1 08:00:00 utc+8:00.

```
>>> from datetime import datetime  
>>> dt = datetime(2015,4,19,12,20) #Create using specified datetime  
>>> dt.timestamp()   #Convert datetime into timestamp  
1429417200.0
```



Converting timestamp to datetime

```
>>> from datetime import datetime  
>>> t = 1429417200.0  
>>> print(datetime.fromtimestamp(t))  
2015-04-19 12:20:00
```

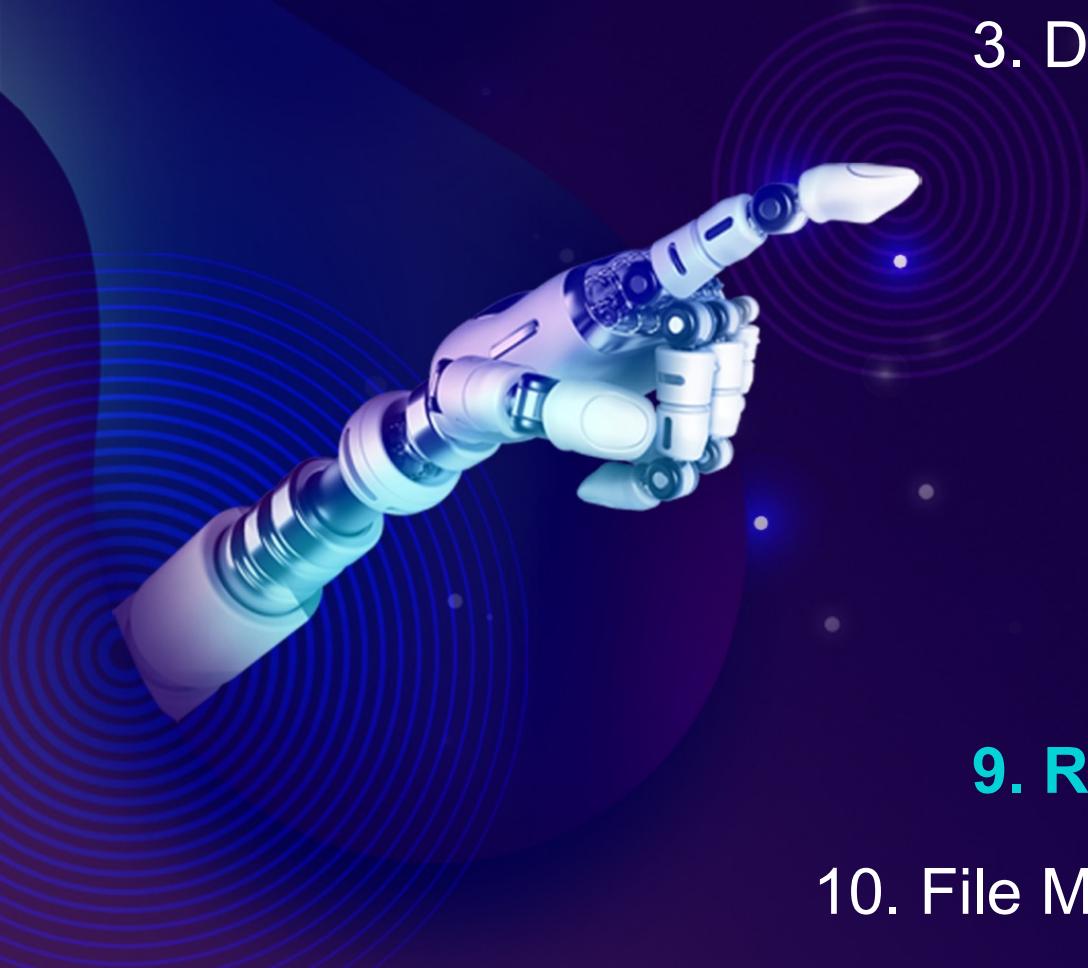
```
>>> from datetime import datetime  
>>> t = 1429417200.0  
>>> print(datetime.fromtimestamp(t)) # local time  
2015-04-19 12:20:00  
>>> print(datetime.utcfromtimestamp(t)) # UTC time  
2015-04-19 04:20:00.
```



Converting Local Time to UTC Time

```
>>> from datetime import datetime, timedelta, timezone  
>>> tz_utc_8 = timezone(timedelta(hours=8)) #create time UTC+8:00  
>>> now = datetime.now()  
>>> now  
datetime.datetime(2019, 3, 11, 9, 53, 3, 787311)  
  
>>> dt = now.replace(tzinfo=tz_utc_8)    #force time zone as UTC+8:00  
>>> dt  
datetime.datetime(2019, 3, 11, 9, 53, 3, 787311,  
tzinfo=datetime.timezone(datetime.timedelta(0, 28800)))
```

Contents

- 
1. Introduction to Python
 2. Lists and Tuples
 3. Dictionaries
 4. Strings
 5. Conditional and Looping Statements
 6. Functions
 7. Object-Oriented Programming
 8. Date and Time
 - 9. Regular Expressions**
 10. File Manipulation

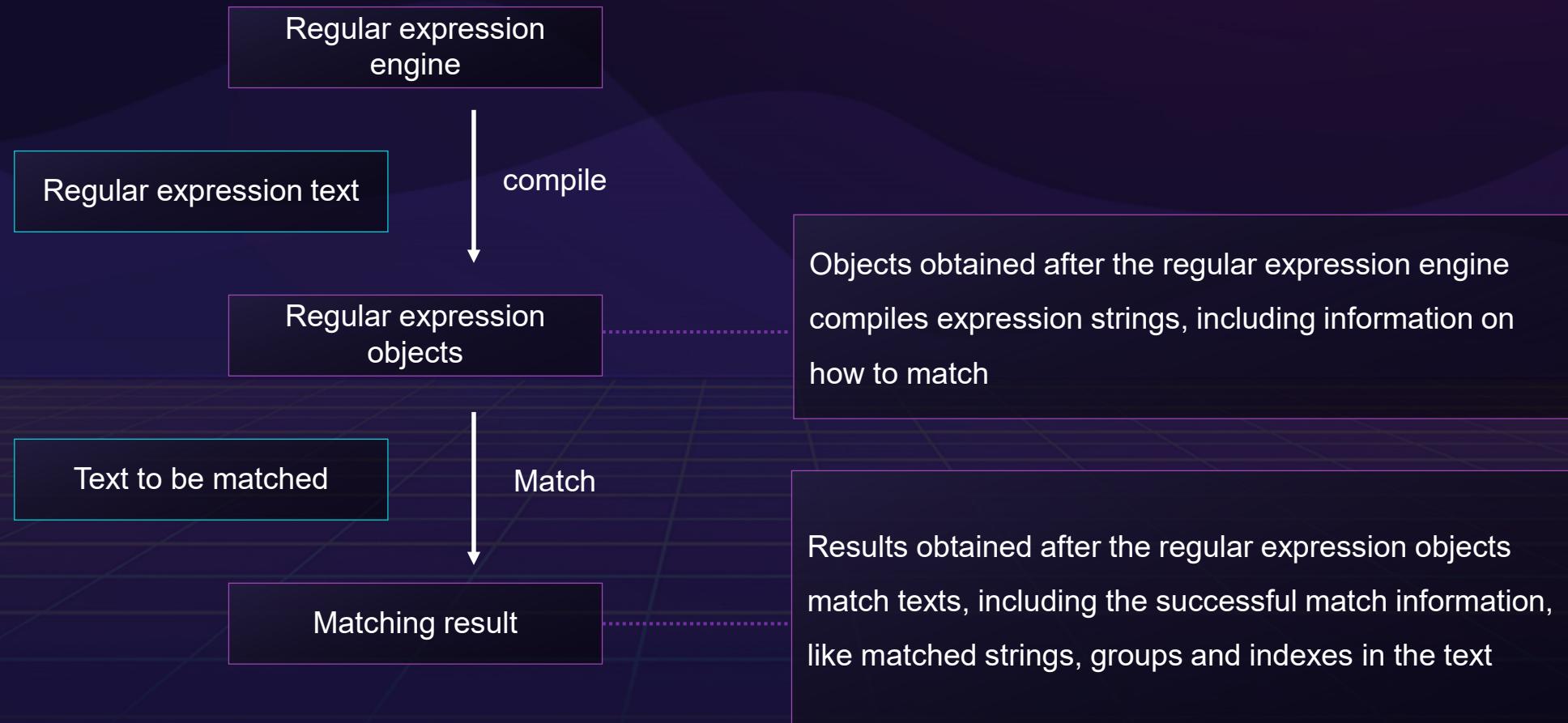


Regular Expressions (1)



- ◆ The re module enables the Python language to have full regular expression functionality.
- ◆ The re module also provides functions that are identical to those of the methods.
- ◆ match attempts to match a pattern from the starting position of the string.

Matching Process for Regular Expressions





re Modules

```
import re

# Compile a regular expression into a pattern object
pattern = re.compile(r'hello')

# Match text with pattern and return none if no match
match = pattern.match('hello world!')

if match:
    # Get group information using match
    print(match.group())

# Output
hello
```



re Module Functions and re Object Methods

Function/Metho	Description	Example	res.group()/res
compile(pattern,flag=0)	Compiles a regular expression pattern using any optional flag and returns regular expression objects.	res = re.compile("."*) print res.search("abcd").group()	abcd
match(pattern,string,flag=0)	Matches from the start of string.	res = re.match(".*","abcdxxxx")	abcd
search(pattern,string,flag=0)	Matches from any position of string.	res = re.search(".*","xxxabcdxx")	abcd
findall(pattern,string,flag=0)	Finds all regular expression patterns in strings and returns a list.	res = re.findall("a","abdadafaf")	[‘a’,’a’,’a’,’a’]
finditer(pattern,string,flag=0)	Finds all regular expression patterns in strings and returns an iterator.	res = re.finditer("a","abdadafaf") print res.next().group()	a
split(pattern,string,max=0)	Splits a string into a list by regular expression pattern.	re.split(",","li,yang,zhao")	[‘li’,’yang’,’zhao’]
sub(pattern,repl,string,count=0)	Counts the positions with repl regular expressions in strings.	res = re.sub(",","-","l,y,z")	l-y-z



Common Object Matching Methods

Function/Method	Description	Example	Result
group(num=0)	Returns the entire match object or a numbered group.	Print(re.match(".*","abcdxxxx").group())	abcdxxxx
groups(default=None)	Returns a tuple with all groups.	Print(re.search("(\\w\\w\\w)-(\\d\\d\\d)","abc-123").groups())	('abc', '123')
groupdict(default=None)	Returns a dictionary with all matched named groups (names are key values).	res = re.search("(?P<lamb>\\w\\w\\w)-(?P<num>\\d\\d\\d)","abc-123") Print(str(res.groupdict()))	{"lamb":'abc', 'num': '123'}
re.I,re.IGNORECASE	Ignore the upper and lower case.	res =re.search("abc","aBcxx",re.I) Print(res.group())	aBc
re.L,re.LOCAL	Performs matching by using \w, \W,\b,\B,\s,\S based on the local language environment.	res = re.search("\\w\\w\\w","aBcxx",re.L) Print(res.group())	aBc
re.M,re.MULTILINE	Respectively matches start and end of target strings with ^ and \$, but not exact start and end of strings.	res = re.search("^aB","aBcxx",re.M) Print(res.group())	aB

Compile

- ◆ This method is the factory method of the pattern class, which is used to compile a regular expression in a string as a patterns object. The second argument flag is a matching pattern, which can be used by bitwise or operator ' | ', meaning that it takes effect at the same time, such as re.I | re.M. Alternatively, you can specify patterns in the regex string, such as e.compile('pattern', re.I | re.M), which is equivalent to re.compile('(?im)pattern').
- ◆ Optional values:



re.I



M(MULTILINE)



S(DOTALL)



L(LOCALE)



U(UNICODE)



X(VERBOSE)

Pattern

- ◆ A pattern object is a compiled regular expression that can be matched to a search by a series of methods provided by pattern.
- ◆ Pattern cannot be directly instantiated and must be constructed using `re.compile()`.
- ◆ Pattern provides several readable properties for getting information about an expression:
 - `pattern`
 - `Groups`
 - `flags`
 - `groupindex`



Match(1)

- ◆ The match object is a matching result that contains a lot of information about the match, and you can use the readable properties or methods provided by match to get that information.



string



re



pos



endpos



lastindex



lastgroup



Match(2)

group
([group1, ...])

groups
([default])

roupdict
([default])

start
([grouap])

end
([group])

span
([group])

expand
(template)

Special Symbols and Characters - Symbols (1)

Symbol	Description	Matched Expression	res.group()
literal	Matches literal values of text strings.	res=re.search("foo", "xxxfooxxx")	foo
re1 re 2	Matches regular expressions re1 or re2.	res=re.search("foo bar", "xxxfooxxx")	foo
.	Matches any character (except \n).	res=re.search("b.b", "xxxbobxxx")	bob
^	Matches string start.	res=re.search("^b.b", "bobx xx")	bob
\$	Matches string end.	res=re.search("b.b\$","xx xbob")	bob
*	Matches regular expressions that appear none or many times (from string start).	res= re.search("bob*","bobbo") res1= re.search(".*","bobboddd")	Bobb bobboddd
+	Matches regular expressions that appear once or many times.	res= re.search("bob+","xxxxbobbob")	babbbb

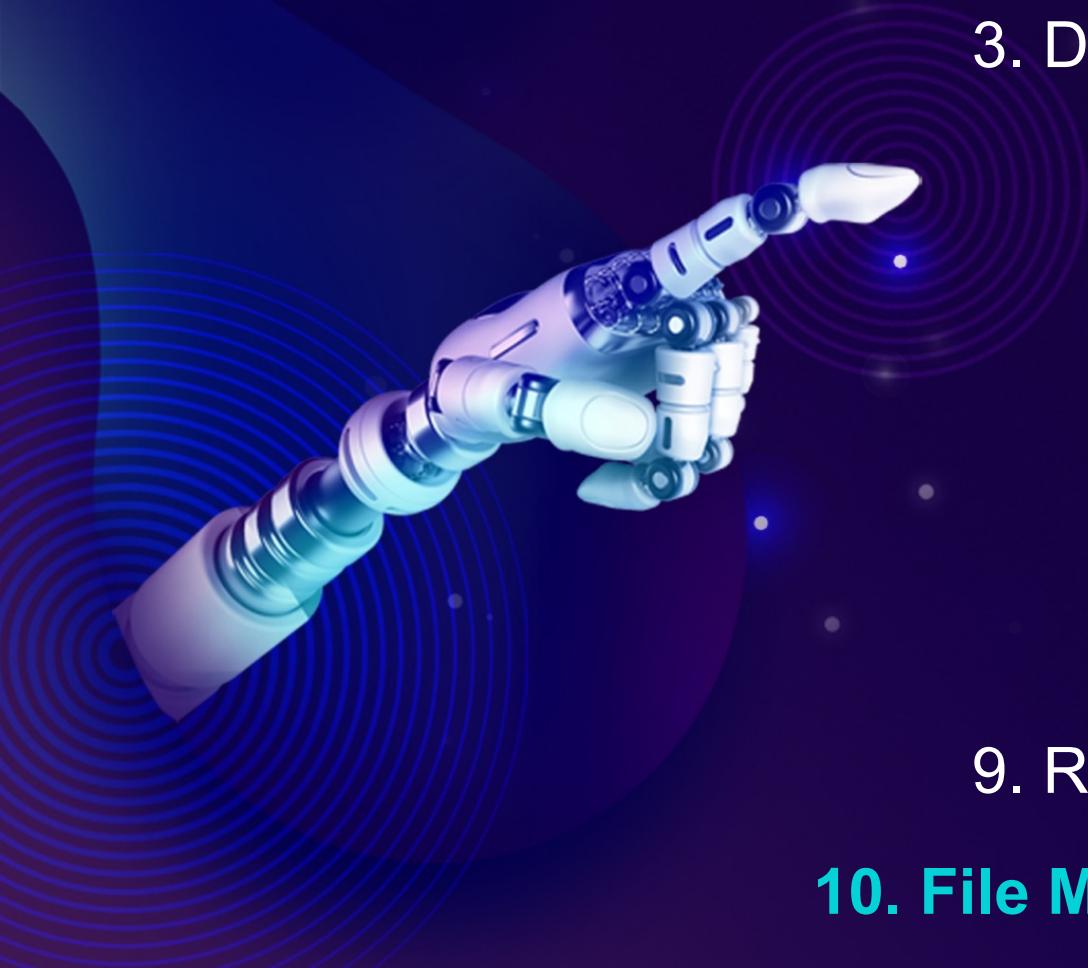
Special Symbols and Characters - Symbols (2)

Symbol	Description	Matched Expression	res.group()
?	Matches regular expressions that do not appear or appear once.	res=re.search("bob?", "bobbod")	bob
{N}	Matches regular expressions that appear N times.	res=re.search("bob{2}", "bobbdbod")	bobb
{M,N}	Matches regular expressions that appear M to N times.	res=re.search("bob{2,3}", "bobbdbod")	bobbb
[...]	Matches any character from a character set.	res= re.search("[b,o,b]", "xxbobxx")	b
[..X-Y..]	Matches any character in the x to y range.	res= re.search("[a-z]", "xxbobxx")	x
[^...]	Does not match any character in a string, including those in a range.	res= re.search("[^a-z]", "xx214bobxx") res1=re.search("[^2,x,1]", "xx214bobxx")	2 4
(* + {})?	Matches non-greedy versions of symbols that appear frequently or repeatedly.	res= re.search(".[+ ?]?[1-9]", "ds4b")	s4

Special Symbols and Characters - Characters

Character	Description	Matched Expression	res.group()
\d	Any decimal number (\D does not match any decimal number).	res=re.search("xx\dxx","oxx4xxo")	xx4xx
\w	Matches any letter or number (\W means no match).	res=re.search("xx\w\wxx","oxxa4xxo")	xxa4xx
\s	Matches any space character (\S means no match).	res=re.search("xx\sxx","oxx xxo")	xx xx
\b	Matches any letter boundary (\B means reverse).	res=re.search(r"\bthe","xxx the xxx")	the
\N	Matches saved sub-group.		
\c	Matches any special character c one by one.	res=re.search("*","x*x")	*
\A(\Z)	Matches string start (or end).	res=re.search("\ADear","Dear Mr.Li")	Dear

Contents

- 
1. Introduction to Python
 2. Lists and Tuples
 3. Dictionaries
 4. Strings
 5. Conditional and Looping Statements
 6. Functions
 7. Object-Oriented Programming
 8. Date and Time
 9. Regular Expressions
 - 10. File Manipulation**

File Manipulation (1)

◆ Opening a file

- `f.open('file name', 'access mode')`
- Common access modes:

Access Mode	Description
r	Opens a file only for reading.
w	Opens a file only for writing.
a	Opens a file only for addition.
rb	Opens a file using the binary format only for reading.
wb	Opens a file using the binary format only for writing.
ab	Opens a file using the binary format only for addition.
r+	Opens a file for reading.
w+	Opens a file for writing.
a+	Opens a file for addition.
rb+	Opens a file using the binary format for reading.
wb+	Opens a file using the binary format for writing.
ab+	Opens a file using the binary format for addition.

File Manipulation (2)

◆ Writing data:

```
f = open("name.txt", "w")
f.write("libai")
f.close()
```

◆ Reading data:

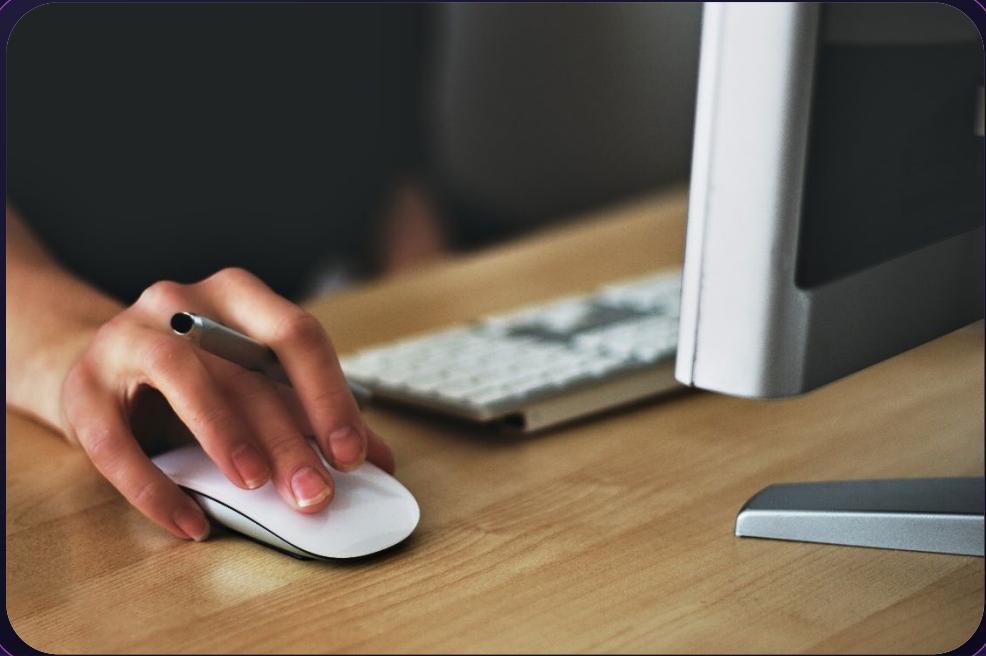
```
f = open("name.txt", "r")
lines = f.readlines()
for line in lines:
    print(line)
```

◆ Closing a file:

```
f.close()
```



Other File Manipulation Functions



- `f.write("a")` `f.write(str)`
- `f.writeline()`
- `f.readlines()`
- `f.read()`
- `f.read(size)`
- `f.readline()`
- `f.readlines()`
- `f.seek(off, where)`
- `f.flush()`



Renaming Files in a Batch

- ◆ Get the target folder:
 - `dirName = input("enter specified folder:")`.
- ◆ Get names of files in the target folder:
 - `fileNames = dirName.listdir(dirName)`.
 - `os.chdir(dirName)`.
- ◆ Rename
 - `for name in fileNames`
 - `os.rename(name,"zhangsan" + name)`





Quiz

1. Python is an object-oriented programming language. Which of the following are Python objects? ()

- A** Function
- B** module
- C** Number
- D** string

2. Which of the following are not Python file object manipulations? ()

- A** Open
- B** delete
- C** Read
- D** write

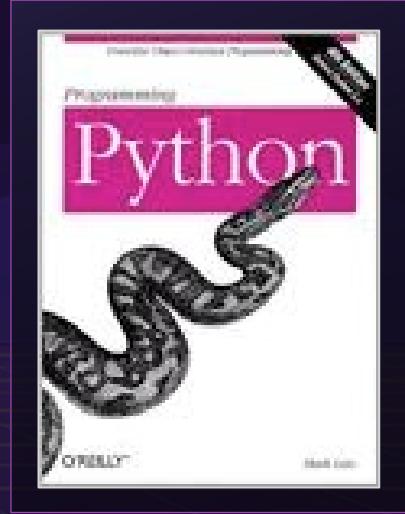
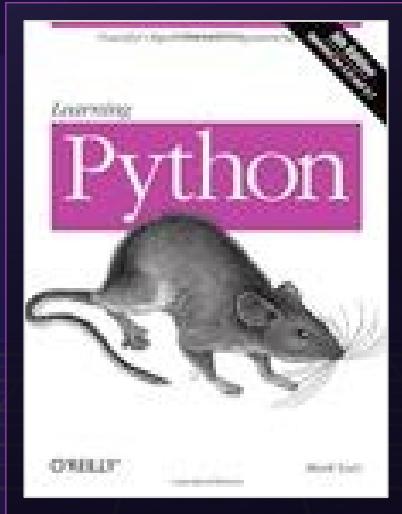


Summary of the Chapter

This course focuses on the Python language and its basic syntax, such as the Python compilation environment and installation of digital expressions, variables, statements, strings, access to user input, functions, modules and other common operations.

More Information

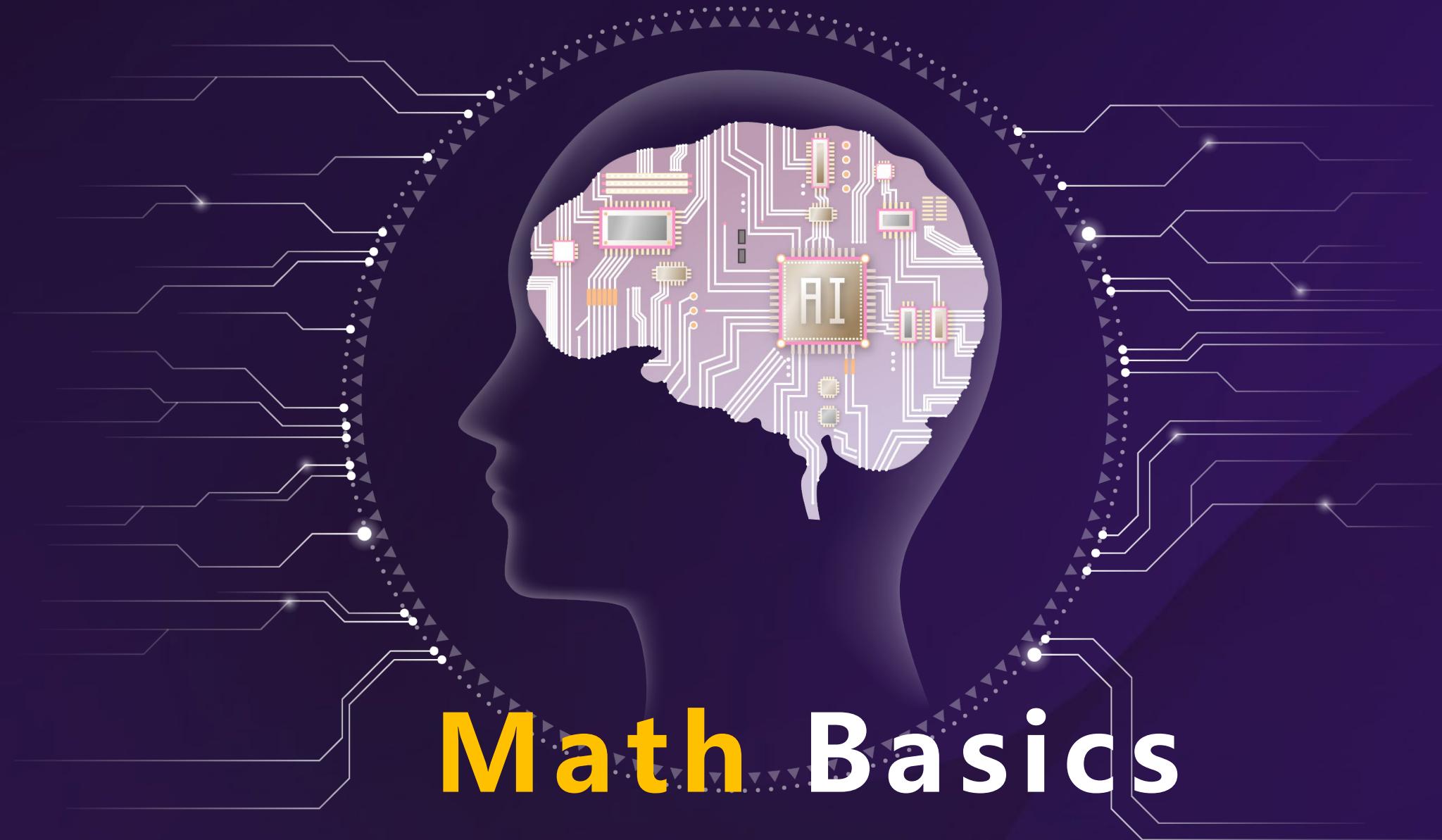
- ◆ Official site:
 - www.python.org
- ◆ References
 - 《Learning Python》
 - 《Python Standard Library》
 - 《Programming Python》



Thanks

www.huawei.com





Math Basics



Objectives

- ◆ After completing this course, you will be able to:
 - Master the basic knowledge and application of Linear Algebra.
 - Master the basic knowledge and application of Probability Theory and Information Theory.
 - Master numerical calculation functions, and the classification and solution of optimization problems

Contents

◆ Linear Algebra

- Concept and Calculation of Matrices
- Special Matrices
- Eigendecomposition

◆ Probability Theory and Information Theory

◆ Numeric Calculation

Case (1)

- ◆ To avoid obesity and improve employees' health, the Big Data Department organized a monthly running activity at the beginning of 2018. The rules were as follows: The department set the monthly target for participants at the beginning of the month. The participants who fulfilled the targets would be rewarded while those who failed would be punished. The calculation rule of the reward or penalty amount was as follows:

$$w_i = (s_i - d_i)x_i = h_i x_i$$

- ◆ In monthly target and total reward/penalty amount the preceding equation, w_i is the total reward/penalty amount in the month i , s_i is the total mileage, d_i is the monthly target, h_i is the difference between the actual distance and monthly target, and x_i is the reward/penalty amount of each kilometer every month. This activity received good feedback and was later adopted by the Cloud Department. The following tables listed the difference between the actual distance and of some participants in the first quarter:

Table 1 Big Data Department

Month Name	h_1	h_2	h_3	w
A	10	8	12	20
B	4	4	2	8
C	2	-4	-2	-5

Table 2 Cloud Department

Month Name	h_1	h_2	h_3	w
A	2	4	5	10
B	4	2	2	6
C	-2	2	2	3

Case (2)

- ◆ In the preceding case, what is the reward/penalty amount set by the Big Data Department for each kilometer in each month? The equations are as follows using the given data:

$$\begin{cases} 10x_1 + 8x_2 + 12x_3 = 20 \\ 4x_1 + 4x_2 + 2x_3 = 8 \\ 2x_1 - 4x_2 - 2x_3 = -5 \end{cases} \quad (1.1)$$

In this way, the solutions of the equations are the answer to the question.

Vector and Matrix

◆ Vector:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \mathbf{M} \\ x_n \end{pmatrix}$$

◆ matrix

$$A = \begin{pmatrix} a_{11} & a_{12} & L & a_{1n} \\ a_{21} & a_{22} & L & a_{2n} \\ M & M & M & M \\ a_{m1} & a_{m2} & L & a_{mn} \end{pmatrix}$$

$$A = A_{m \times n} = (a_{ij})_{m \times n} = (a_{ij})$$

Determinant

◆ $\det(A)$:

$$\det(A) = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ M & M & \dots & M \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{vmatrix}$$

◆ Significance:

- The determinant is equal to the product of all the eigenvalues of the matrix.
- The absolute value of the determinant can be thought of as a measure of how much multiplication by the matrix expands or contracts space. If the determinant is 0, then space is contracted completely along at least one dimension, causing it to lose all its volume. If the determinant is 1, then the transformation preserves volume.

Matrix Operation

- ◆ **Matrix addition:** Suppose that $A = (a_{ij})_{s \times n}$ and $B = (b_{ij})_{s \times n}$ are $s \times n$ matrices, and the sum of the two matrices is $C = A + B = (a_{ij} + b_{ij})_{s \times n}$.

Note: The two matrices can be added only when the matrices have the same row quantity and column quantity.

- ◆ **Scalar and matrix multiplication:** Suppose $A = (a_{ij})_{s \times n}$ and $k \in K$. The product of k and matrix A is $kA = (ka_{ij})_{s \times n}$. The addition of a scalar and matrix follows the same rule.

- ◆ **Matrix multiplication:** Suppose $A = (a_{ij})_{s \times n}$ and $B = (b_{ij})_{n \times p}$,

$$C = AB = (c_{ij})_{s \times p},$$

where $c_{i,j} = \sum_k A_{i,k}B_{k,j}$

Note: In order for AB to be defined, A must have the same number of columns as B has rows.



Matrix Transposition

- ◆ **Transposed matrix:** The transpose of a matrix is an operator which flips a matrix over its diagonal, that is, it switches the row and column indices of the matrix by producing another matrix denoted as A^T (also written as A').

Example:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \Rightarrow A^T = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \end{bmatrix}.$$

- ◆ **Nature of a transposed matrix:**

- $(A^T)^T = A$
- $(\lambda A)^T = \lambda A^T$
- $(A + B)^T = A^T + B^T$
- $(AB)^T = B^T A^T$

Trace Operator

◆ Trace operator :

$$\text{Tr}(A) = \sum_i A_{i,i}.$$

◆ Nature of a trace operator:

- $\text{Tr}(A) = \text{Tr}(A^T)$
- $\text{Tr}(A) = A$
- $\text{Tr}(ABC) = \text{Tr}(CAB) = \text{Tr}(BCA)$



Case Calculation

In the preceding case, the calculation is as follows:

- ◆ The running result of the big data department and cloud department in the first quarter can be calculated as follows:

$$A = \begin{bmatrix} 10 & 8 & 12 \\ 4 & 4 & 2 \\ 2 & -4 & -2 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 4 & 5 \\ 4 & 2 & 2 \\ -2 & 2 & 2 \end{bmatrix}.$$

- ◆ Perform the following operation on the matrices:

$$C_1 = A + B = \begin{bmatrix} 12 & 12 & 17 \\ 8 & 6 & 4 \\ 0 & -2 & 0 \end{bmatrix}, \quad C_2 = 2A = \begin{bmatrix} 20 & 16 & 24 \\ 8 & 8 & 4 \\ 4 & -8 & -4 \end{bmatrix}, \quad C_3 = AB = \begin{bmatrix} 28 & 80 & 90 \\ 20 & 28 & 32 \\ -8 & -4 & -2 \end{bmatrix}.$$

- ◆ According to the matrix multiplication rule, the equations (1.1) can be represented by a matrix as follows:

$$\begin{cases} 10x_1 + 8x_2 + 12x_3 = 20 \\ 4x_1 + 4x_2 + 2x_3 = 8 \\ 2x_1 - 4x_2 - 2x_3 = -5 \end{cases} \Rightarrow \begin{bmatrix} 10 & 8 & 12 \\ 4 & 4 & 2 \\ 2 & -4 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 20 \\ 8 \\ -5 \end{bmatrix} \Rightarrow Ax = C.$$

Contents

◆ Linear Algebra

- Concept and Calculation of Matrices
- Special Matrices
- Eigendecomposition

◆ Probability Theory and Information Theory

◆ Numeric Calculation



Identity and Inverse Matrices

- ◆ **Identity matrix:** All the entries along the main diagonal are 1, while all the other entries are 0. An identity matrix does not change any vector when we multiply that vector by that matrix.

$$I_n = \begin{bmatrix} 1 & 0 & L & 0 \\ 0 & 1 & L & 0 \\ M & M & O & M \\ 0 & 0 & L & 1 \end{bmatrix}.$$

- ◆ **Matrix inverse:** The **matrix inverse** of A is denoted as A^{-1} , and it is defined as the matrix such that $A^{-1}A = I_n$.

Diagonal Matrix

- ◆ **Diagonal matrix:** consists mostly of zeros and have non-zero entries only along the main diagonal. It is often written as $\text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$

$$D = \begin{bmatrix} \lambda_1 & 0 & L & 0 \\ 0 & \lambda_2 & L & 0 \\ M & M & O & M \\ 0 & 0 & M & \lambda_n \end{bmatrix}.$$

- ◆ **Nature of a diagonal matrix:**

- The sum, difference, product, and square power of the elements on the diagonal matrix are the sum, difference, product, and square power of the elements along the main diagonal.
- The inverse matrix is as follows:

$$D^{-1} = \begin{bmatrix} \lambda_1^{-1} & 0 & L & 0 \\ 0 & \lambda_2^{-1} & L & 0 \\ M & M & O & M \\ 0 & 0 & M & \lambda_n^{-1} \end{bmatrix}.$$

Symmetric Matrix

- ◆ **Symmetric matrix:** If $A^T = A$ ($a_{ij} = a_{ji}$) in square matrix $A = (a_{ij})_{n \times n}$, A is a symmetric matrix.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{12} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \dots & a_{nn} \end{bmatrix}.$$

- ◆ **Orthogonal matrix:** If $AA^T = A^TA = I_n$ in the square matrix $A = (a_{ij})_{n \times n}$, A is an orthogonal matrix. That is, $A^{-1} = A^T$.

Contents

◆ Linear Algebra

- Concept and Calculation of Matrices
- Special Matrices
- Eigendecomposition

◆ Probability Theory and Information Theory

◆ Numeric Calculation

Eigen decomposition (1)

- ◆ One of the most widely used kinds of **matrix decomposition** is called eigen decomposition, in which we decompose a matrix into a set of **eigenvectors** and **eigenvalues**. We can decompose matrices in ways that show us information about their functional properties that is not obvious from the representation of the matrix as an array of elements.
- ◆ Suppose that A is a n -level matrix in the digital domain K . If there is a non-zero column vector α in K^n that meets the following:

$$A\alpha = \lambda\alpha, \text{ and } \lambda \in K,$$

λ is called an eigenvalue of A , and α is a eigenvector of A and belongs to the eigenvalue λ .

Example:

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad \alpha = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad A\alpha = \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 2\alpha.$$

Therefore, 2 is an eigenvalue of A , and α is an eigenvector of A and belongs to eigenvalue 2.



Eigen decomposition (2)

- ◆ Obtaining the eigenvalues and eigenvectors of matrix A:

$$\begin{aligned}A\alpha &= \lambda\alpha \\ \Leftrightarrow A\alpha - \lambda\alpha &= 0 \\ \Leftrightarrow (A - \lambda I)\alpha &= 0 \\ \stackrel{\alpha \neq 0}{\Leftrightarrow} |A - \lambda I| &= 0 \\ \Leftrightarrow \begin{vmatrix} a_{11} - \lambda & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} - \lambda & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} - \lambda \end{vmatrix} &= 0.\end{aligned}$$

In the preceding information, $|A - \lambda I| = 0$ is a feature equation of matrix A, λ is a solution (characteristic root) of the feature equation. To obtain the eigenvector α , substitute the characteristic root λ into $A\alpha = \lambda\alpha$.

Example:

Example: Find the eigenvalues and eigenvectors of the matrix $A = \begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix}$.

Solution: The characteristic polynomial of A is $\begin{vmatrix} 3-\lambda & -1 \\ -1 & 3-\lambda \end{vmatrix} = (3-\lambda)^2 - 1 = (4-\lambda)(2-\lambda)$. Therefore, the eigenvalues of A are $\lambda_1 = 2$ and $\lambda_2 = 4$.

Taking $\lambda_1 = 2$, the corresponding eigenvector satisfies $\begin{bmatrix} 3-2 & -1 \\ -1 & 3-2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, we find that $x_1 = x_2$.

Therefore, the corresponding eigenvector is $p_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. When $\lambda_1 = 2$, the eigenvector is $kp_1(k \neq 0)$.

Taking $\lambda_2 = 4$, we find that $x_1 = -x_2$. The eigenvector is $p_2 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$. When $\lambda_2 = 4$, the eigenvector is $kp_2(k \neq 0)$.

Matrix accuracy

- ◆ Suppose that a matrix A has n linearly independent eigenvectors $\{\alpha_1, \dots, \alpha_n\}$ with corresponding eigenvalues $\{\lambda_1, \dots, \lambda_n\}$. The eigendecomposition of A is then given by

$$A = P \text{diag}(\lambda) P^{-1}$$

where $P = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$, and $\lambda = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$.

- ◆ Matrix accuracy:

- A matrix whose eigenvalues are all positive is called positive definite.
- If all eigenvalues are negative, the matrix is negative definite.
- A matrix whose eigenvalues are all positive or zero valued is called positive semidefinite.
- If all eigenvalues are negative or zero valued, it is negative semidefinite.

Singular Value Decomposition

◆ **Singular Value Decomposition:** The matrix is decomposed into singular vectors and singular values. The matrix $A = (a_{ij})_{m \times n}$ can be decomposed into a product of three matrices:

$$A = UDV^T,$$

Among $U = (b_{ij})_{m \times m}$, $D = (c_{ij})_{m \times n}$, and $V^T = (d_{ij})_{n \times n}$, the matrices U and V are both defined to be orthogonal matrices. The columns of U are known as the left-singular vectors. The columns of V are known as the right-singular vectors. The matrix D is defined to be a diagonal matrix. Note that D is not necessarily square. Elements on the diagonal line of D are referred to as a singular value of the matrix.



Moore-Penrose Pseudoinverse

- ◆ The Moore-Penrose pseudoinverse enables us to make some headway in finding the solution of $Ax = y$ ($A = (a_{ij})_{m \times n}$, $m \neq n$). The pseudoinverse of A is defined as a matrix:

$$A^+ = \lim_{\alpha \downarrow 0} (A^T A + \alpha I)^{-1} A^T$$

Practical algorithms for calculating the pseudoinverse are based on the formula:

$$A^+ = V D^+ U^T$$

where U , D and V are the singular value decomposition of A , and the pseudoinverse D^+ of a diagonal matrix D is obtained by taking the reciprocal of its non-zero elements then taking the transpose of the resulting matrix.



Principal Component Analysis (1)

- ◆ **Principal Component Analysis (PCA):** a statistical method. Through orthogonal transform, a group of variables that may have correlation relationships are converted into a set of linear unrelated variables, and the converted variables are called main components.
- ◆ **Basic :** Assume that there are n objects, and each object is composed of $\{x_1, \dots, x_p\}$. The following table lists the factor data corresponding to each object.

Factor Object	x_1	x_2	...	x_j	...	x_p
1	x_{11}	x_{12}	...	x_{1j}	...	x_{1p}
2	x_{21}	x_{22}	...	x_{2j}	...	x_{2p}
...
i	x_{i1}	x_{i2}	...	x_{ij}	...	x_{ip}
...
n	x_{n1}	x_{n2}	...	x_{nj}	...	x_{np}

Principal Component Analysis (2)

- ◆ The original variables are x_1, \dots, x_p . After the dimension-reduction processing, set their comprehensive indexes. That is, the new variables are z_1, \dots, z_m ($m \leq p$). z_1, \dots, z_m are called the first, the second, ..., the m th main component of x_1, \dots, x_p . We have the following expression:

$$\begin{cases} z_1 = l_{11}x_1 + l_{12}x_2 + \dots + l_{1p}x_p \\ z_2 = l_{21}x_1 + l_{22}x_2 + \dots + l_{2p}x_p \\ \vdots \\ z_m = l_{m1}x_1 + l_{m2}x_2 + \dots + l_{mp}x_p \end{cases}$$

- ◆ To obtain m principal components, the steps are as follows:

- The coefficient l_{ij} meets the following rules: z_i is not related to z_j ($i \neq j; i, j = 1, 2, \dots, m$). z_1 has the largest variance among all linear combinations of x_1, \dots, x_p . z_2 has the largest variance among all linear combinations of x_1, \dots, x_p that is not related to z_1 . z_m has the largest variance among all linear combinations of x_1, \dots, x_p that is not related to z_1, z_2, \dots, z_{m-1} .
- According to the above rules, l_{ij} is a eigenvector of m large eigenvalues of the coefficient matrix corresponding to x_1, \dots, x_p .
- If the cumulative contribution rate of the first i main components reaches 85% to 90%, those components are used as the new variables.



Principal Component Analysis (3)

- ◆ Correlation coefficient matrix and correlation coefficient:

$$R = \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1p} \\ r_{21} & r_{22} & \dots & r_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ r_{p1} & r_{p2} & \dots & r_{pp} \end{bmatrix}, \quad r_{ij} = \frac{\sum_{k=1}^p (x_{ki} - \bar{x}_i)(x_{kj} - \bar{x}_j)}{\sqrt{\sum_{k=1}^p (x_{ki} - \bar{x}_i)^2 \sum_{k=1}^p (x_{kj} - \bar{x}_j)^2}}.$$

- ◆ Contribution rate of the main components and cumulative contribution rate:

$$Q_i = \frac{\lambda_i}{\sum_{k=1}^p \lambda_k} (i = 1, 2, \dots, p), \quad Q = \frac{\sum_{k=1}^i \lambda_k}{\sum_{k=1}^p \lambda_k} (i = 1, 2, \dots, p).$$

Contents

◆ Linear Algebra

◆ **Probability Theory and Information Theory**

- Basic Concepts of Probability Theory
- Random Variables and Their Distribution Functions
- Numerical Characteristics of Random Variables
- Information Theory

◆ Numeric Calculation



Why Do We Use Probability?

- ◆ While probability theory allows us to make uncertain statements and to reason in the presence of uncertainty, information theory enables us to quantify the amount of uncertainty in a probability distribution.
- ◆ There are three possible sources of uncertainty:



Inherent stochasticity in the system being modeled



Incomplete observability



Incomplete modeling

Random Test

◆ The test that meets the following three characteristics is called a random test:

- It can be repeated under the same condition.
- There may be more than one result of each test, and all possible results of the test can be specified in advance.
- Before a test, we cannot determine which result will appear.

◆ Example:

- E_1 :Toss two coins and check the outcome (front or back).
- E_2 :Throw a dice and check the number of points that may appear.





Basic Concepts

- ◆ **Sample point:** each possible result of a random test, which is represented by e .
- ◆ **Sample space:** a collection of all possible results of a random test, which is represented by $S = \{e_1, e_2, \dots, e_n\}$.
- ◆ **Random variables event:** any subset of the sample space S . If a sample point of event A occurs, event A occurs. In particular, a random event containing only one sample point is called a basic event.
- ◆ **Example:**

Random test: Throw a dice and check the outcome.

- Sample space: $S=\{1, 2, 3, 4, 5, 6\}$
- Sample point: $e_i = 1, 2, 3, 4, 5, 6$
- Random event A_1 : "The outcome is 5", that is, $A_1 = \{x|x = 5\}$.



Frequency and Probability

- ◆ **Frequency:** Under the same conditions, perform tests for n times. The occurrence of event A is called the frequency of event A. The ratio $\frac{n_A}{n}$, occurrence probability of event A, is recorded as $f_n(A)$.
- ◆ **Probability:** Suppose that E is a random test and S is the sample space. Assign a real number $P(A)$ (event probability) on each event A of E. The set function $P(*)$ must meet the following conditions:
 - Non-negative: For each event A, $0 \leq P(A) \leq 1$.
 - Standard: For the inevitable event S, $P(S) = 1$.
- ◆ **Countable additivity:** $\{A_1, \dots\}$ are events incompatible with each other. That is, if $A_i A_j = \emptyset, i \neq j, i, j = 1, 2, \dots$, we have $P(A_1 \cup A_2 \cup \dots) = P(A_1) + P(A_2) + \dots$.

Random Variable

- ◆ **random variable** : indicates a single- and real-valued function that represents a random test of various results.
- ◆ **Example 1:** Random test E_4 : Toss two dice and check the sum of the results. The sample space of the test is $S = \{e\} = \{(i, j) | i, j = 1, 2, 3, 4, 5, 6\}$. i indicates the first outcome and j indicates the second outcome. X is the sum of the two outcomes, which is a random variable.

$$X = X(e) = X(i, j) = i + j, i, j = 1, 2, \dots, 6.$$

- ◆ **Example 2:** Random test E_1 : Throw two coins and check the outcome (front side H or back side T). The sample space for the test is $S = \{HH, HT, TH, TT\}$. Y , as the total occurrence of the back side T, is a random variable.

$$Y = Y(e) = \begin{cases} 0, & e = HH, \\ 1, & e = HT, TH, \\ 2, & e = TT. \end{cases}$$

Contents

- 
- ◆ Linear Algebra
 - ◆ **Probability Theory and Information Theory**
 - Basic Concepts of Probability Theory
 - Random Variables and Their Distribution Functions
 - Numerical Characteristics of Random Variables
 - Information Theory
 - ◆ Numeric Calculation

Discrete Random Variables and Distribution Law

- ◆ **Discrete random variables:** All the values of random variables may be finite or infinite. A typical random variable is the number of vehicles passing through a monitoring gate within one minute.
- ◆ **Distribution law:** If all the possible values of discrete random variable X are $x_k (k = 1, 2, \dots)$, the probability of X getting a possible value $\{X = x_k\}$ is:

$$P\{X = x_k\} = p_k, k = 1, 2, \dots$$

As defined for probability, p_k should meet the following conditions:

$$(1) p_k \geq 0, k = 1, 2, \dots$$

$$(2) \sum_{k=1}^{\infty} p_k = 1$$

The distribution law can also be expressed in a table:

X	x_1	x_2	\dots	x_n	\dots
p_k	p_1	p_2	\dots	p_n	\dots



Special Distribution - Bernoulli Distribution

- ◆ **Bernoulli distribution (0-1 distribution, two-point distribution, a-b distribution):** If random variable X can be either 0 or 1, its distribution law is:

$$P\{X = k\} = p^k(1 - p)^{1-k}, k = 0, 1 \quad (0 < p < 1)$$

That is, X obeys Bernoulli distribution with the p parameter.

- ◆ **The distribution law of Bernoulli distribution** can also be written as below:

X	0	1
p_k	$1 - p$	p

$$E(X) = p, \text{Var}(X) = p(1 - p).$$

Special Distribution - Binomial Distribution

- ◆ **n independent repetitive tests:** The experiment E is repeated n times. If the results of each experiment do not affect each other, the n experiments are said to be independent of each other.
- ◆ **The experiments that meet the following conditions are called n Bernoulli experiments:**
 - Each experiment is repeated under the same conditions.
 - There are only two possible results per experiment: A and \bar{A} and $P(A) = p$.
 - The results of each experiment are independent of each other.

If the times of event A occurring in n Bernoulli experiments are expressed by X , the probability of event A occurring for k times in n experiments is as below:

$$P(X = k) = C_n^k p^k (1 - p)^{n-k}, k = 0, 1, 2, \dots, n,$$

At this time, X obeys binomial distribution with n and p parameters. This is expressed as $X \sim B(n, p)$, where $E(X)$ equals np and $Var(x)$ equals $np(1-p)$.



Special Distribution - Poisson Distribution

- ◆ **Poisson theorem:** If $\lambda > 0$ is set as a constant, n is any positive integer, and np equals λ , the following applies to any fixed non-negative integer k :

$$\lim_{n \rightarrow \infty} C_n^k p^k (1-p)^{n-k} \approx \frac{\lambda^k e^{-\lambda}}{k!}$$

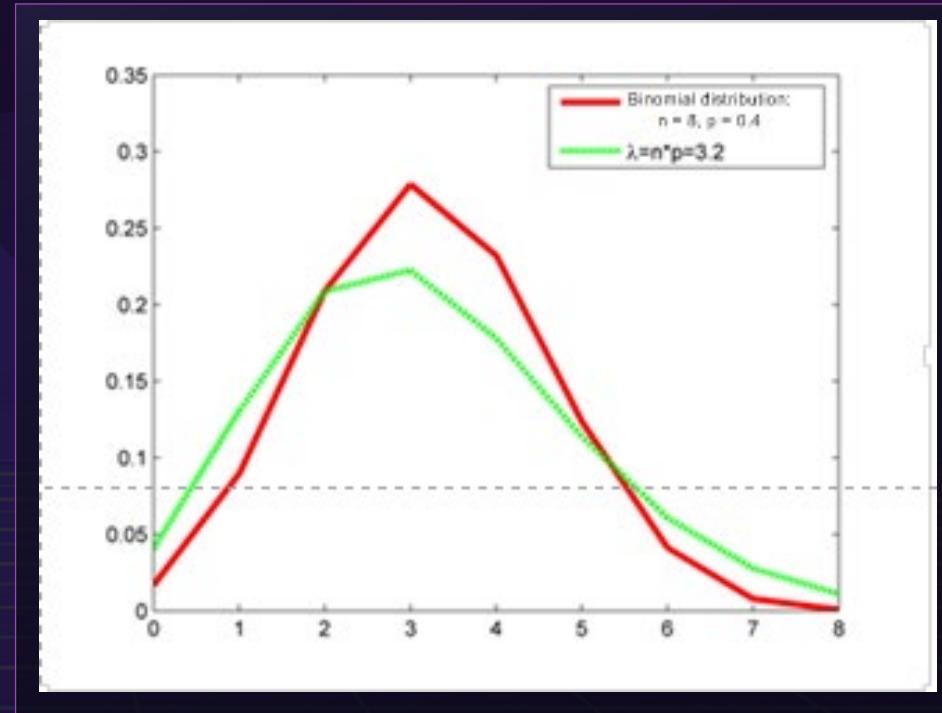
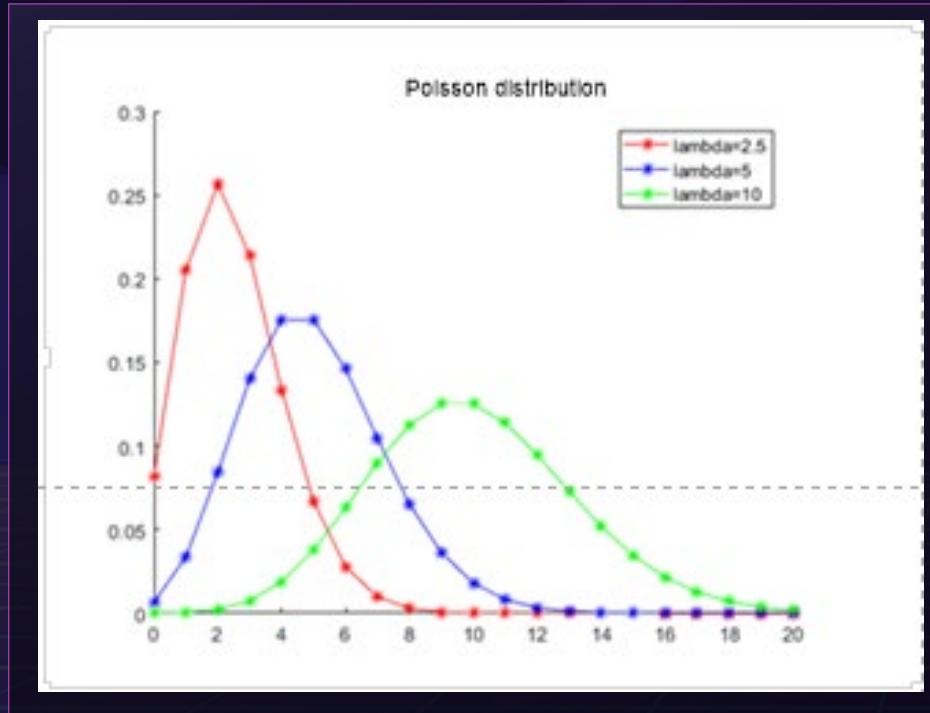
- ◆ **Poisson distribution:** If all possible values of random variables are $0, 1, 2, \dots$, the probability of taking each value is:

$$P\{X = k\} = \frac{\lambda^k e^{-\lambda}}{k!}, k = 0, 1, 2, \dots$$

Then, X obeys Poisson distribution with parameter λ . It is expressed as $X \sim P(\lambda)$, where $E(X)$ equals λ , and $D(X)$ equals λ .



Association Between Poisson Distribution and Binomial Distribution



The mathematical models of Poisson distribution and Binomial distribution are both Bernoulli-type. Poisson distribution has the appropriately equal calculation as binomial distribution when n is very large and p very small.



Distribution Function

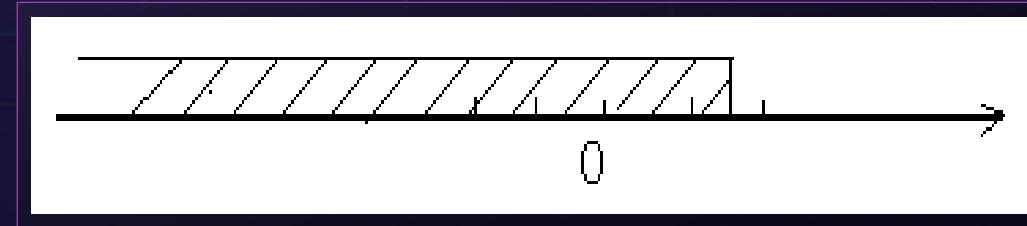
- ◆ **Distribution function:** If X is a random variable, and x is an arbitrary real number, function $F(x)$ is called the distribution function of X .

$$F(x) = P\{X \leq x\}, -\infty < x < \infty.$$

- ◆ **Distribution function $F(x)$ has the following basic properties:**

- $F(x)$ is a function of no subtraction.
- $0 \leq F(x) \leq 1$, and $F(-\infty) = \lim_{x \rightarrow -\infty} F(x) = 0$, $F(\infty) = \lim_{x \rightarrow \infty} F(x) = 1$.
- $F(x+0) = F(x)$, that is, $F(x)$ is of right continuity.

- ◆ **Significance of distribution function $F(x)$:** If X is regarded as the coordinate of a random point on the number axis, the function value of distribution function $F(x)$ at x indicates the probability that X falls in the interval $(-\infty, x]$.





Continuous Random Variables and Probability Density Function

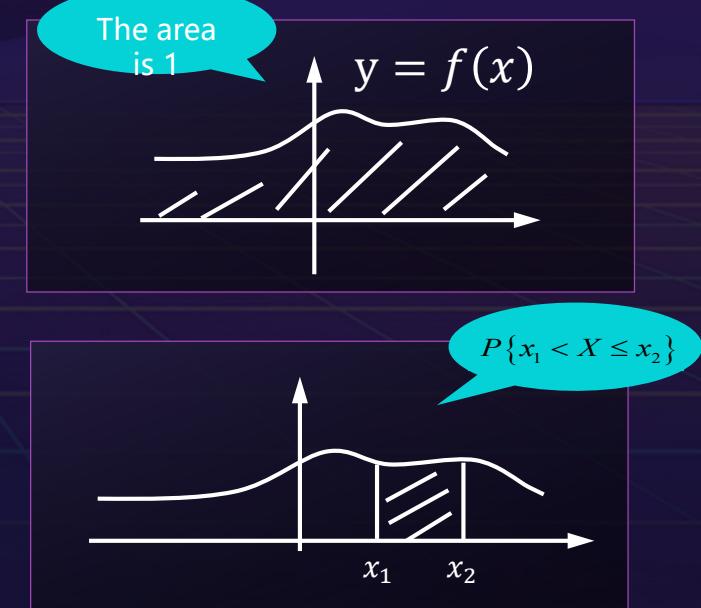
- ◆ If distribution function $F(x)$ for random variable X has a non-negative function $f(x)$, and the following applies to arbitrary real number x :

$$F(x) = \int_{-\infty}^x f(t)dt,$$

Then, X is called a continuous random variable, and function $f(x)$ is called the probability density function of X , or probability density.

- ◆ Probability density $f(x)$ has the following properties:

- $f(x) \geq 0$.
- $\int_{-\infty}^{+\infty} f(x)dx = 1$.
- For arbitrary real number $x_1, x_2 (x_1 < x_2)$, $P\{x_1 < X \leq x_2\} = F(x_2) - F(x_1) = \int_{x_1}^{x_2} f(x)dx$.
- If $f(x)$ is continuous at x , $F'(x) = f(x)$.
- The probability value of random variable X taking any real number is 0, that is, $P(X=a) = 0$.

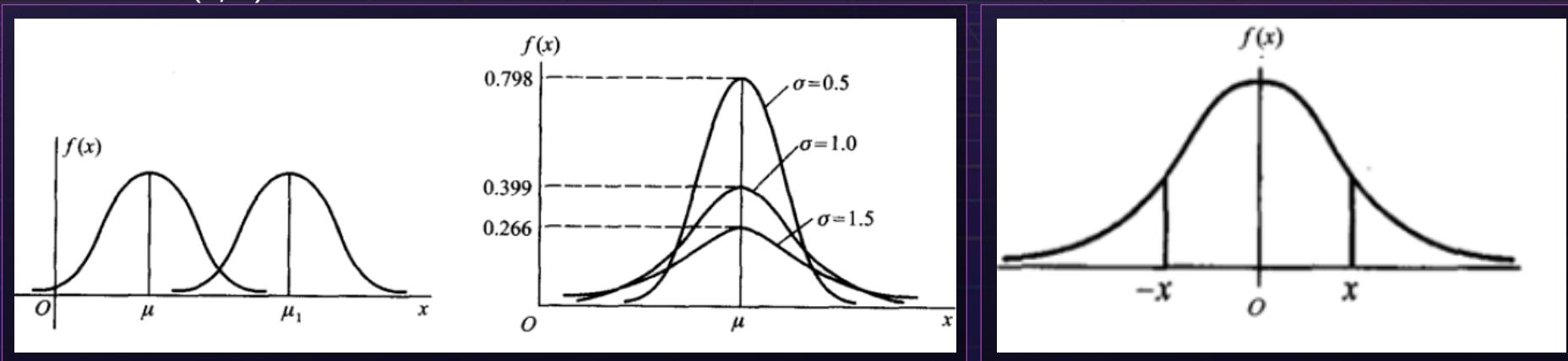


Special Distribution - Normal Distribution

- ◆ If the probability density function of continuous random variable X is

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \quad -\infty < x < \infty$$

where $\mu, \sigma (\sigma>0)$ is constant, X obeys the normal distribution or Gaussian distribution of μ, σ , which is expressed as $X \sim N(\mu, \sigma^2)$. Especially when $\mu = 0, \sigma = 1$, random variable X obeys the standard normal distribution, which is expressed as $X \sim N(0, 1)$.

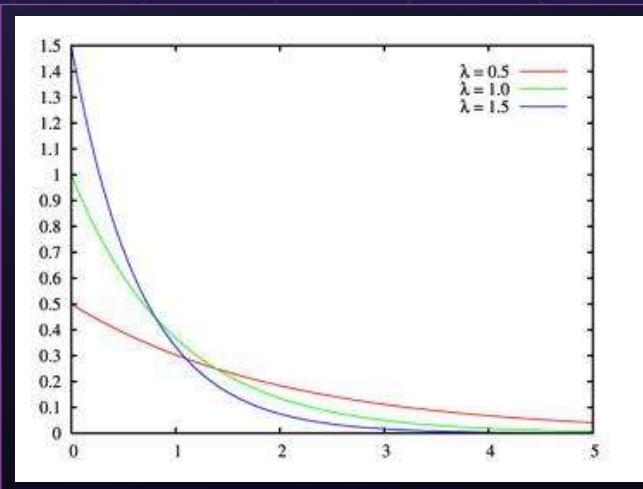


Special Distribution - Exponential Distribution

- ◆ If the probability density of continuous random variable X is

$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & x > 0 \\ 0, & \text{otherwise} \end{cases}$$

where $\lambda > 0$ is a constant, indicating the time when a random event occurs once, X obeys the exponential distribution with parameter λ . This distribution is expressed as $X \sim E(\lambda)$. $E(X) = 1/\lambda$, $\text{Var}(X) = 1/\lambda^2$.



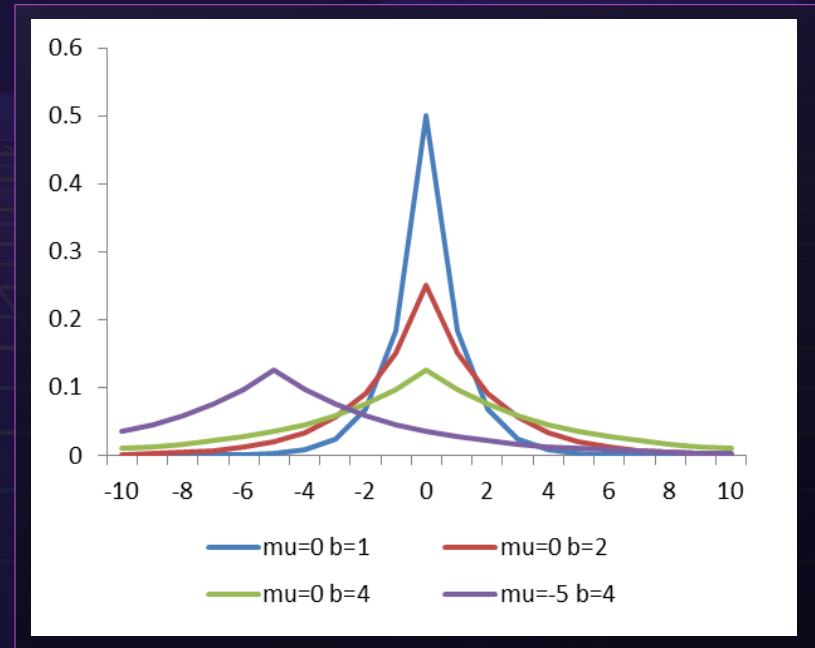


Special Distribution – Laplace Distribution

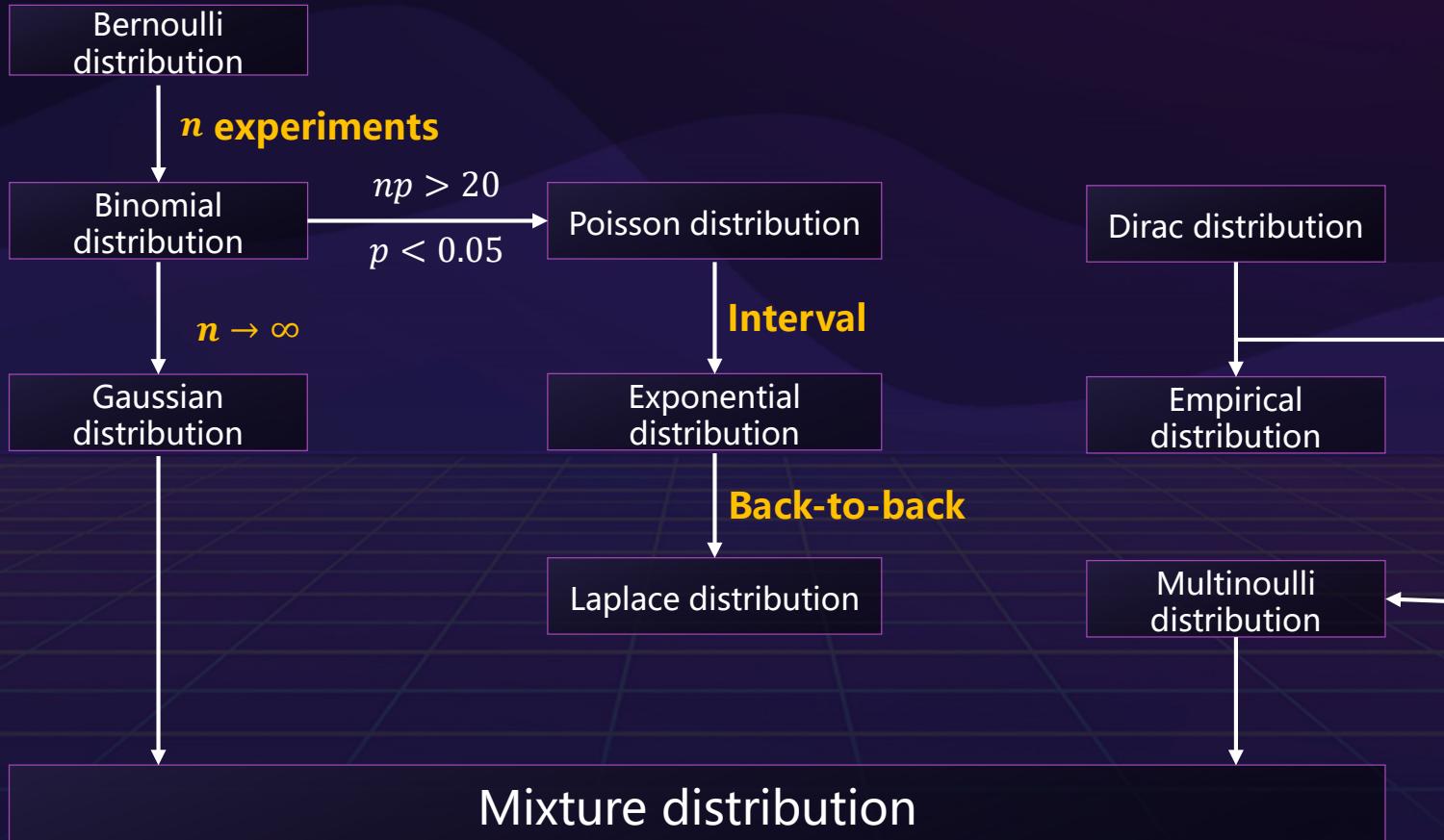
- ◆ If the probability density of continuous random variable X is

$$Laplace(x; \mu, b) = \frac{1}{2b} e^{-\frac{|x-\mu|}{b}},$$

where μ is the position parameter, and b is the scale parameter, X obeys the Laplace distribution. This distribution is expressed as $X \sim Laplace(x; \mu, b)$. $E(X) = \mu$, $V ar(X) = 2b^2$.



Summary of Probability Distribution





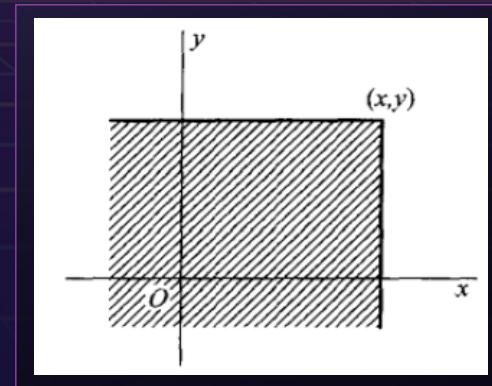
Two-Dimensional Random Variable and Joint Distribution Function

- ◆ **Two-dimensional random variable:** E is a random experiment, and its sample space is $S = \{e\}$. If $X = X(e)$ and $Y = Y(e)$ are defined as random variables on S , they make a vector (X, Y) , called two-dimensional random variable.
- ◆ **Distribution function of two-dimensional random variable:** If (X, Y) is a two-dimensional random variable, for any real numbers x, y , the binary function applies:

$$F(x, y) = P\{(X \leq x) \cap (Y \leq y)\} = P\{X \leq x, Y \leq y\}$$

It is called a distribution function for a two-dimensional random variable (X, Y) , or a joint distribution function for random variables X and Y .

- ◆ **Significance of the joint distribution function:** If (X, Y) is considered as the coordinate of a random point on the plane, distribution function $F(x, y)$ at (x, y) is the probability of random point (X, Y) falling in the infinite rectangular field at the point (x, y) vertex and at the lower left of the point.





Two-Dimensional Discrete Random Variable and Joint Distribution Law

- ◆ **Two-dimensional discrete random variable:** All possible values of discrete random variable (X,Y) can be finite or infinite pairs.
- ◆ **Joint distribution law of X and Y :**

X	x_1	x_2	...	x_i	...
y_1	p_{11}	p_{12}	...	p_{1i}	...
y_2	p_{21}	p_{22}	...	p_{2i}	...
:	:	:		:	
y_j	p_{j1}	p_{j2}	...	p_{ji}	...
:	:	:		:	



Two-Dimensional Continuous Variable and Joint Probability Density

If distribution function $F(x,y)$ of two-dimensional random variable (X,Y) has a non-negative function $f(x,y)$ that makes the following apply to arbitrary x, y

$$F(x,y) = \int_{-\infty}^y \int_{-\infty}^x f(u,v) du dv$$

(X,Y) is a continuous two-dimensional random variable and function is the joint probability density for two-dimensional random variable X .

Handwritten mathematical derivation:

$$\begin{aligned} & \int \sin x \cdot \sin 2x \cdot \sin 3x dx \\ &= \int (\sin x \cdot \sin 2x) \sin 3x dx \\ &= \frac{1}{2} \int (\sin 3x \cdot \cos 2x - \frac{1}{4} \cos 4x + \frac{1}{6} \cos 6x) dx \\ &= \frac{1}{4} \left(-\frac{1}{2} \cos 2x - \frac{1}{4} \cos 4x - \frac{1}{8} \cos 6x \right) \end{aligned}$$



Marginal Distribution

◆ **Marginal distribution function:** Two-dimensional random variable (X,Y) as a whole has distribution function $F(x,y)$. X and Y are random variables, and they also have their distribution functions, which are expressed as $F_X(x)$ and $F_Y(y)$ and are called as the marginal distribution functions of two-dimensional random variable (X,Y) about X and Y , respectively. $F_X(x) = P\{X \leq x\} = P\{X \leq x, Y \leq \infty\} = F(x, \infty)$

◆ **For discrete random variable:**

- Marginal distribution function: $F_X(x) = \sum_{x_i \leq x} \sum_{j=1}^{\infty} p_{ij}$.
- Marginal density function: $p_{i\cdot} = \sum_{j=1}^{\infty} p_{ij}, j = 1, 2, \dots$.

◆ **For continuous random variable:**

- Marginal distribution function: $F_X(x) = F(x, \infty) = \int_{-\infty}^x [\int_{-\infty}^{+\infty} f(x, y) dy] dx$.
- Marginal density function: $f_X(x) = \int_{-\infty}^{+\infty} f(x, y) dy$.



Conditional Probability and Bayes Formula

◆ conditional probability:

$$P(Y|X) = \frac{P(YX)}{P(X)}$$

◆ Bayes formula :

$$P(X|Y) = \frac{P(XY)}{P(Y)} = \frac{P(Y|X)P(X)}{P(Y)}$$

- ◆ Assuming that X is a probabilistic space $\{X_1, X_2, \dots, X_n\}$ composed of independent events, $P(Y)$ can be expanded with a full probability formula: $P(Y) = P(Y|X_1)P(X_1) + P(Y|X_2)P(X_2) + \dots + P(Y|X_n)P(X_n)$. Then, **the Bayes formula** can be expressed as:

$$P(X_i|Y) = \frac{P(Y|X_i)P(X_i)}{\sum_{i=1}^n P(Y|X_i)P(X_i)}$$

◆ The chain rule of conditional probability:

$$P(X_1, X_2, \dots, X_n) = P(X_1) \prod_{i=2}^n P(X_i|X_1, \dots, X_{i-1})$$

Independence and Conditional Independence

- ◆ Two random variables X and Y , if for all x, y , the following applies

$$P(X = x, Y = y) = P(X = x)P(Y = y)$$

Random variables X and Y are of mutual independence, which is expressed as $X \perp Y$.

- ◆ If for each value of Z for the conditional probability about X and Y , the following applies

$$P(X = x, Y = y|Z = z) = P(X = x|Z = z)P(Y = y|Z = z)$$

Random variables X are of conditional independence at given random variable Z , which is expressed as $X \perp Y|Z$.



Examples of Bayesian rules

- ◆ Wang went to hospital for a blood test, and got a positive result, indicating that he may have been attacked by the X disease. According to data on the Internet, 1% of the people who were sick of this disease were false positive, and 99% were true positive. In those who did not get sick of this disease, 1% of the people were false negative, and 99% were true negative. As a result, Wang thought, with only 1% false positive rate, and 99% true positive rate, the probability of Wang getting infected with the X disease should be 99%. However, the doctor told him that the probability of his infection was only about 0.09.

$X = 1$ (infected), $X = 0$ (not infected), $y = 1$ (tested as positive), $y = 0$ (tested as negative)

$$P(X = 1|y = 1) = \frac{P(X = 1)P(y = 1|X = 1)}{P(y = 1|X = 1)P(X = 1) + P(y = 1|X = 0)P(X = 0)} = \frac{P(X = 1) \times 0.99}{0.99 \times P(X = 1) + 0.01 \times (1 - P(X = 1))}$$

If $P(X=1)=0.001$, $P(X=1,y=1)=0.09$.

Contents

◆ Linear Algebra

◆ **Probability Theory and Information Theory**

- Basic Concepts of Probability Theory
- Random Variables and Their Distribution Functions
- Numerical Characteristics of Random Variables
- Information Theory

◆ Numeric Calculation



Expectation and Variance

◆ Mathematical expectation (or mean, also referred to as expectation):

- For discrete random variable: $E(X) = \sum_{k=1}^{\infty} x_k p_k, k = 1, 2, \dots$.
- For continuous random variable: $E(X) = \int_{-\infty}^{\infty} xf(x)dx.$

◆ **Variance:** A measure of the degree of dispersion in which the probability theory and statistical variance measure random variables or a set of data. According to the probability theory, variance measures the deviation between the random variable and its mathematical expectation.

$$D(X) = Var(X) = E\{[X - E(X)]^2\}$$

In addition, $\sqrt{D(X)}$, expressed as $\sigma(X)$, is called standard variance or mean variance.

$X^* = \frac{X - E(X)}{\sigma(X)}$ is called standard variable for X .



Covariance, Correlation Coefficients, and Covariance Matrices

◆ Covariance:

$$\text{Cov}(x, y) = E((X - E(X))(Y - E(Y)))$$

◆ correlation coefficient:

$$\rho_{XY} = \frac{\text{Cov}(X,Y)}{\sqrt{D(X)}\sqrt{D(Y)}}.$$

◆ Covariance matrices for random variable (X_1, X_2) :

$$C = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$$

where $c_{ij} = \text{Cov}(X_i, X_j) = E\{[X_i - E(X_i)][X_j - E(X_j)]\}, i, j = 1, 2, \dots, n.$

Contents

◆ Linear Algebra

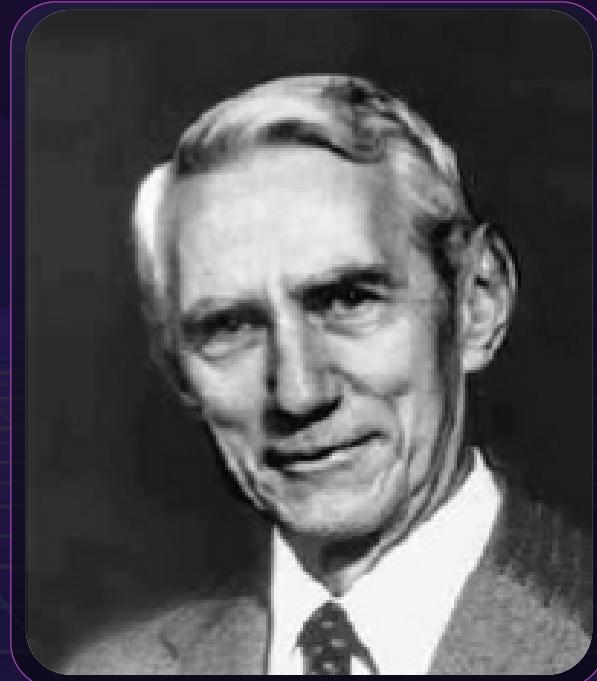
◆ **Probability Theory and Information Theory**

- Basic Concepts of Probability Theory
- Random Variables and Their Distribution Functions
- Numerical Characteristics of Random Variables
- **Information Theory**

◆ Numeric Calculation

Information Theory

As a branch of applied mathematics, **information theory** mainly studies how to measure information contained in a signal. The sign of information theory was the publication of Shannon's paper, "A Mathematical Theory of Communication" in 1948. In this paper, Shannon creatively used probability theory to study communication problems, gave a scientific and quantitative description of information, and for the first time proposed the concept of **information entropy**.



Information Quantity

- ◆ The following conditions should be met to define **self-information** $I(x)$ for event $X=x$:
 - $f(p)$ should be a strictly monotonic decreasing function of probability, that is, $p_1 > p_2$, $f(p_1) < f(p_2)$.
 - When $p = 1$, $f(p) = 0$.
 - When $p = 0$, $f(p) = \infty$.
 - The joint information content of two independent events should be equal to the sum of their respective information quantity.

Therefore, if the probability of a message is p , the information quantity contained in this message is:

$$I(x) = -\log_2 p.$$

- ◆ Example: If you throw a coin, the **information quantity** about the coin showing the front or opposite is $I(\text{front})= I(\text{opposite})= 1\text{bit}$.



Information Entropy

- ◆ The information contained in the source is the average uncertainty of all possible messages transmitted by the source. Shannon, the founder of Information theory, refers to the amount of content that the source contains as information entropy, which is the statistical average of the amount of content in data partition D . The information entropy for the classification of m tuples in D is calculated as follows:

$$Info(D) = - \sum_{i=1}^m p_i \log_2(p_i).$$

where p_i is a none-zero probability that any tuple in D belongs to class C_i , $p_i = \frac{|C_{i,D}|}{|D|}$.

- ◆ For example, what is the entropy of throwing a coin?

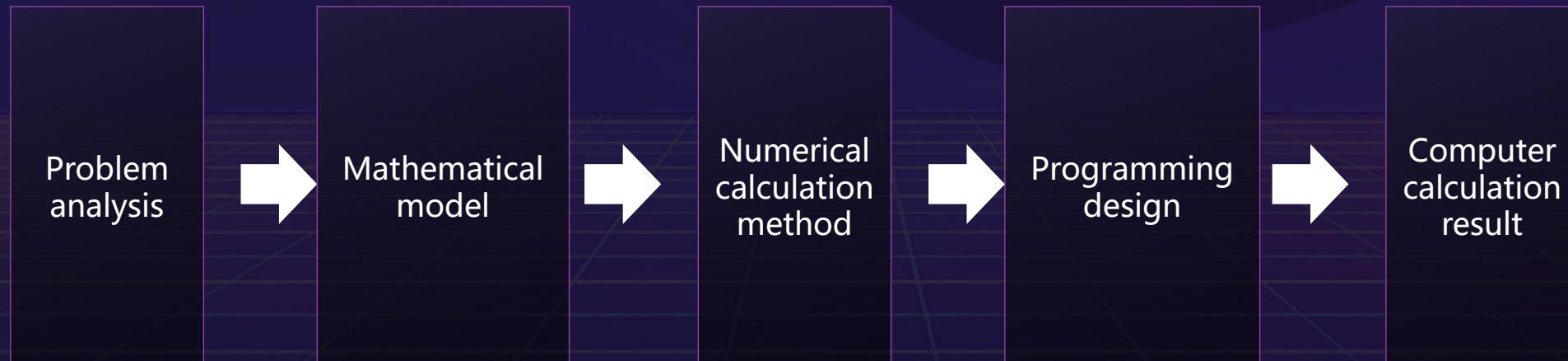
$$Info(D) = - \left(\frac{1}{2} \log_2 \frac{1}{2} + \frac{1}{2} \log_2 \frac{1}{2} \right) = 1 \text{bit.}$$

Contents

- ◆ Linear Algebra
- ◆ Probability Theory and Information Theory
- ◆ **Numerical Calculation**
 - Basic Concepts
 - Classification of and Solutions to the Optimization Problem

Numerical Calculation

- ◆ **Numerical calculation:** Refers to the method and process of effectively using a digital computer to solve approximate solutions of mathematical problems, and the disciplines formed by related theories. The process of solving practical problems with computers is as follows:





Overflow and Underflow

- ◆ **Underflow:** An underflow occurs when a number approximate to 0 is rounded to zero. Many functions show a qualitative difference when their arguments are zero rather than a small positive number.
- ◆ **Overflow:** Overflow occurs when a large number is approximated to ∞ or $-\infty$. Further operations usually cause these infinite values to become non-numeric.
- ◆ **The large number "swallows" the small number:** When $a \gg b$, $a + b = a$, a numerical abnormality occurs.
- ◆ The **Softmax** function can **numerically stabilize** overflow and underflow:

$$\text{softmax}(x)_i = \frac{\exp(x_i)}{\sum_{j=1}^n \exp(x_j)}.$$



Number of III-Conditions

- ◆ **III-condition number:** Refers to the speed for a function to change with small changes of input.
- ◆ Considering function $f(x) = A^{-1}x$, when $A \in \mathbb{R}^{n \times n}$ has feature decomposition, the number of conditions is:

$$\max_{i,j} \left| \frac{\lambda_i}{\lambda_j} \right|$$

This is the modulus ratio of the maximum and minimum eigenvalues. When this ratio is large, matrix inversion is particularly sensitive to input errors.

Contents

- 
- ◆ Linear Algebra
 - ◆ Probability Theory and Information Theory
 - ◆ **Numerical Calculation**
 - Basic Concepts
 - Classification of and Solutions to the Optimization Problem

Optimization Problem

- ◆ Optimization problem: It can be expressed as

$$\min(\max) f(x)$$

s.t. $g_i(x) \geq 0, i=1, 2, \dots, m$, inequality constraints

$h_j(x)=0, j=1, 2, \dots, p$, equality constraints

where $x = (x_1, x_2, \dots, x_n)^T \in R^n$. We refer to $f(x)$ as the objective function or guideline, or as a **cost function, loss function**, or **error function** when minimizing it.



Classification of Optimization Problems (1)

- ◆ **Constraint optimization:** a branch of optimization problems. Sometimes, the maximized or minimized $f(x)$ function under all possible values is not what we desire. Instead, we might want to find the maximum or minimum value of $f(x)$ when x is in a certain collection s . The points within the collection s are called **feasible points**.
- ◆ **With no constraints, it can be expressed as:**

$$\min f(x)$$

The common method is Fermat theorem. If $f'(x) = 0$, the critical point is obtained. Then, verify that the extreme value can be obtained at the critical point.

- ◆ **With equality constraints, it can be expressed as:**

$$\begin{aligned} & \min f(x) \\ & \text{s. t. } h_i(x) = 0, i = 1, 2, \dots, n. \end{aligned}$$

The common method is Lagrange multiplier method, that is, introducing n Lagrange multipliers λ to construct Lagrange function $L(x, \lambda) = f(x) + \sum_{i=1}^n \lambda_i h_i(x)$ and then seeking the partial derivative of each variable to be zero. Then, we can get the collection of candidate values, and get the optimal value through verification.



Classification of Optimization Problems (2)

- ◆ With inequality constraints, it can be expressed as:

$$\begin{aligned} & \min f(x) \\ \text{s.t. } & h_i(x) = 0, i = 1, 2, \dots, n, \\ & g_j(x) \leq 0, j = 1, 2, \dots, m. \end{aligned}$$

A common method is to introduce new variables λ_i and α_j , to **Generalized Lagrangian functions** based on all equality, inequality constraints and $f(x)$.

$$L(x, \lambda, \alpha) = f(x) + \sum_i \lambda_i h_i(x) + \sum_j \alpha_j g_j(x),$$

We can use a set of simple properties to describe the most advantageous properties of constrained optimization problems, which are called KKT (kuhn-kuhn-tucker) conditions.

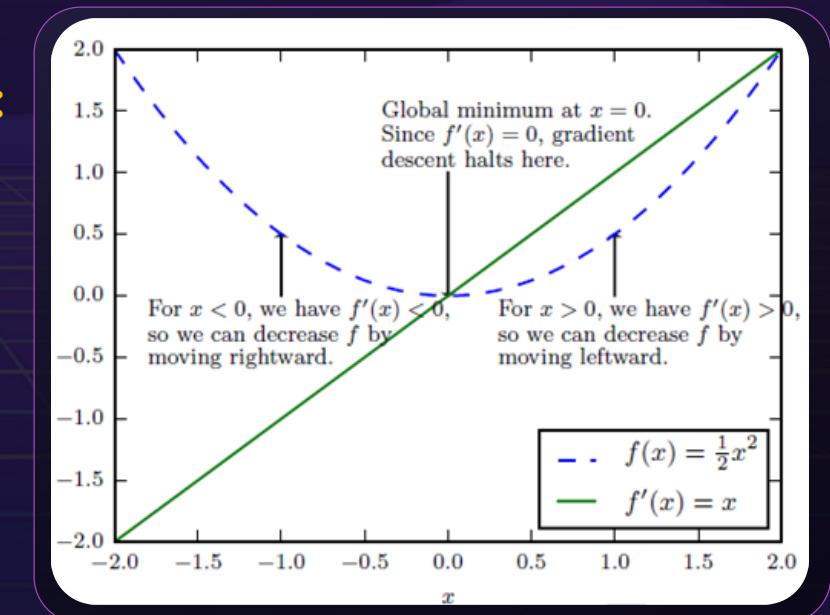
- The gradient of the generalized Lagrangian is 0.
- All constraints on x and KKKT multiplier are met.
- Inequality constraints show "complementary slackness type": $\alpha \odot h(x) = 0$.

Gradient Based Optimization Method (1)

◆ **Gradient descent:** The derivative indicates how to change x to slightly improve y . For example, we know that $f(x - \Delta x \text{sign}(f'(x)))$ is smaller than $f(x)$ for Δx that is small enough. So we can move x in the opposite direction of the derivative by a small step to reduce $f(x)$. This technique is called gradient descent.

◆ The extremum problem of a one-dimensional function:

- The local extremum point of the function means that $f(x)$ cannot be reduced or increased by moving x .
- The point where $f'(x) = 0$ is called a critical point or a stationary point.
- The extremum point of a function must be a stationary point, but a stationary point may not be the extremum point.



Gradient Based Optimization Method (2)

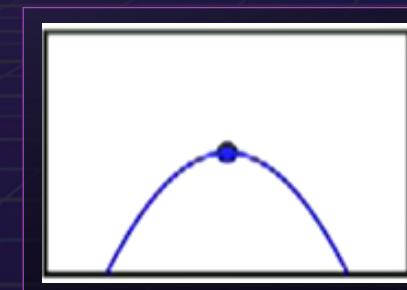
◆ **Convex function:** For $\lambda \in (0,1)$, given arbitrary $x_1, x_2 \in R$, the following applies:

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2)$$

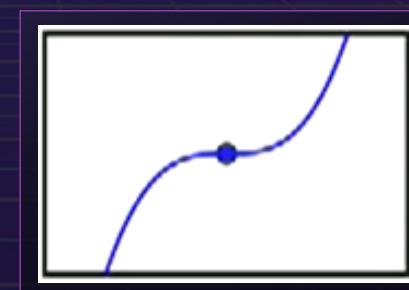
Then, $f(x)$ is called a convex function. The extremum point of the convex function is present at the stationary point.



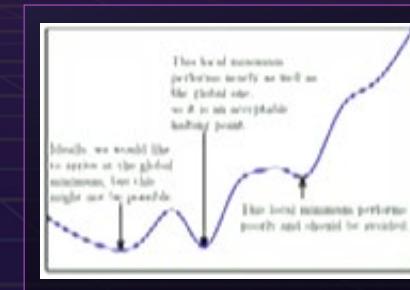
(a)



(b)



(c)



(d)

Gradient Based Optimization Method (3)

- ◆ To the case of multidimensional functions, the partial derivative is used to describe the degree of variation of the function relative to the respective variable.
- ◆ **Gradient:** It is a derivative relative to vector X , and is expressed as $\nabla_x f(x)$. The derivative of $f(x)$ in the direction of u (unit vector) is $u^T \nabla_x f(x)$.
- ◆ For a task to minimize $f(x)$, we want to find the direction with the fastest downward change, where θ is the angle between u and gradient $\nabla_x f(x)$.

$$\begin{aligned} & \min_{u, u^T u=1} u^T \nabla_x f(x) \\ & = \min_{u, u^T u=1} \|u\|_2 \|\nabla_x f(x)\|_2 \cos \theta \end{aligned}$$

- ◆ You can see that the direction in which $f(x)$ value decreases the maximum is the negative direction of the gradient.

Gradient Based Optimization Method (4)

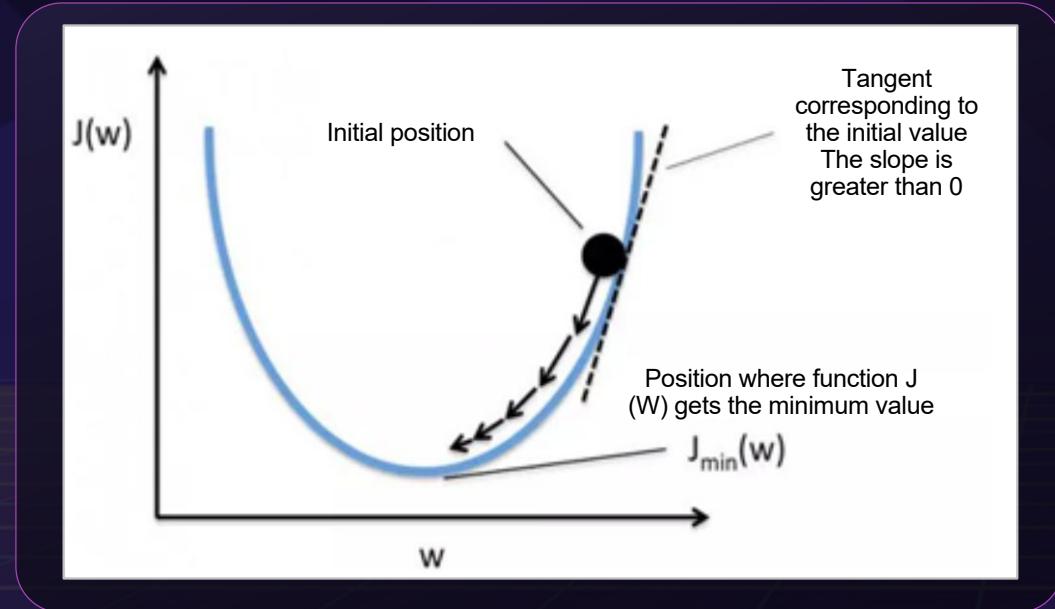
- ◆ A positive gradient vector points uphill, and a negative gradient vector points downhill.
A move in the negative gradient direction can reduce $f(x)$, which is called method of steepest descent or gradient descent.
- ◆ Under the gradient descent method, the update point is proposed as:

$$x' = x - \varepsilon \nabla_x f(x)$$

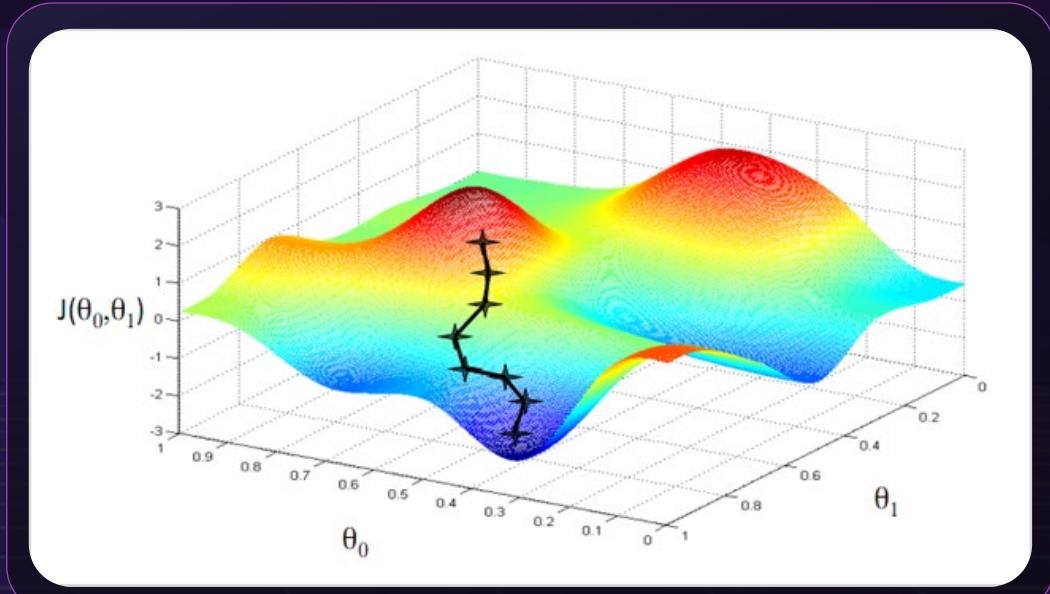
where ε is the learning rate, which is a positive scalar with a fixed step length.

- ◆ Iteration converges when the gradient is zero or approaching zero.

Gradient Based Optimization Method (5)



Two-dimensional space



Three-dimensional space



What are the relations and differences between
a distribution function, distribution law and
density function of a random variable?



Quiz

1. (Single-Choice) Matrix A has 3 rows and 2 columns. Matrix B has 2 rows and 3 columns. Matrix C has 3 rows and 3 columns. Which of the following operations makes sense? ()

A AC

B BC

C A+B

D AB-BC

2. (Single-Choice) X and Y are random variables, and C is a constant. Which of the following descriptions of the properties of mathematical expectations is incorrect? ()

A $E(C)=C$

B $E(X+Y)=E(X)+E(Y)$

C $E(CX)=CE(X)$

D $E(XY)=E(X)E(Y)$



Quiz

3. (True or False) Principal component analysis (PCA) is a statistical method. By means of orthogonal transformation, a group of variables that may have correlations are converted to a group of linearly related variables, and the converted group of variables is called principal component. ()

A True

B False

4. (True or False) The correlation coefficient, also called the linear correlation coefficient, is used to measure the linear relationship between two variables, and the value is a real number greater than 0. ()

A True

B False

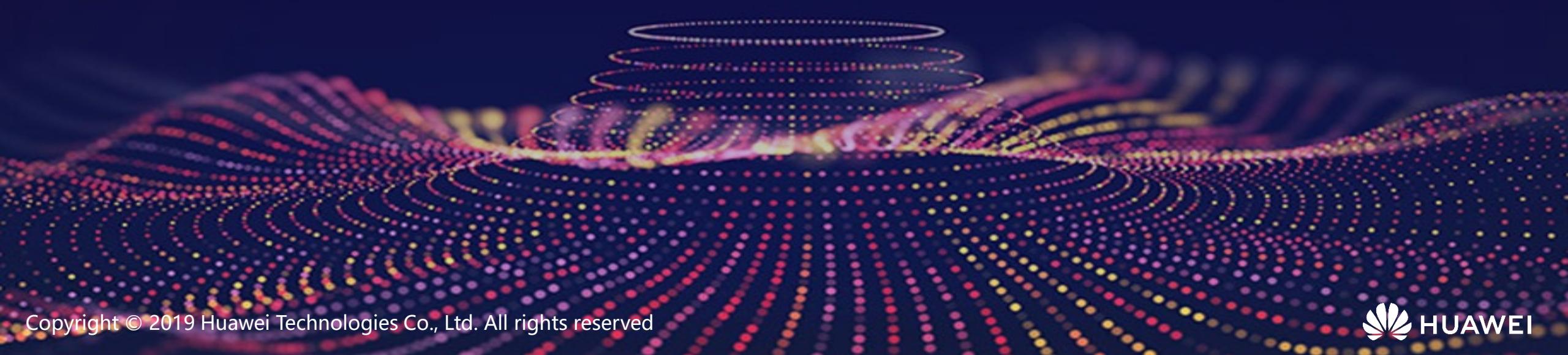


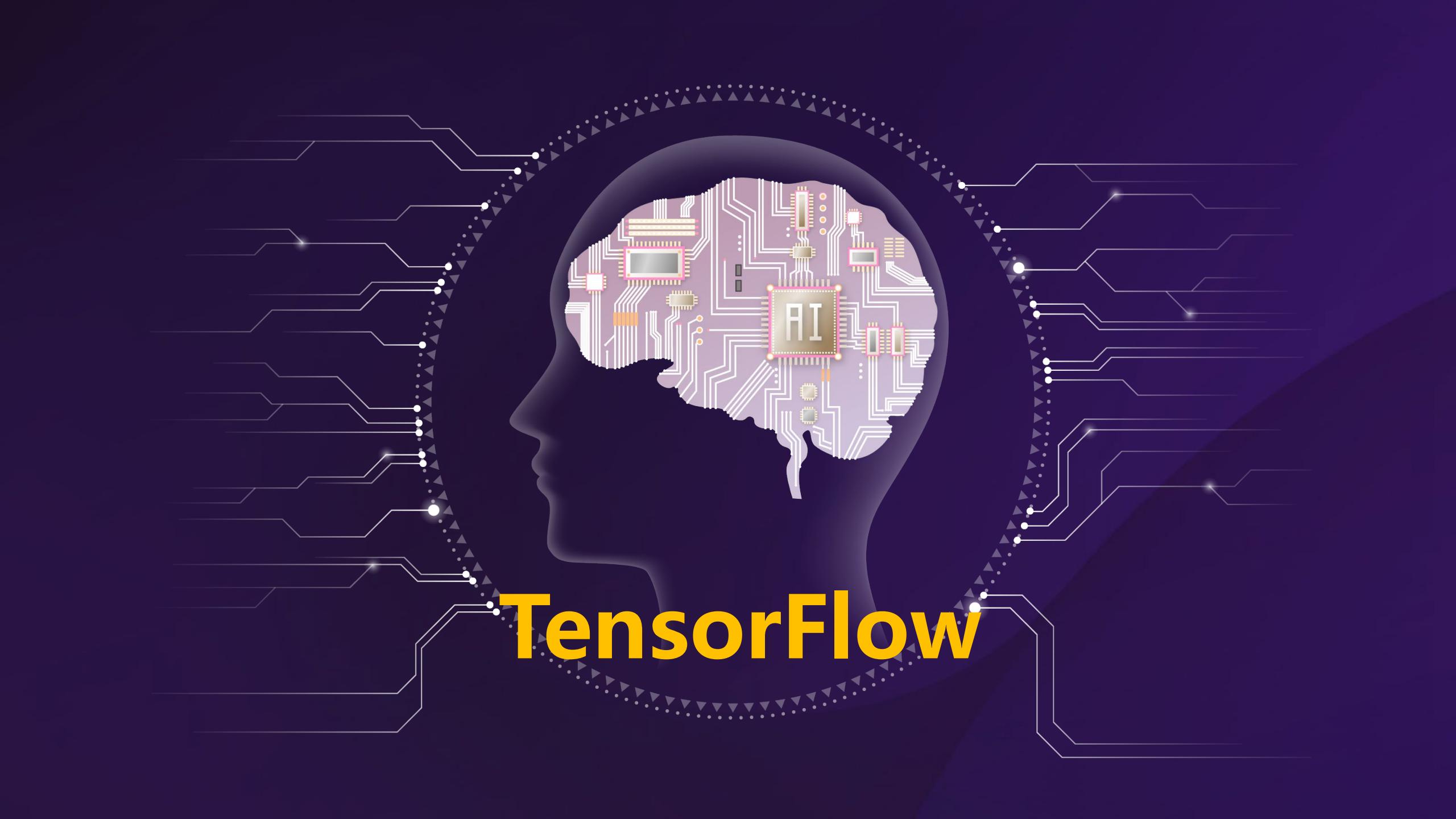
Summary

This chapter mainly describes the basics of deep learning, covering linear algebra, probability and information theory, and numerical calculation, and builds a foundation for further learning.

Thanks

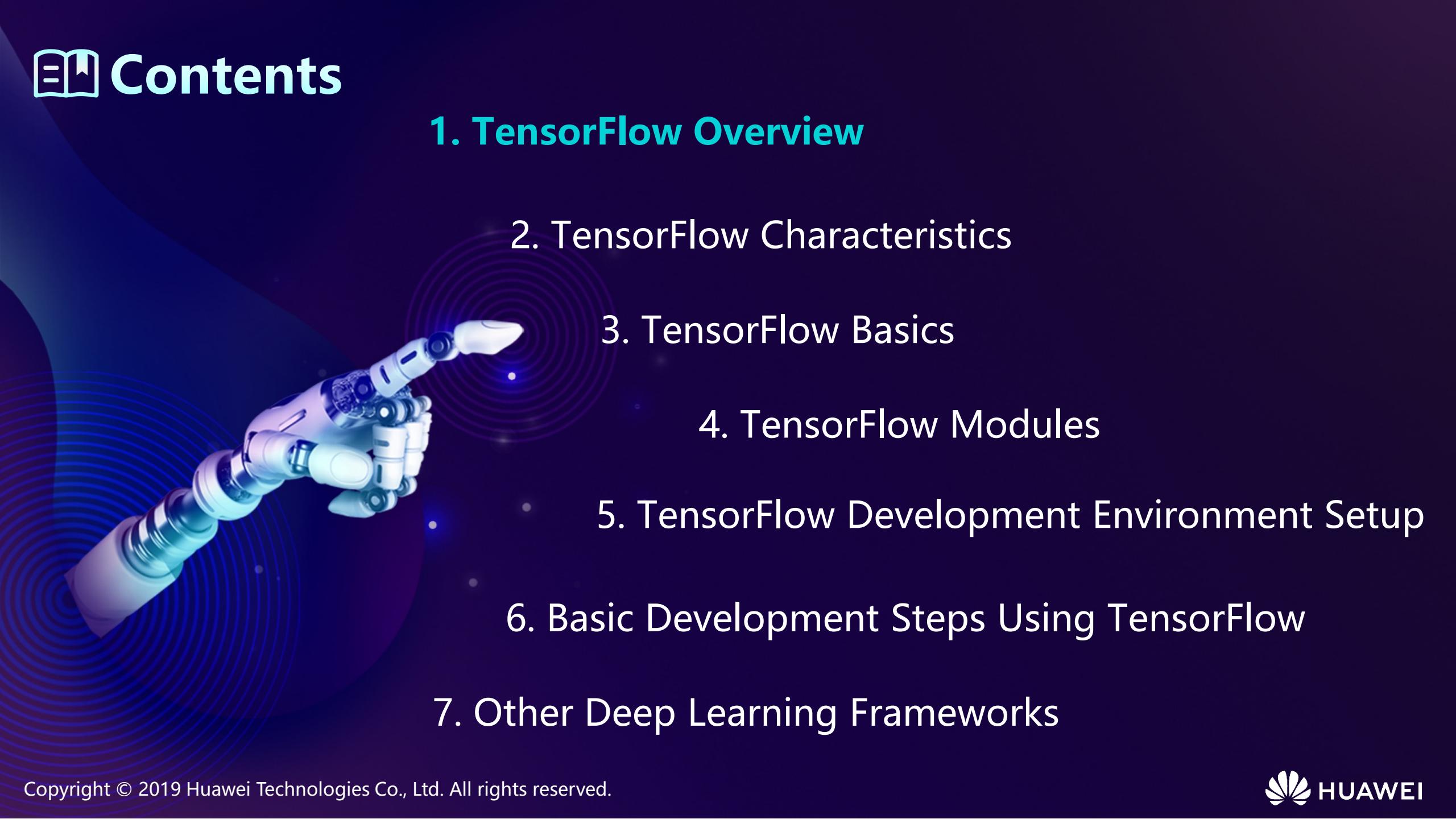
www.huawei.com





TensorFlow

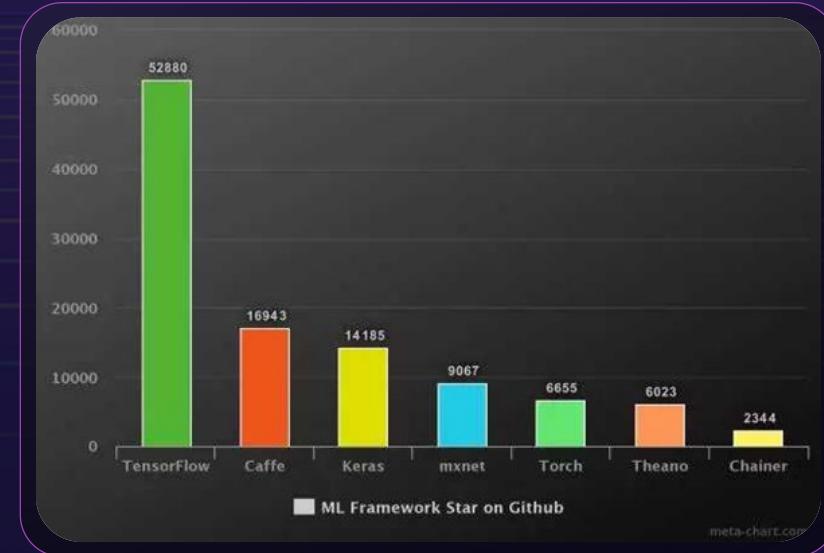
Contents

- 
- 1. TensorFlow Overview**
 - 2. TensorFlow Characteristics**
 - 3. TensorFlow Basics**
 - 4. TensorFlow Modules**
 - 5. TensorFlow Development Environment Setup**
 - 6. Basic Development Steps Using TensorFlow**
 - 7. Other Deep Learning Frameworks**



What Is TensorFlow

- ◆ TensorFlow is Google's second-generation open-source software library for dataflow computing.
- ◆ The TensorFlow framework supports various deep learning algorithms, as well as many computing platforms other than those for deep learning.
- ◆ TensorFlow is open-source, which facilitates maintenance and update and improves the development efficiency.





Quiz

1. TensorFlow is a frame designed for deep-learning? ()
- A. True
 - B. False

Contents

1. TensorFlow Overview

2. TensorFlow Characteristics

3. TensorFlow Basics

4. TensorFlow Modules

5. TensorFlow Development Environment Setup

6. Basic Development Steps Using TensorFlow

7. Other Deep Learning Frameworks



TensorFlow Characteristics

Flexible and
scalable

GPU

Powerful
computation



Multi-language

Cross-platform

Distributed

What Can We Do Using TensorFlow



Self-driving cars



Image recognition



Language models



Automated theorem proving



Music creation



Speech recognition



Human activity recognition



Gaming



What Can We Do Using TensorFlow (2)

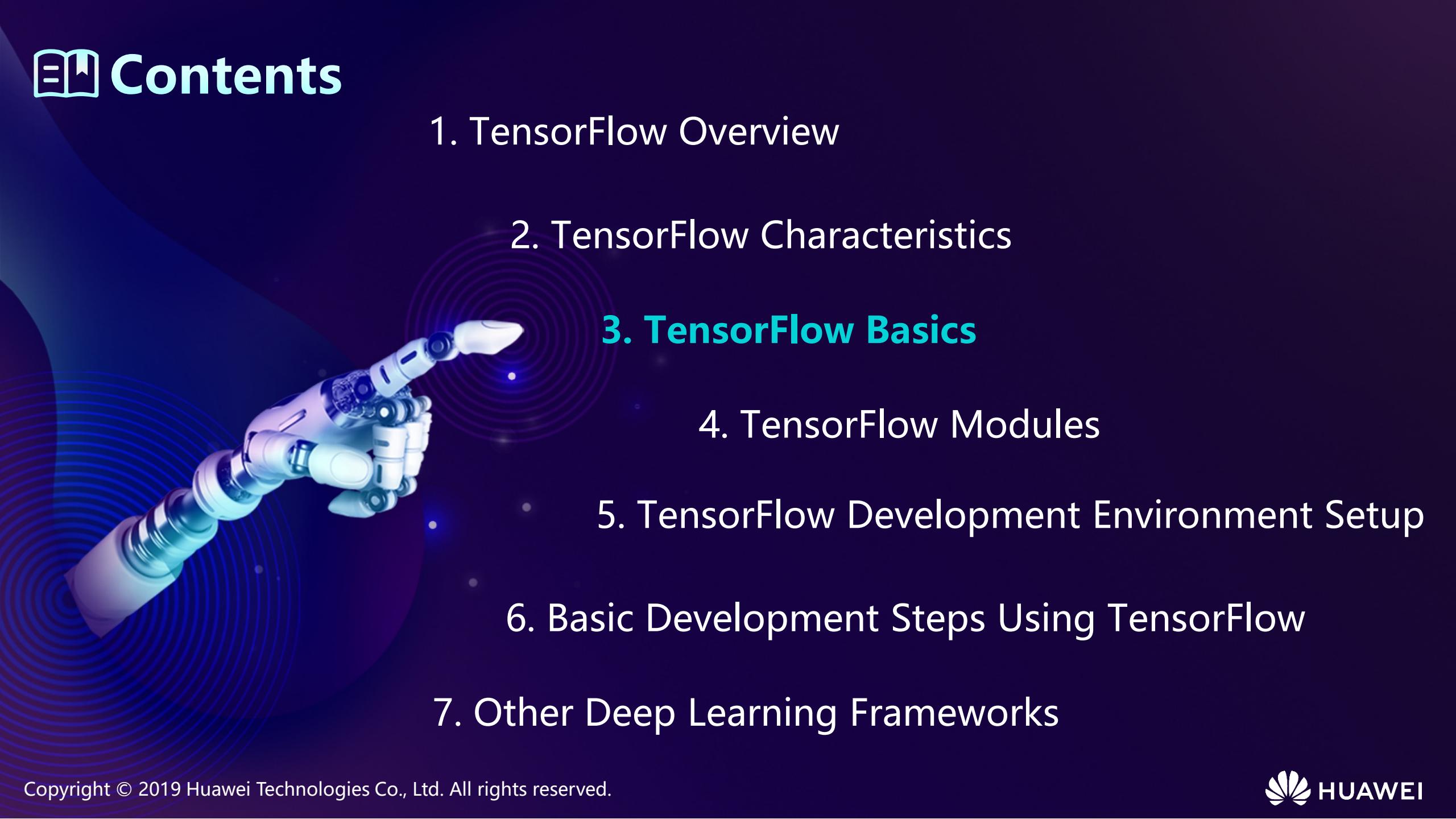


Artistic style transfer

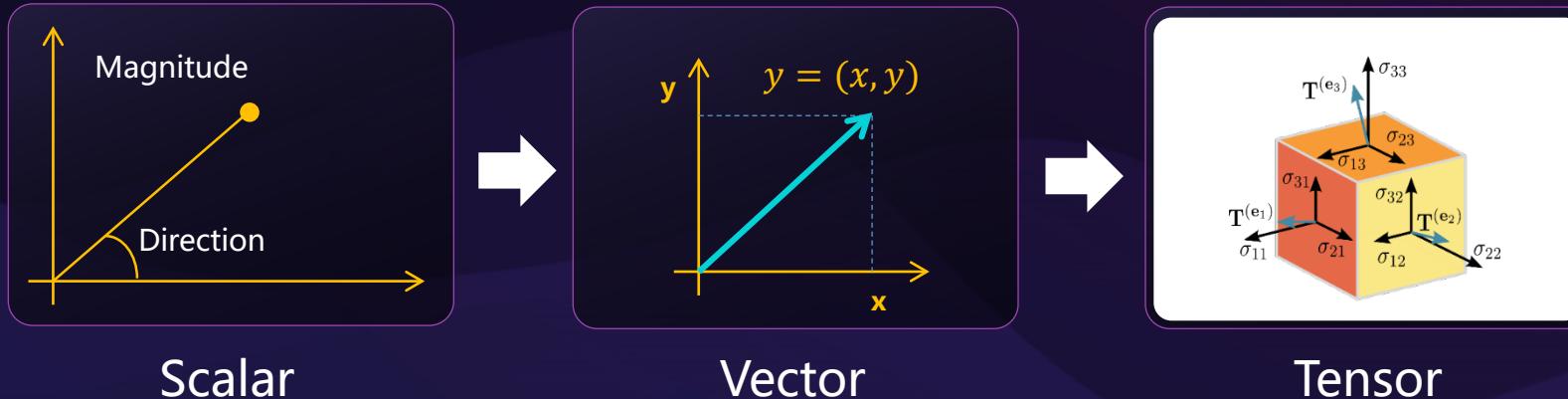


Facial recognition

Contents

- 
1. TensorFlow Overview
 2. TensorFlow Characteristics
 - 3. TensorFlow Basics**
 4. TensorFlow Modules
 5. TensorFlow Development Environment Setup
 6. Basic Development Steps Using TensorFlow
 7. Other Deep Learning Frameworks

TensorFlow



Scalar

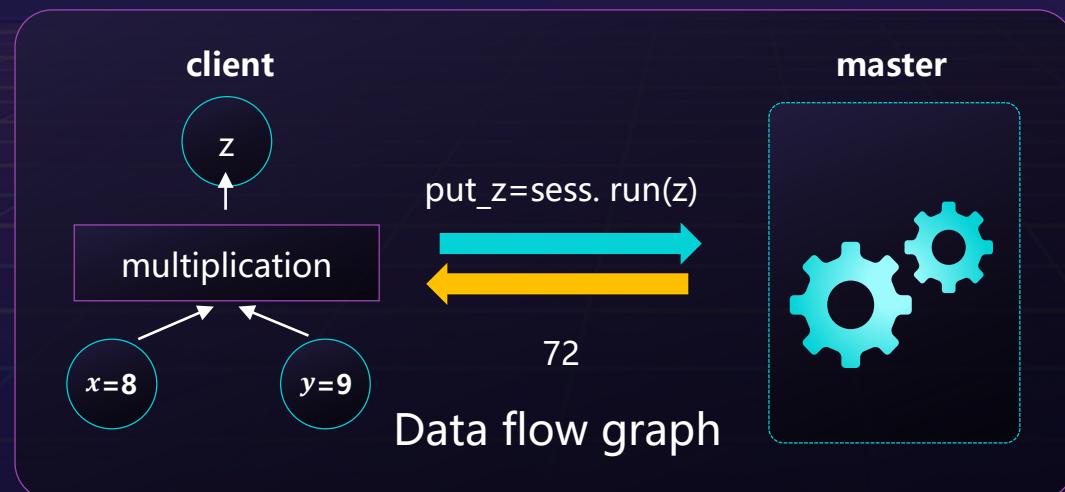
Vector

Tensor

Tensor

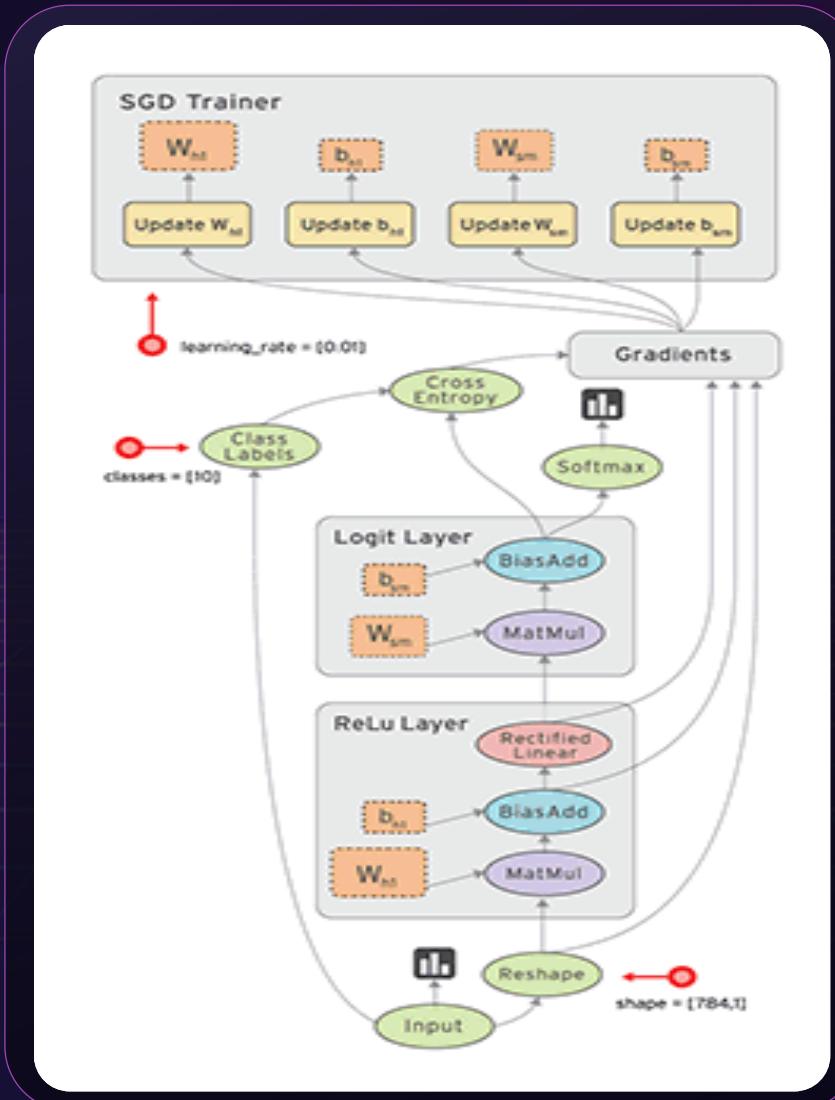


Flow





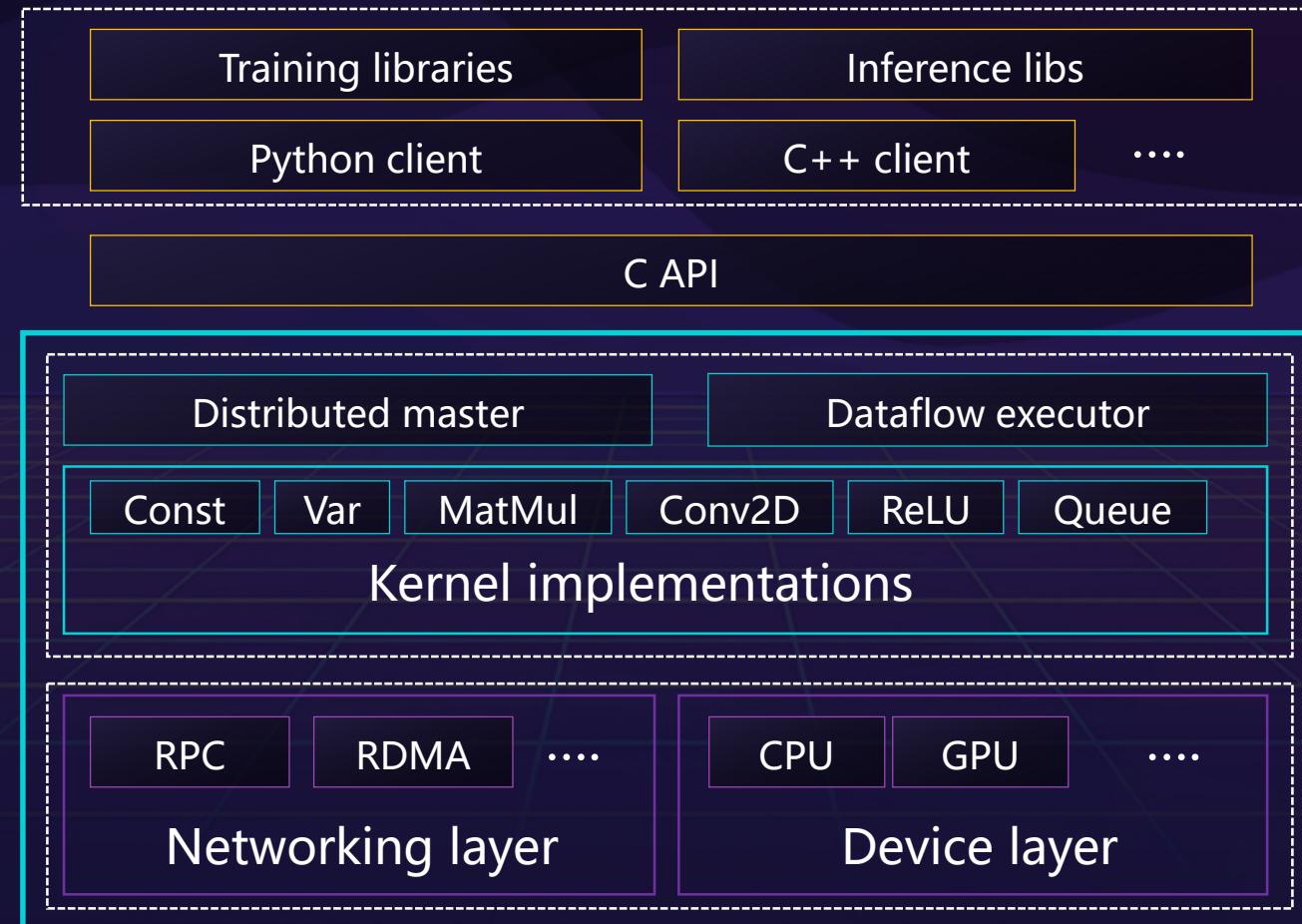
TensorFlow Computation Process





TensorFlow Architecture

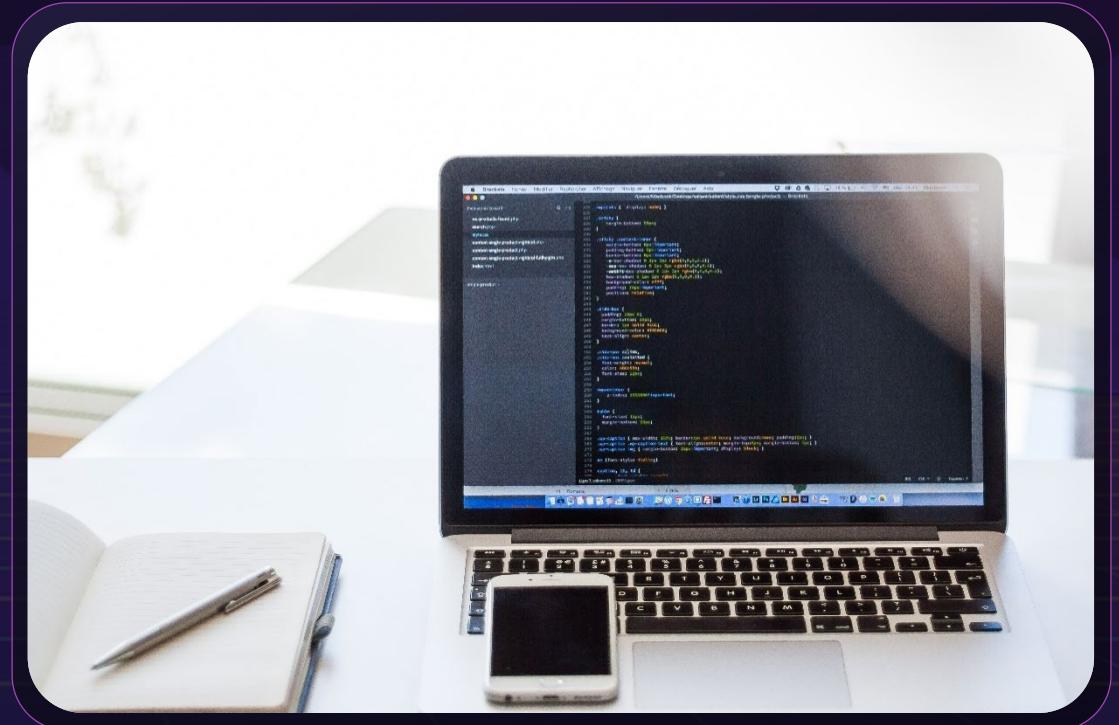
Layer-by-layer breakdown and decoupling





Basic Concepts of TensorFlow

- ◆ **tensor**
- ◆ **graph**
 - node
 - edge
- ◆ **operation**
- ◆ **session**
 - feed
 - fetch
- ◆ **variable**





tensor

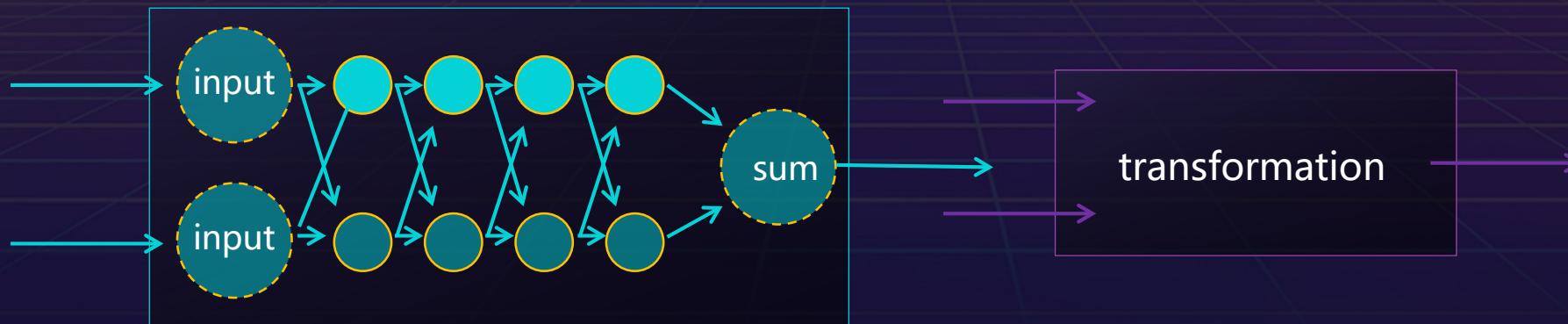
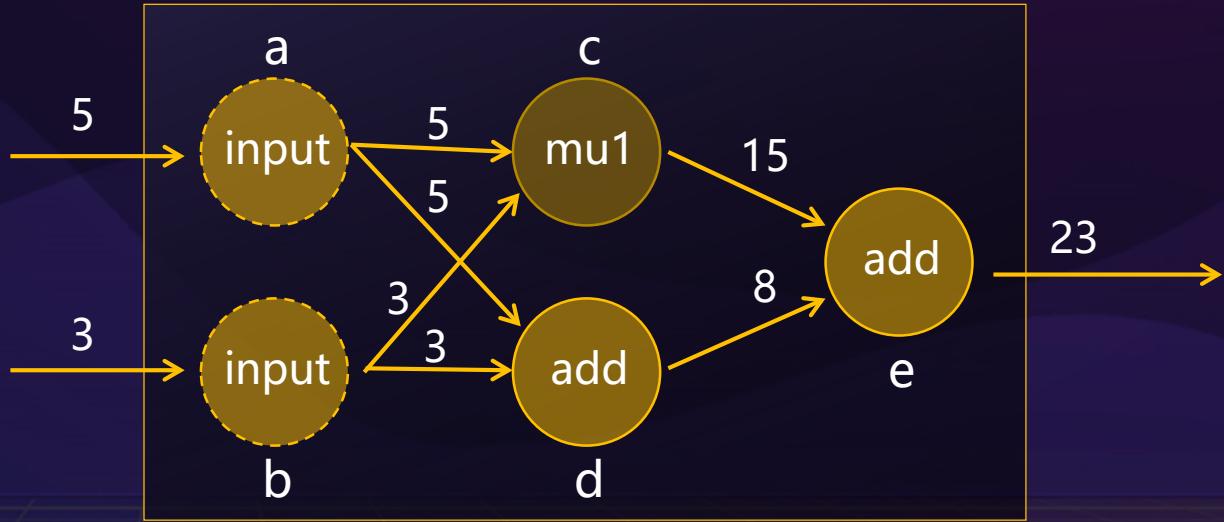
- ◆ Tensor is the primary data structure in TensorFlow programs. Tensors are N-dimensional (where N could be 1, 2, 3, 4, or very large) data structures. In a running graph, tensors are the data that flows between nodes.
- ◆ The data structure contained in a tensor is: name + shape + type.

```
224 #wpstats { display: none; }
225
226 .sticky {
227   margin-bottom: 50px;
228 }
229
230 .sticky .content-inner {
231   margin-bottom: 0px !important;
232   padding-bottom: 0px !important;
233   border-bottom: 0px !important;
234   -o-box-shadow: 0 1px 2px rgba(0,0,0,.2);
235   -moz-box-shadow: 0 1px 2px rgba(0,0,0,.2);
236   -webkit-box-shadow: 0 1px 2px rgba(0,0,0,.2);
237   box-shadow: 0 1px 2px rgba(0,0,0,.2);
238   background-color: #ffff;
239   padding: 25px !important;
240   position: relative;
241 }
242
243 .side-box {
244   padding: 10px 0;
245   margin-bottom: 16px;
246   border: 1px solid #ccc;
247   background-color: #E6E6E6;
248   text-align: center;
249 }
250
251 .side-box a:link,
252 .side-box a:visited {
253   font-weight: normal;
254   color: #00c55b;
255   font-size: 12px;
256 }
```



Graph (Data Flow Graph)

- ◆ Node: A data operation (OP) in the TensorFlow graph
- ◆ Edge: Data and its dependency relationships

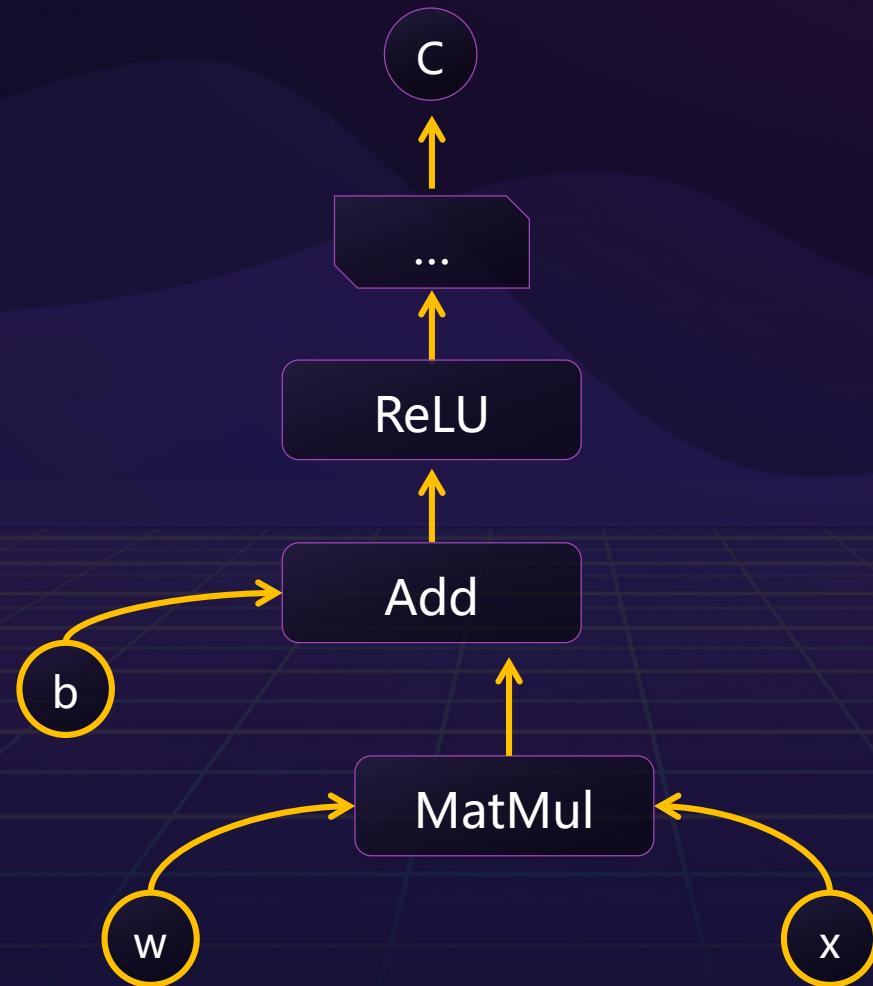




TensorFlow Operators

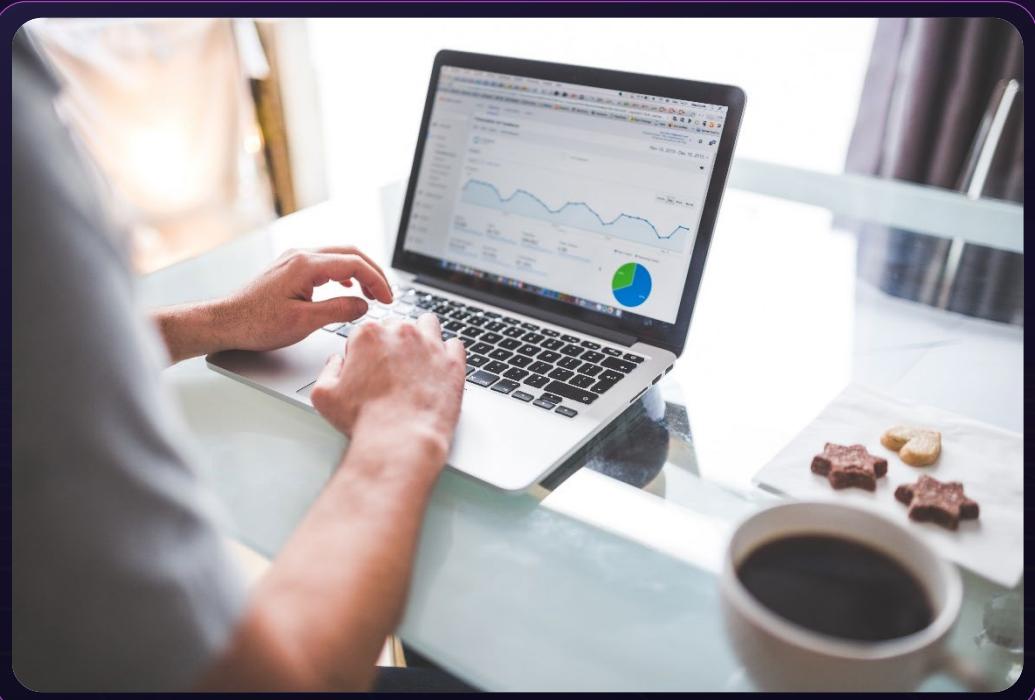
Category	Examples
Element——wise mathematical Operations	Add, Sub, Div, Exp, Log, Greater, Less, Equal, ...
Array operations	Concat, Slice, Split, Constant, Rank, Shape, Shuffle, ...
Matrix operations	MatMul, MatrixInverse, MatrixDeterminant, ...
Stateful operations	Variable, Assign, AssignAdd, ...
Neural-net building blocks	SoftMax, Sigmoid, ReLU, Convolution2D, MaxPool, ...
Checkpointing operations	Save, Restore, ...
Queue and synchronization operations	Enqueue, Dequeue, Mutex Acquire, Mutex Release, ...
Control flow operations	Merge, Switch, Enter, Leave, NextIteration

Session



Feed

- ◆ The feed mechanism directly connects the tensor to a node in the graph, and temporarily replaces the tensor value on the node. A feed is not created when the graph is formed. Rather, it is applied for when graph execution is triggered. That is, the feed data is initialized in the form of parameters when the "run" or "eval" command is executed. After the execution is complete, the replaced feed data disappears, but the behavior of the node defined in the graph does not change. Generally, the feed mechanism is used together with `tf.placeholder ()`.

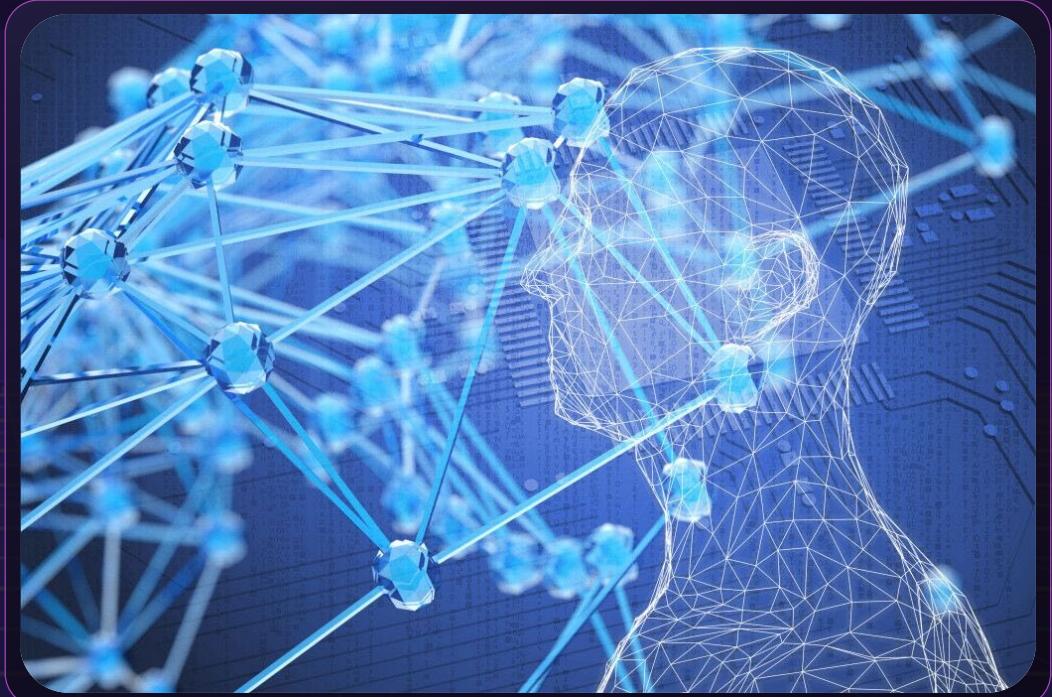




- ◆ The fetch mechanism retrieves the result of an operation in the graph. The fetch application occurs when graph execution is triggered, not when the graph is generated. To fetch the tensor value of one or more nodes, you can call the run () method on the session object and use the list of the nodes to be fetched as parameters to execute the graph.

Variable

- ◆ A variable maintains state across multiple calls to run during graph execution. For example, in a neural network, it is used to identify coefficients such as w and b. The variable in TensorFlow is actually a variable object in Python.
- ◆ The initial value of a variable in TensorFlow can be set to a random number, a constant, or a number calculated based on the initial values of other variables.

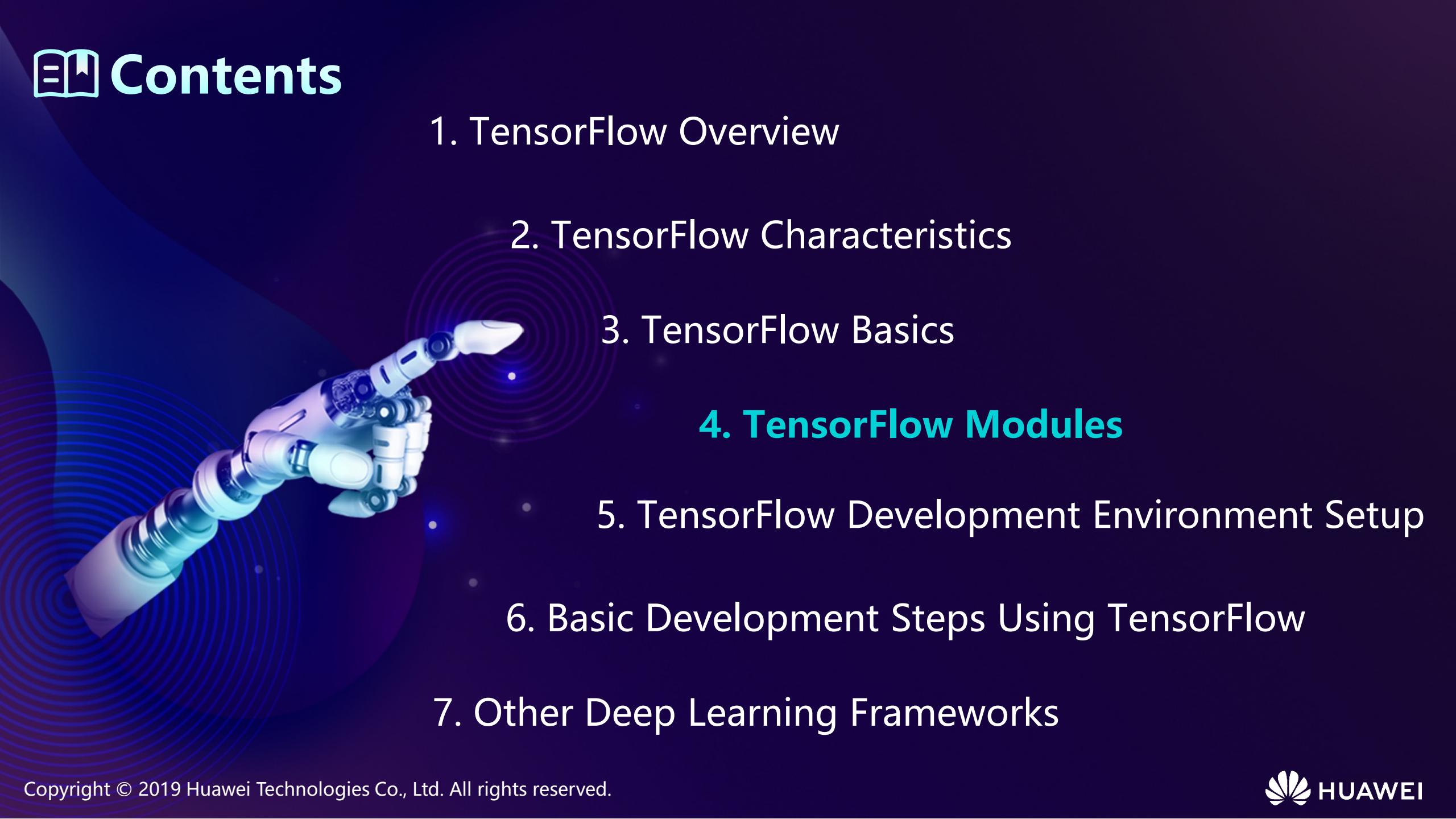




Quiz

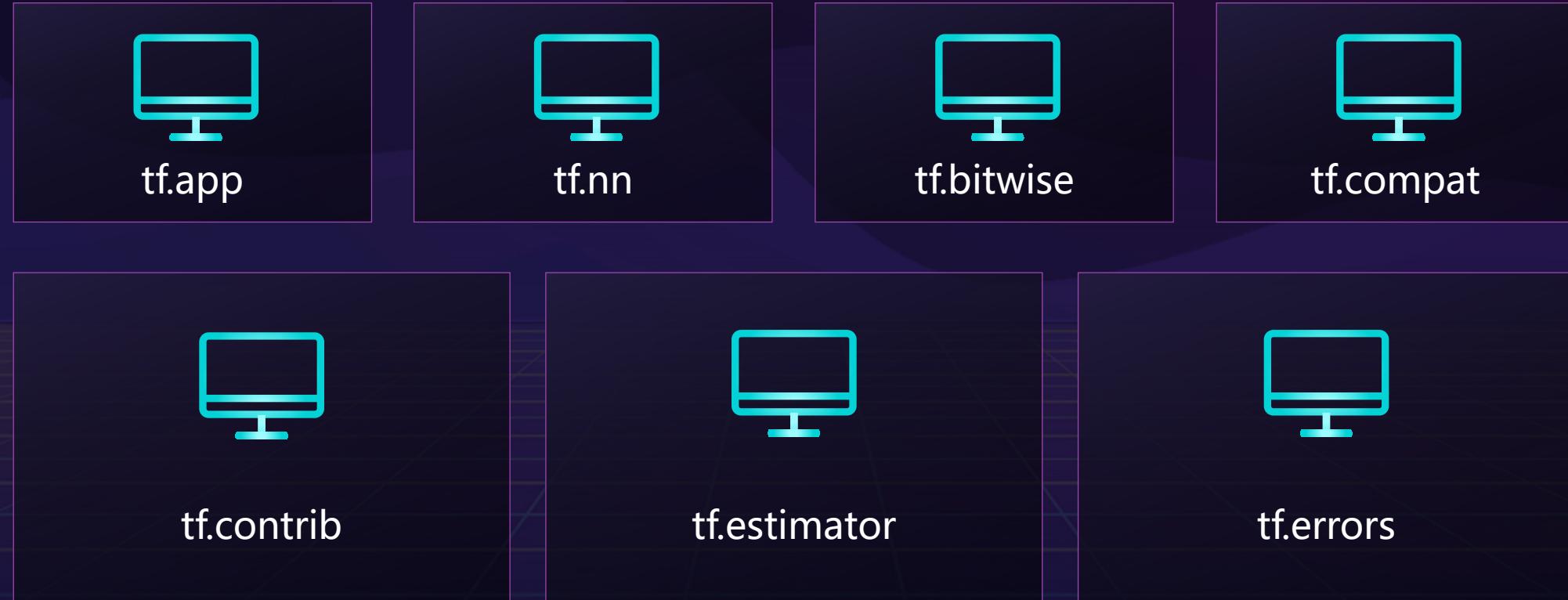
- 1. A session owns and manages all resources when the TensorFlow program is running. So, building session is required when performing any calculations. ()**
- A. True
- B. False

Contents

- 
1. TensorFlow Overview
 2. TensorFlow Characteristics
 3. TensorFlow Basics
 - 4. TensorFlow Modules**
 5. TensorFlow Development Environment Setup
 6. Basic Development Steps Using TensorFlow
 7. Other Deep Learning Frameworks



TensorFlow Modules (1)





TensorFlow Modules (2)

◆ **tf.nn:**

- tf.nn is used to create RNN. It is the most commonly used module for constructing classical convolutional networks. It includes a sub-module, rnn_cell, which is used to construct recurrent neural networks.

◆ **tf.estimator:**

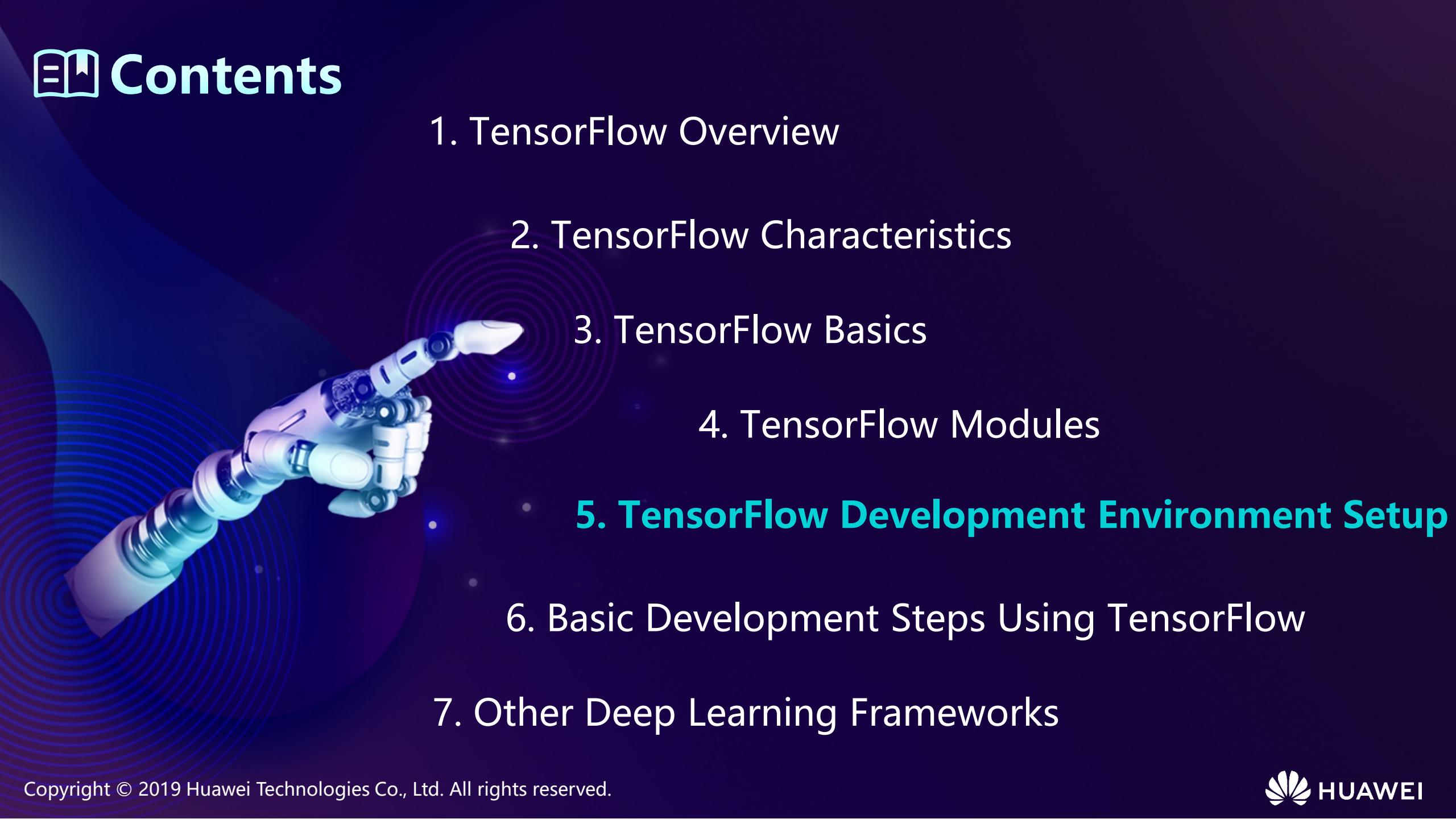
- Used to create one or more input functions.
- Defines the feature column of a model.
- Instantiates an estimator to define feature columns and various types of hyper parameters.
- Calls one or more methods on the estimator object and passes the appropriate input function as the data source.



TensorFlow Modules (3)

- ◆ **tf.Layers:** The network layer encapsulates variables and operations on the variables.
 - For example, the full connection layer performs a weighted sum operation on the input and can use an optional activation function. The weight and bias of a connection are managed by network layer objects.
- ◆ **tf.Contrib:** This module provides functions for computing streaming metrics.
 - The slim sub-module is the most commonly used in this module. All the functions that are experimental or easy to change are in this module, which has rich functional modules.
 - Tf. contrib provides advanced operations for computing layers in the calculation layer, regularization, summary operations, and calculations

Contents

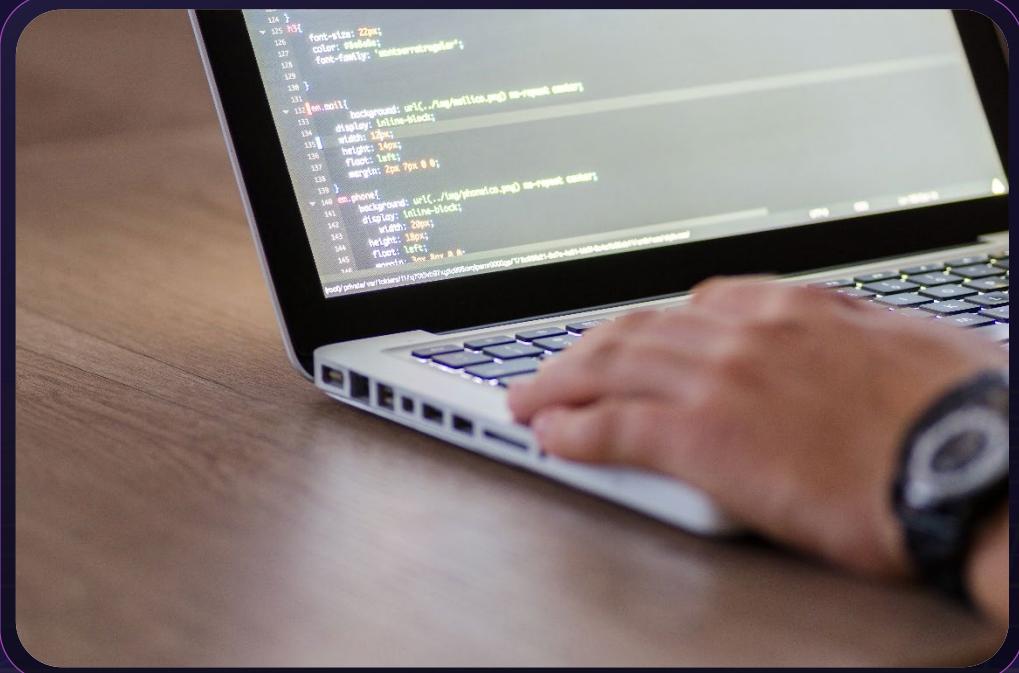
- 
1. TensorFlow Overview
 2. TensorFlow Characteristics
 3. TensorFlow Basics
 4. TensorFlow Modules
 5. **TensorFlow Development Environment Setup**
 6. Basic Development Steps Using TensorFlow
 7. Other Deep Learning Frameworks



Windows (1)

- ◆ Operating systems: Windows/MacOS/Linux
- ◆ Python 3 <https://www.python.org/downloads/release/python-350/>
- ◆ The Anaconda 3 (compatible with Python 3) contains the pip software.
- ◆ Installing TensorFlow
 - Installing the nightly package online (`pip install tf-nightly`)
 - Install the pure edition of TensorFlow (`pip install tensorflow`).
 - Install offline.

Windows (2)



- ◆ Installing the GPU version:
 - Install the CUDA software package (mapping with the TensorFlow version).
 - Install the cuDNN library (mapping with the TensorFlow version).
 - Test the graphics card.
 - Run the nvidia-smi command to view the graphics card information.
 - Check the CUDA version.



MacOS

- ◆ On the Mac OS X system, it is recommended that you install homebrew first and then run the brew install python command, so that you can use Python in homebrew to install TensorFlow. Another recommended method is to install TensorFlow in virtual env.

```
# In the current version, only CPU is supported
$ pip install https://storage.googleapis.com/tensorflow/mac/tensorflow-0.5.0-py2-none-any.whl
```



- ◆ The simplest installation mode on Linux is to use pip.

```
# Only the CPU version is used.
```

```
$ pip install https://storage.googleapis.com/tensorflow/linux/cpu/tensorflow-0.5.0-cp27-none-linux_x86_64.whl
```

```
# Start the version supporting GPU. (The version can only be installed after installing the CUDA sdk.
```

```
$ pip install https://storage.googleapis.com/tensorflow/linux/gpu/tensorflow-0.5.0-cp27-none-linux_x86_64.whl
```



Toolkit (1)

◆ Protocol Buffer

- Protocol Buffers serialize structured data into a binary stream.
- The precompiled format is a .proto file.
- ProtoBuf's storage space is 10%-30% of XML. But, its parsing time is 20-100 times faster.

```
name: Zhang San  
id: 12345  
email: zhangsan@abc.com
```

Toolkit (2)

◆ Bazel (automatic build tool)

- Compared with the traditional Makefile and Ant, Bazel is faster, more scalable, and flexible.
- The basic concept is workspace. It can be considered a folder that contains the source code required for software compilation and the soft-make address of the output.
- Only the py_binary, py_library, and py_test compilation modes are supported.

```
-rw-rw-r--  root  root  208   BUILD
-rw-rw-r--  root  root  48    hello_lib.py
-rw-rw-r--  root  root  47    hello_main.py
-rw-rw-r--  root  root  0     WORKSPACE
```

Running TensorFlow

```
$ python

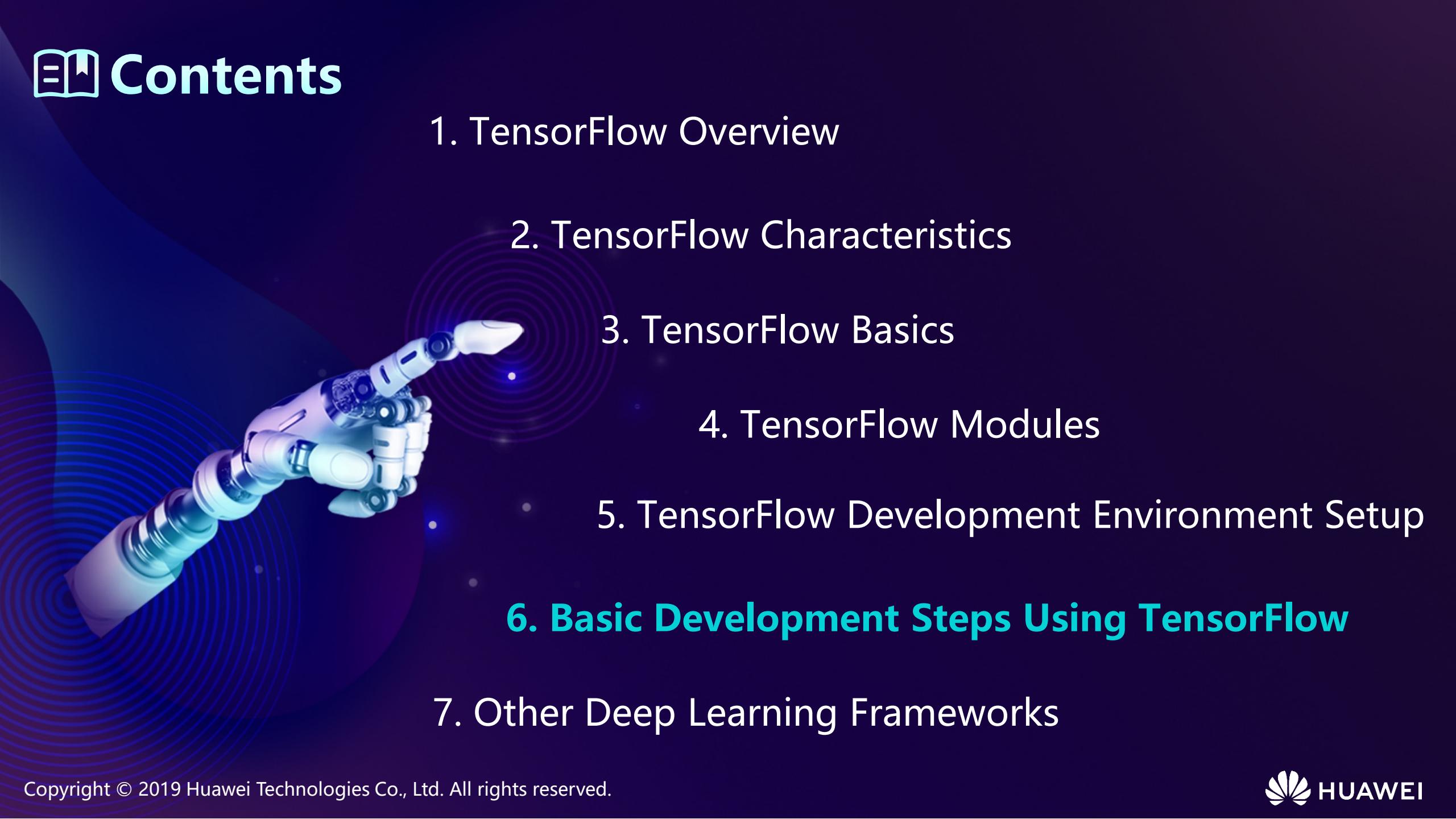
>>> import tensorflow as tf
>>> hello = tf.constant('Hello, TensorFlow!')
>>> sess = tf.Session()
>>> print sess.run(hello)
Hello, TensorFlow!
>>> a = tf.constant(10)
>>> b = tf.constant(32)
>>> print sess.run(a+b)
42
>>>
```



Quiz

1. The CPU version and GPU version of TensorFlow are separate. ()
- A. True
 - B. False

Contents

- 
1. TensorFlow Overview
 2. TensorFlow Characteristics
 3. TensorFlow Basics
 4. TensorFlow Modules
 5. TensorFlow Development Environment Setup
 - 6. Basic Development Steps Using TensorFlow**
 7. Other Deep Learning Frameworks



Development

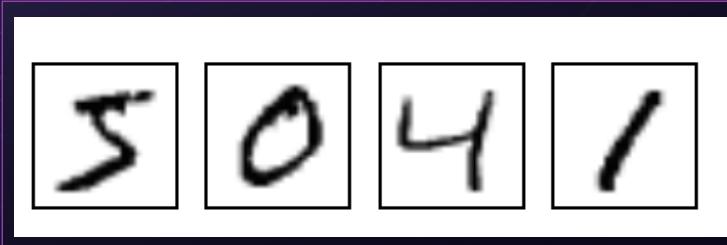
- ◆ Define the model (input node, “learning parameter” variables, operation, Optimize functions and objectives.)
- ◆ Initialize all variables.
- ◆ model training.
- ◆ Test the model.
- ◆ Use the model.





Preparing Data (1)

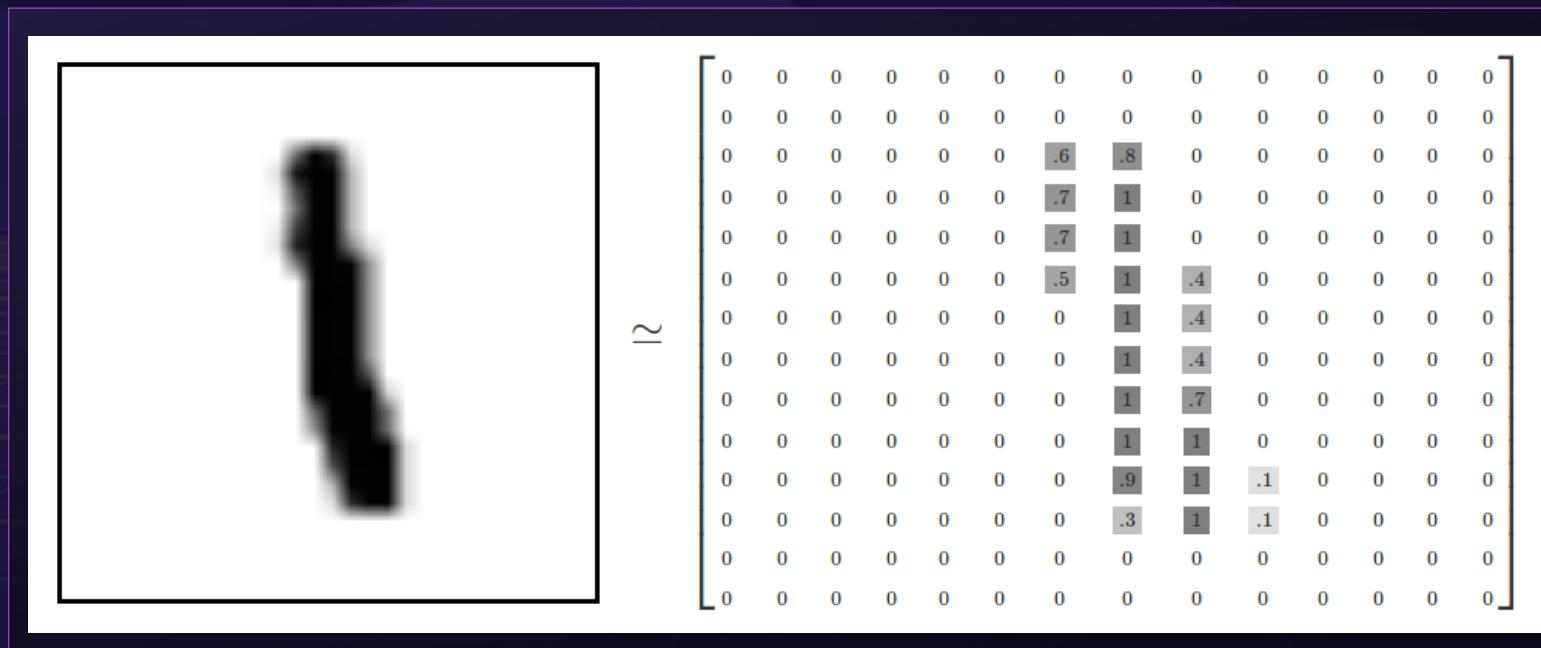
- ◆ MNIST is a classic problem in machine learning. The problem is solved by identifying 28 x 28 pixel grayscale images of handwritten digits as corresponding numbers ranging from 0 to 9.
- ◆ The MNIST dataset is an example of TensorFlow, you don't need to download it.





Preparing Data (2)

- ◆ An image is a matrix of 28 pixels x 28 pixels, which can be represented by a two-dimensional integer matrix of the same size.



MNIST Dataset (1)

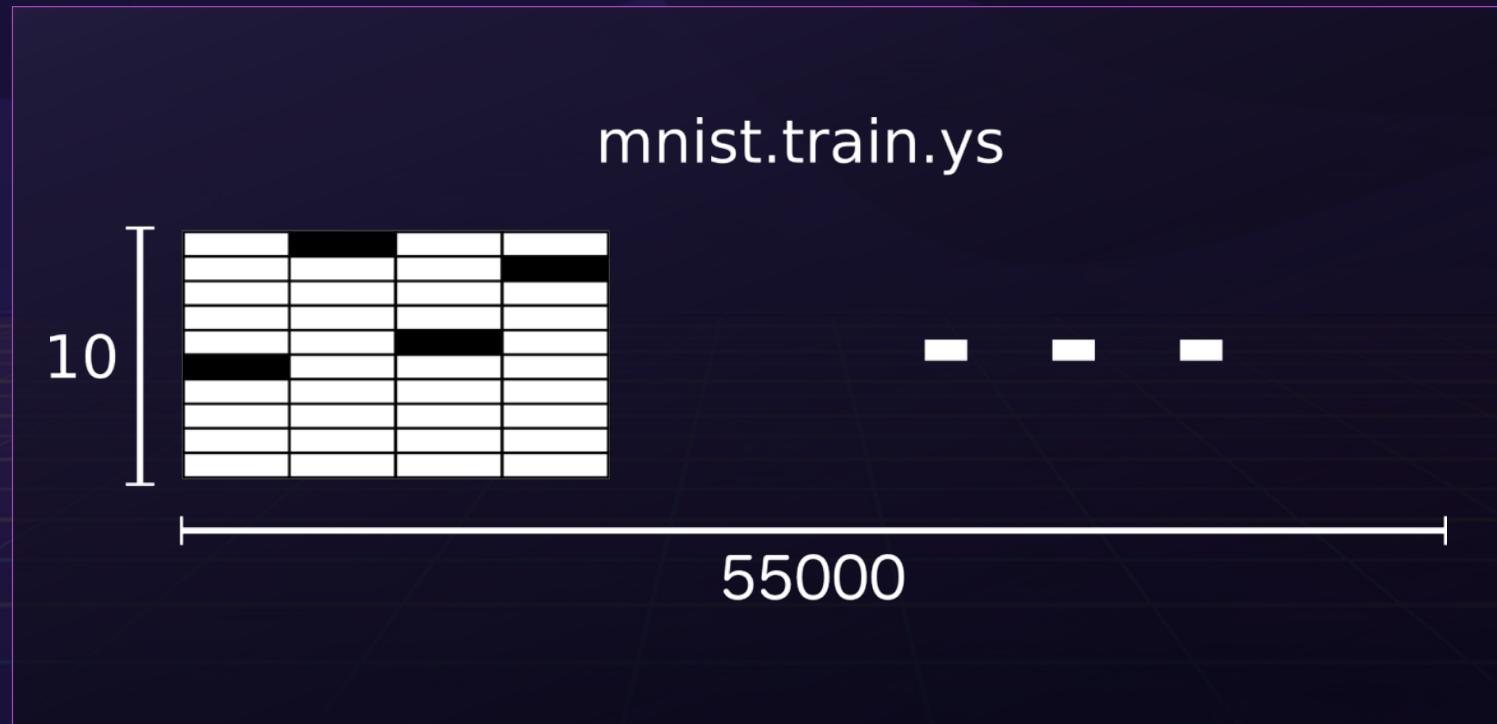
- ◆ The MNIST training dataset may be a 55000×784 tensor, that is, a multidimensional array. The first dimension represents the index of an image, and the second dimension represents the index of any pixel in the image (the pixel value in the tensor is between 0 and 1).





MNIST Dataset (2)

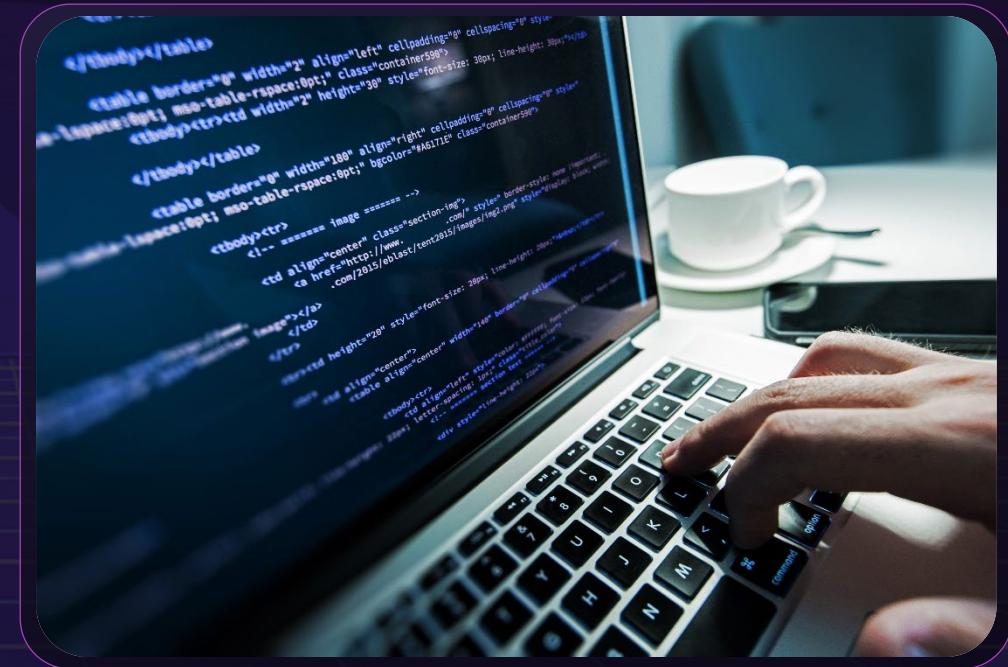
- ◆ `mnist.train.labels` is a two-dimensional array of 55000×10 , as is shown in the following figure:





Defining the Input Node

- ◆ TensorFlow has the following methods for defining input nodes:
 - Defined by placeholders (commonly used)
 - Defined by dictionary types (used when there are many inputs)
 - Directly defined (seldom used)
- ◆ The input images are 55000 x 784 matrices.
Therefore, create a placeholder x of [None, 784], and a placeholder y of [None and 10], and use the feed mechanism to input images and tags.





Defining Learning Parameters

- ◆ Define "learning parameter" variables.



Directly defined



Defined by dictionary

- ◆ The learning parameters include weight values and bias values. In TensorFlow, learning parameters are defined by variables.
- ◆ A variable represents a modifiable tensor, which is defined in the TensorFlow graph. Learning parameters defined by variables can be used for computing input values, or be modified in computation.



Defining the Operation

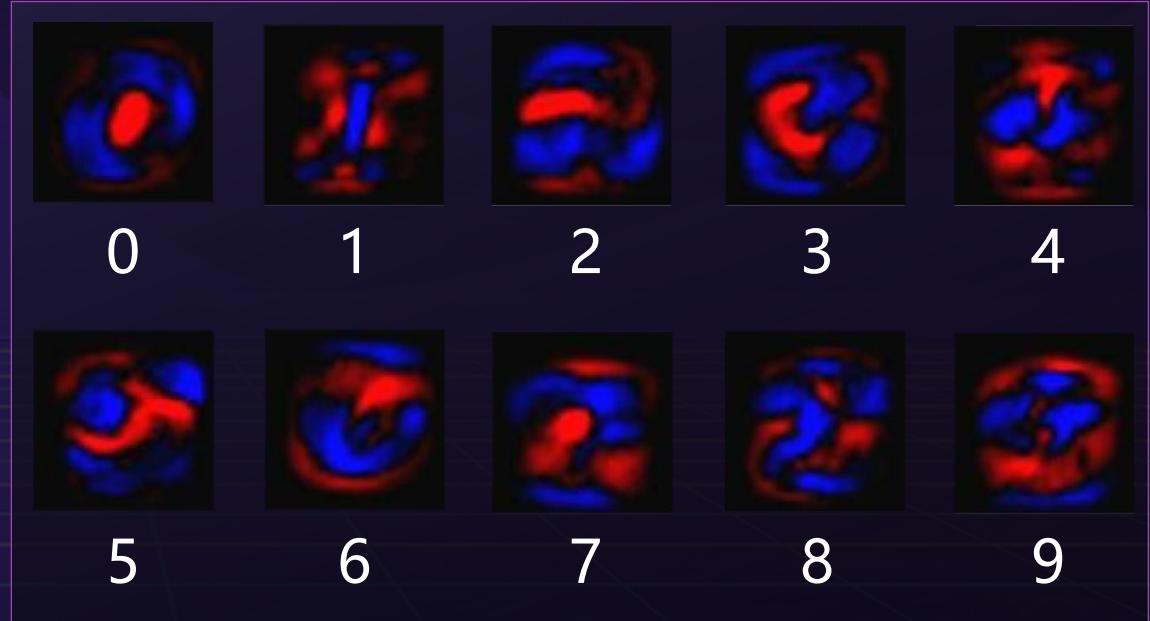
```
13     * NO interface Transformer<K, V> {
14     *
15     * Transforms from Entity to DTO.
16     *
17     * @param item entity to convert
18     * @return result of transformation DTO or <code>null</code> if the specified item is <code>null</code>
19     * @throws IllegalArgumentException if the specified object is <code>null</code>
20     */
21     V transformToDto(K item) throws IllegalArgumentException;
22
23 /**
24  * Transforms from DTO to Entity.
25  *
26  * @param item DTO to convert into Entity
27  * @return result of transformation - Entity or <code>null</code> if the specified item is <code>null</code>
28  * @throws IllegalArgumentException if the specified object is <code>null</code>
29  */
30 K transformFromDto(V item) throws IllegalArgumentException;
31
32 /**
33  * Transforms from Entity list to DTO list.
34  *
35  * @param items Entity list to convert
36  * @return result of transformation DTO list or empty list if the specified list was empty or <code>null</code>
37  * @throws IllegalArgumentException if the specified object is <code>null</code>
38  */
39 List<V> transformFromEntityList(List<K> items) throws IllegalArgumentException;
```

- ◆ The core process of creating a model determines the fitting effect of the model.
- ◆ Define the operation type by:
 - Defining the forward propagation model
 - Defining the loss function (calculate the error between the output value and the target value and work in collaboration with backpropagation). The common loss functions in TensorFlow are mean square error and cross entropy.
 - Defining the backpropagation structure



Creating a Model

- ◆ The softmax model can be used to assign probabilities to different objects. Even after we train more elaborate models, the final step also needs to use softmax to assign probabilities.
- ◆ The image below shows the weight of each pixel in a picture that the model learns for a particular number class. Red represents negative weights and blue represents positive weights.





Defining a Model

- ◆ We also need to add an extra bias because the input tends to have some extraneous interference. So for a given input image x , the evidence proving that it represents the digit i can be expressed as:

$$evidence_i = \sum_j W_{i,j}x_j + b_i .$$

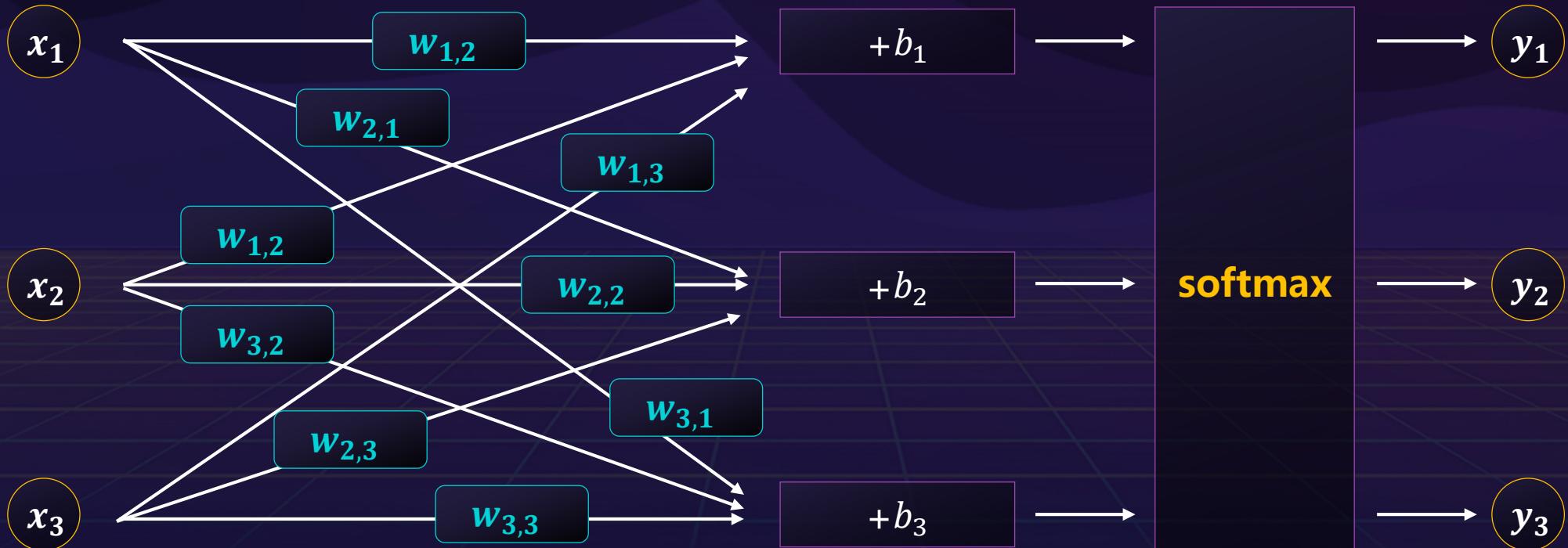
- ◆ Where W_i indicates weight. b_i indicates the bias of the numeric class i . j indicates the pixel index of a given image, x , and is used for summing the pixels. Then, use the softmax function to convert the evidence to the probability y .

$$y = softmax(evidence)$$



Regression Model (1)

- ◆ Add a bias to the weighted sum of the inputs x_i , and then input the biases to the softmax function.



Regression Model (2)

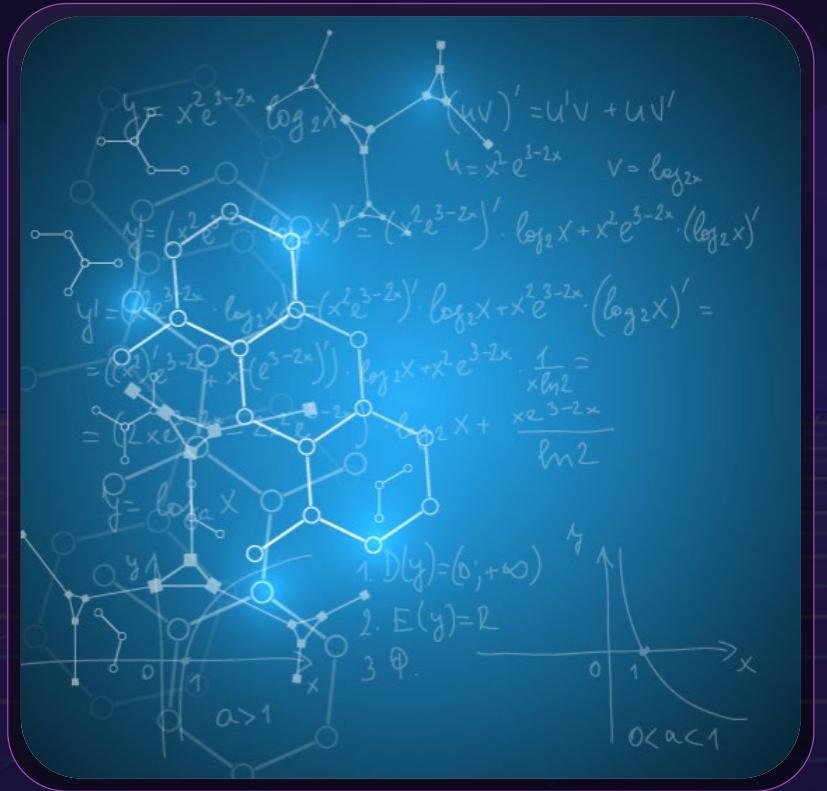
- In actual cases, the calculation process is expressed using vectors, as shown in the following figure:



Further, it can be written in a more compact way:

$$y = \text{softmax}(Wx + b).$$

Optimizing Functions and Objectives



- ◆ This process is completed in backpropagation. The backpropagation process is to transfer the error back along the reverse direction of forward propagation, involving L1 and L2 regularization, impulse adjustment, learning rate adaptation, and the Adam random gradient descent algorithm.

Initializing

- ◆ To create a model, you need to create a huge number of weights and biases. The weights in the model should be initialized with a small amount of noise to break the symmetry and avoid the zero gradient.
- ◆ To avoid repeated initialization during model creation, we define two functions for initialization.

```
def weight_variable(shape):  
    initial = tf.truncated_normal(shape, stddev=0.1)  
    return tf.Variable(initial)  
  
def bias_variable(shape):  
    initial = tf.constant(0.1, shape=shape)  
    return tf.Variable(initial)
```



Implementing the Model



- ◆ In order to train our model, we first need to define an indicator to evaluate this model is good. In fact, in machine learning, we usually define indicators to indicate that a model is bad, this indicator is called cost or loss, and then try to minimize this indicator. In this experiment. The cost function is the cross-entropy function.



Iterating and Training the Model to Obtain the Optimal Solution

- ◆ Here, we require TensorFlow to use a gradient descent algorithm to minimize the cross-entropy at a learning rate of 0.01. The gradient descent algorithm is a simple learning process. TensorFlow simply moves each variable little by little in a direction that keeps costs down.
- ◆ Then, start training the model, here we let the model cycle training 1000 times!

```
for i in range(1000):
    batch_xs, batch_ys = mnist.train.next_batch(100)
    sess.run(train_step, feed_dict={x: batch_xs, y_: batch_ys})
```

Testing the Model



- ◆ For `tf.argmax(y, 1)`, the return of the model prediction for any input x to the tag value, and `tf.argmax(y_,1)`, the representative of the correct label, we can use `tf.equal` to test whether the prediction is a true tag match (the same as the index position indicates a match).

```
correct_prediction = tf.equal(tf.argmax(y,1), tf.argmax(y_,1))
```



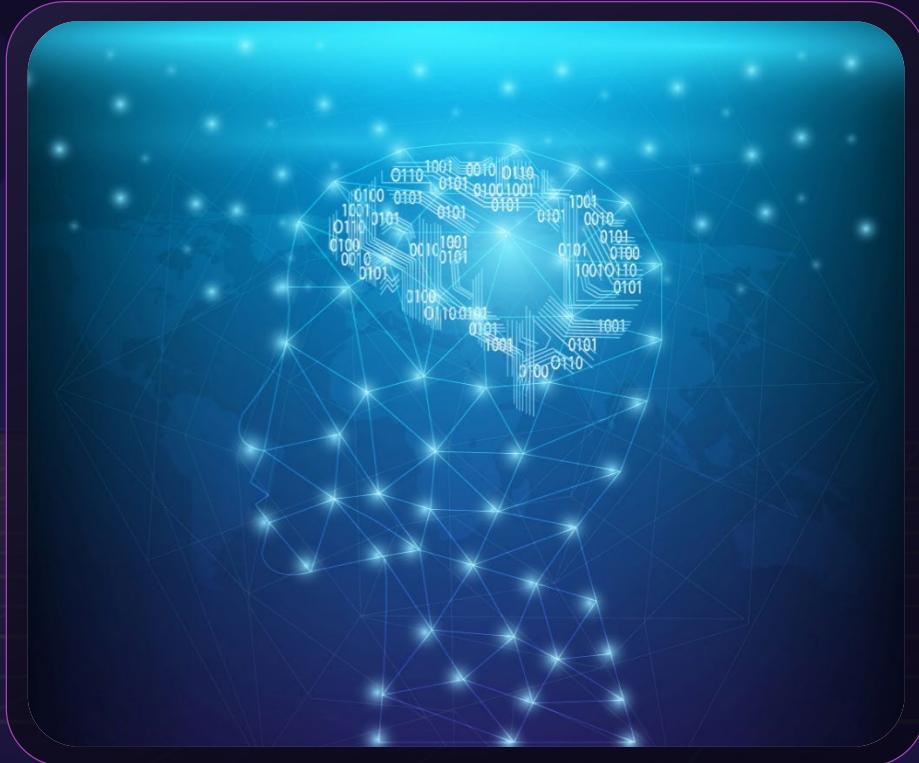
Evaluating the Model

- ◆ TensorFlow needs to add the model evaluation to the computation graph, and then call the model evaluation after model training.
- ◆ The model evaluation mainly involves the following indicators: average accuracy, identification time, and loss decrease.
- ◆ During model training, model evaluation helps gain insight into the model algorithms and provide prompt information to debug, improve, or modify the whole model. After training the model, the performance of the model needs to be evaluated quantitatively.





Using the Model (1)



- ◆ Save the model:
 - Create a saver and a path, and invoke the save command to save the parameters in the session.
- ◆ Use the model:
 - Replace the nodes of loss value with the output nodes. This is similar to what has been done in model testing.
- ◆ Read the model:
 - Read the model. Add images to the model for prediction, and display the images and their corresponding tags.

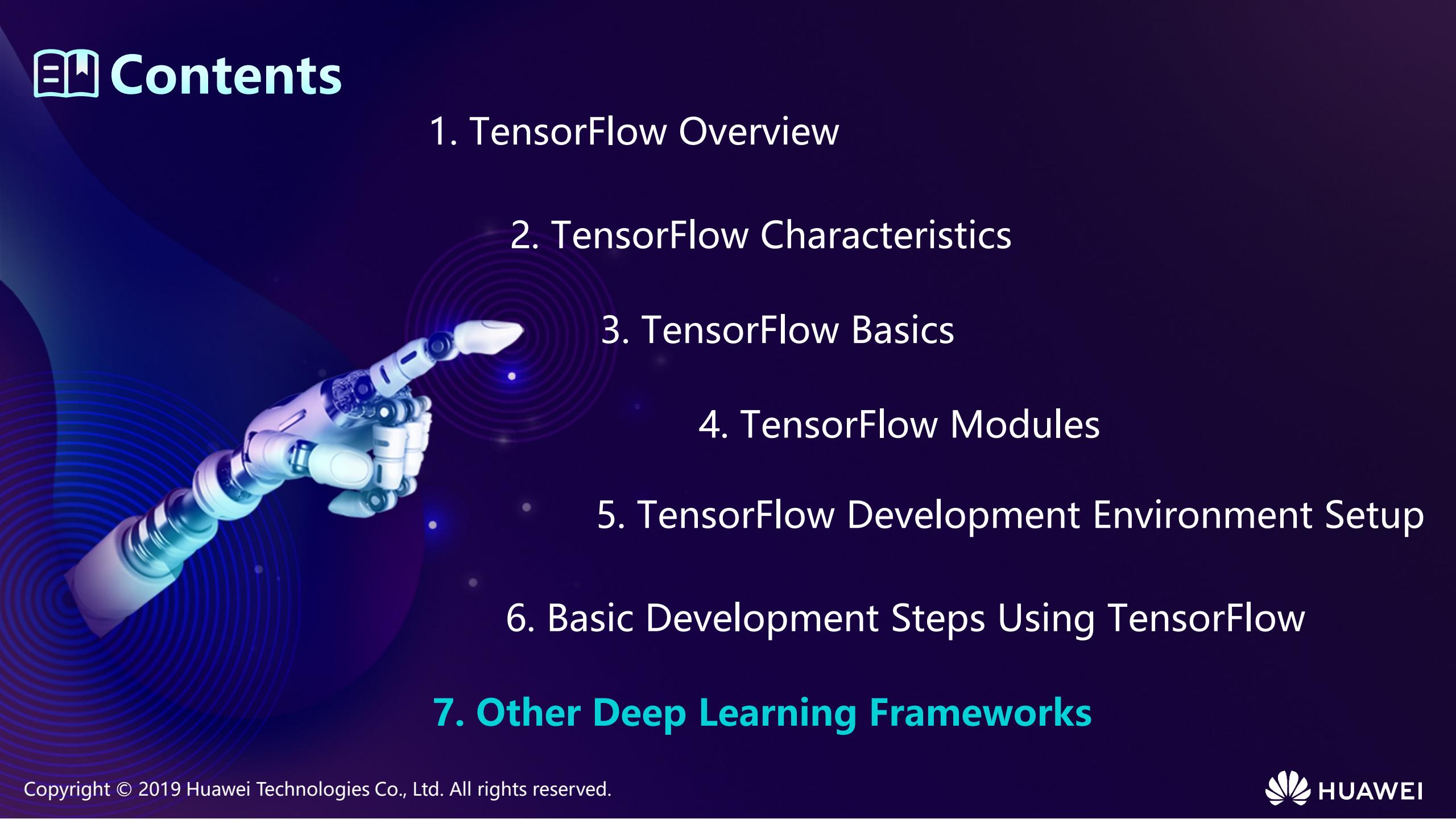


Using the Model (2)

- ◆ Read the model. Add two images to the model to predict results and display the labels corresponding to the two images.
- ◆ The definition of the network model remains unchanged during code execution. Create a new session. All operations are performed in the new session.

```
After 0 training step(s), validation accuracy using average model is 0.103
After 1000 training step(s), validation accuracy using average model is 0.9044
After 2000 training step(s), validation accuracy using average model is 0.9174
After 3000 training step(s), validation accuracy using average model is 0.9258
After 4000 training step(s), validation accuracy using average model is 0.93
After 5000 training step(s), validation accuracy using average model is 0.9346
After 6000 training step(s), validation accuracy using average model is 0.94
After 7000 training step(s), validation accuracy using average model is 0.9422
After 8000 training step(s), validation accuracy using average model is 0.9472
After 9000 training step(s), validation accuracy using average model is 0.9498
After 10000 training step(s), test accuracy using average model is 0.9475
```

Contents

- 
1. TensorFlow Overview
 2. TensorFlow Characteristics
 3. TensorFlow Basics
 4. TensorFlow Modules
 5. TensorFlow Development Environment Setup
 6. Basic Development Steps Using TensorFlow
 - 7. Other Deep Learning Frameworks**



Other Deep Learning Frameworks

The collage features the following logos:

- CNTK**: Microsoft logo with the text "CNTK".
- theano**: Theano logo.
- Caffe**: Caffe logo.
- mxnet**: mxnet logo with "dmlc" written above it.
- Keras**: Keras logo with a red square containing a white "K".
- DL4J**: DL4J logo with the text "Deep Learning for Java" below it.
- torch**: torch logo with a stylized molecular or neural network structure to its left.

A SCIENTIFIC COMPUTING FRAMEWORK FOR LUAJIT



Hardware Supported by Deep Learning Frameworks

Property	Caffe	Neon	TensorFlow	Theano	Torch
Core	C++	Python	C++	Python	Lua
CPU	√	√	√	√	√
Multi-threaded CPU	√ Blas	✗ Only date loader	√ Eigen	√ Blas, conv2D, limited OpenMP	√ Widely used
GPU	√	✓ customized Nvidia backend	√	√	√
Multi-GPU	√ (only data parallel)	√	✓ Most flexible	✓ Experimental version available	√
Nvidia cuDNN	√	✗	√	√	√
Quick deploy. On standard models	√ Easiest	√	√	✗ Via secondary libraries	√
Auto. gradient computation	√	✓ Supports Op-Tree	√	✓ Most flexible (also over loops)	√



Advantages and Disadvantages of the Popular Deep Learning Frameworks

◆ Caffe:

- Advantages: Easy to get started, fast, modular, open, and supported by a good community.
- Disadvantages: The layer needs to be defined in C++, and the model needs to be defined using Protobuf. The configuration file of the Caffe cannot be programmed to adjust the hyperparameter. Caffe only supports single-machine multi-GPU training, but does not natively support distributed training.

◆ Torch:

- Advantages: The modeling is simple and highly modular, with fast and efficient GPU support. It uses LuaJIT to connect to C++ and numerical optimization programs. It can be embedded in the interfaces of iOS, Android, and FPGA.
- Disadvantages: Some interfaces are not comprehensive, requiring LuaJIT to use the Lua language. Therefore, Torch lacks universality.

Conclusion

This chapter describes the concepts, characteristics, basic knowledge, and commonly used modules of TensorFlow. It then introduces how to set up the development environment and create TensorFlow programs. It also describes the basic development steps using TensorFlow and internal mechanisms of TensorFlow. In the end, it touches upon other deep learning frameworks.



1. Which of the following are application scenarios of TensorFlow? ()

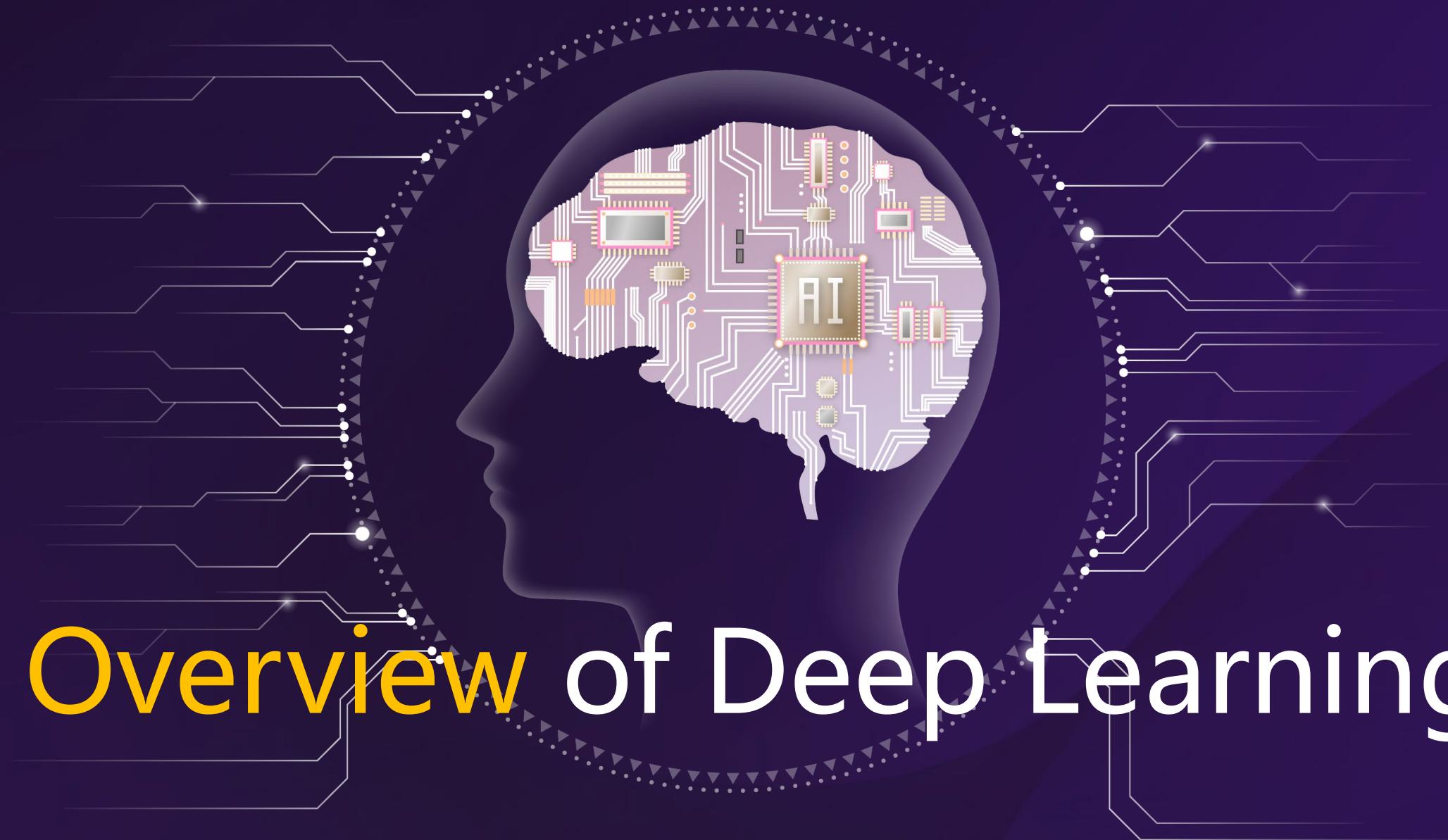
- A** AlphaGo
- B** Facial recognition
- C** Artistic style transfer
- D** Self-driving

Thanks

www.huawei.com



Overview of Deep Learning



Contents

1. Propaedeutics of Deep Learning

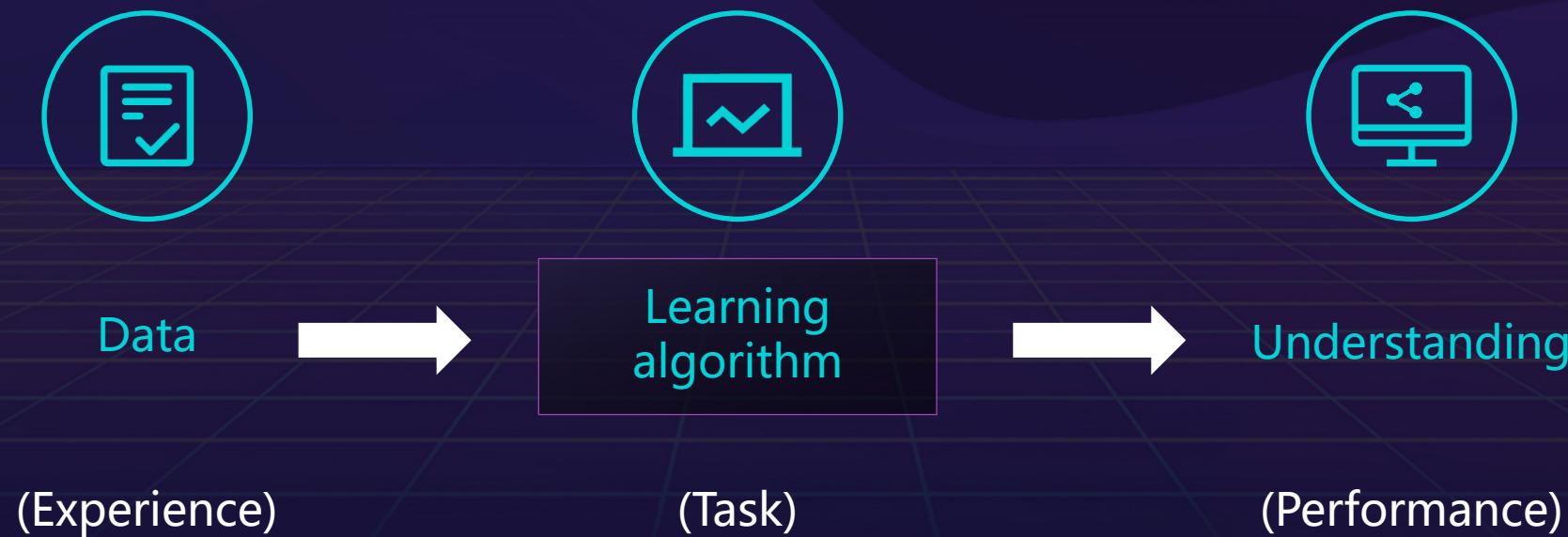
- Learning algorithms
- Common Machine Learning Algorithms
- Hyperparameter and Validation Set
- Parameter Estimation
- Maximum Likelihood Estimation
- Bayes Estimation

2. Overview of Deep Learning



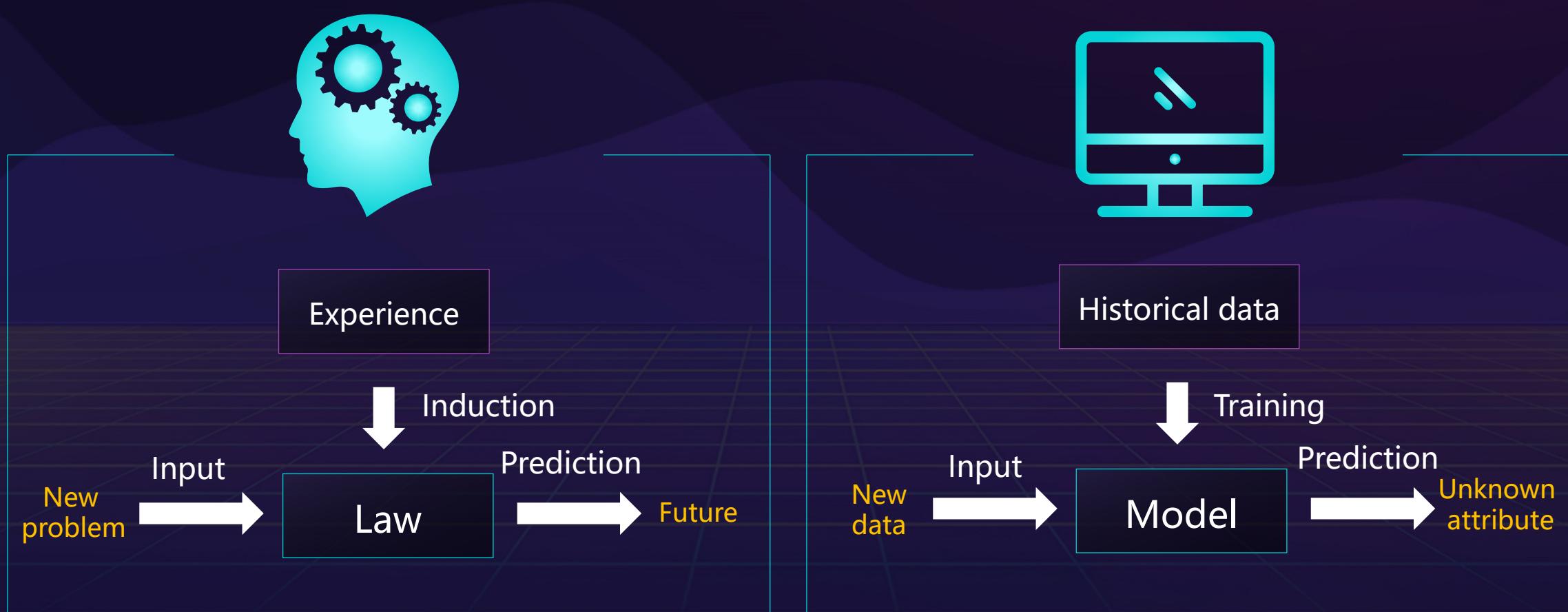
Learning Algorithms (1)

- ◆ Machine learning (including deep learning) is a study of learning algorithms. A computer program is said to learn from **experience E** with respect to some class of **tasks T** and **performance measure P** if its performance at **tasks in T** , as measured by P , improves with **experience E** .





Learning Algorithms (2)





Basic Terms and Concepts (1)



Dataset



Training set



Test set



Basic Terms and Concepts (2)

Generalization
capability

Error

classification

regression

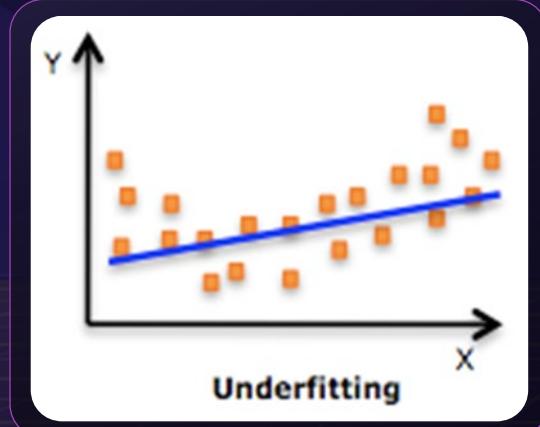
Training
error

Generalization
error

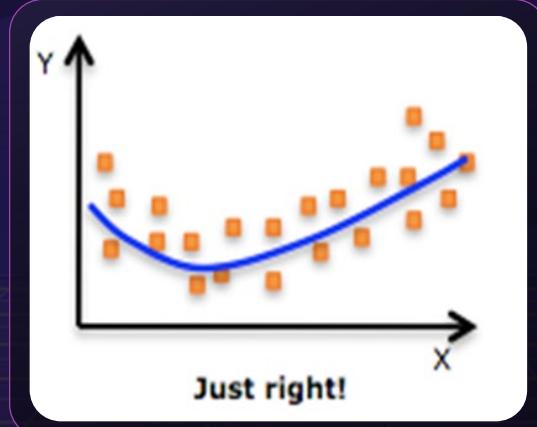


Basic Terms and Concepts (3)

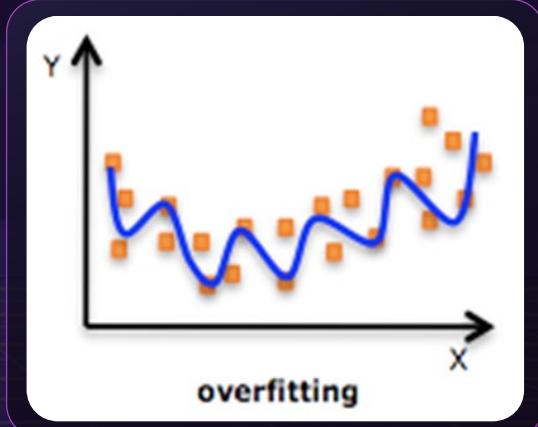
- ◆ **Capacity of a model:** refers to the ability to fit a wide variety of functions.



Failing to learn features



Perfect



Learning noise



Basic Terms and Concepts (5)

◆ Terms:

- P : positive examples, involving major interesting classes
- N : negative examples (other examples)
- TP : true positive examples (positive examples that are correctly classified by the classifier)
- TN : true negative examples (negative examples that are correctly classified by the classifier)
- FP : false positive examples (negative examples that are incorrectly marked as positive examples)
- FN : false negative examples (positive examples that are incorrectly marked as negative examples)

Predicted Observed	yes	no	Total
yes	TP	FN	P
no	FP	TN	N
Total	P'	N'	$P + N$

Confusion matrix

◆ **Confusion matrix:** a table (at least $m \times m$). $CM_{i,j}$, element of the first i -th line and j -th column is the number of examples known to be in class i but marked as j by the classifier.

- Ideally, for a highly accurate classifier, most data samples should be represented by the elements on the diagonal from $CM_{1,1}$ and $CM_{m,m}$. Other elements are 0 or close to 0. In other words, FP and FN are close to 0.

Basic Terms and Concepts (6)

Measure	Formula
Accuracy and correct classification rate	$\frac{TP + TN}{P + N}$
Error rate and false classification rate	$\frac{FP + FN}{P + N}$
Sensitivity, true positive rate, and <i>recall</i>	$\frac{TP}{P}$
Specificity and true negative rate	$\frac{TN}{N}$
Precision	$\frac{TP}{TP + FP}$
<i>F</i> score: harmonic mean of precision and recall	$\frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$
F_β (β is a non-negative real number)	$\frac{(1 + \beta^2) \times \text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}}$



Example of Machine Learning Performance Assessment

- ◆ We have trained a machine learning model to identify that whether the object in an image is a cat. Now we use 200 images to test its performance. Among 200 images, 170 contain cats and 30 do not contain cats. The identification result of the model is that 160 images contain cats and 40 do not contain cats.

$$\text{Precision: } P = \frac{TP}{TP+FP} = \frac{140}{140+20} = 87.5\%$$

$$\text{Recall: } R = \frac{TP}{P} = \frac{140}{170} = 93.3\%.$$

$$\text{Accuracy: } ACC = \frac{TP+TN}{P+N} = \frac{140+10}{170+30} = 85\%$$

		Predicted	yes	no	Total
Observed	yes	140	30	170	
	no	20	10	30	
Total		160	40	200	



Quiz

(Single choice) A machine learning model performs well on the training set, but performs badly on the testing set. What problem might be happened with this model? ()

- A. Underfitting
- B. Overfitting

Contents

1. Propaedeutics of Deep Learning

- Learning algorithms
- Common Machine Learning Algorithms
- Hyperparameter and Validation Set
- Parameter Estimation
- Maximum Likelihood Estimation
- Bayes Estimation

2. Overview of Deep Learning



Types of Machine Learning



Supervised
learning



Unsupervised
learning

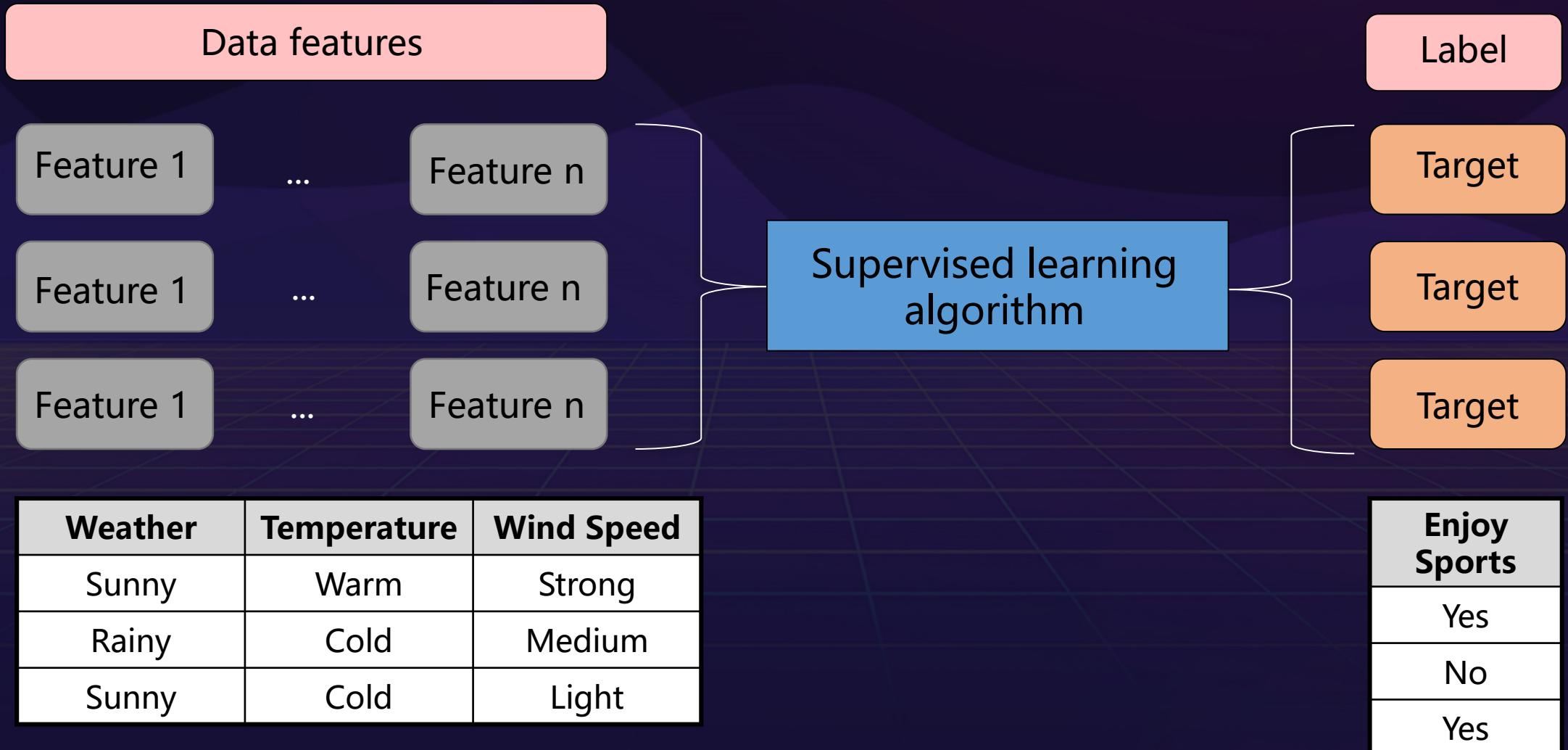


Semi-supervised
learning



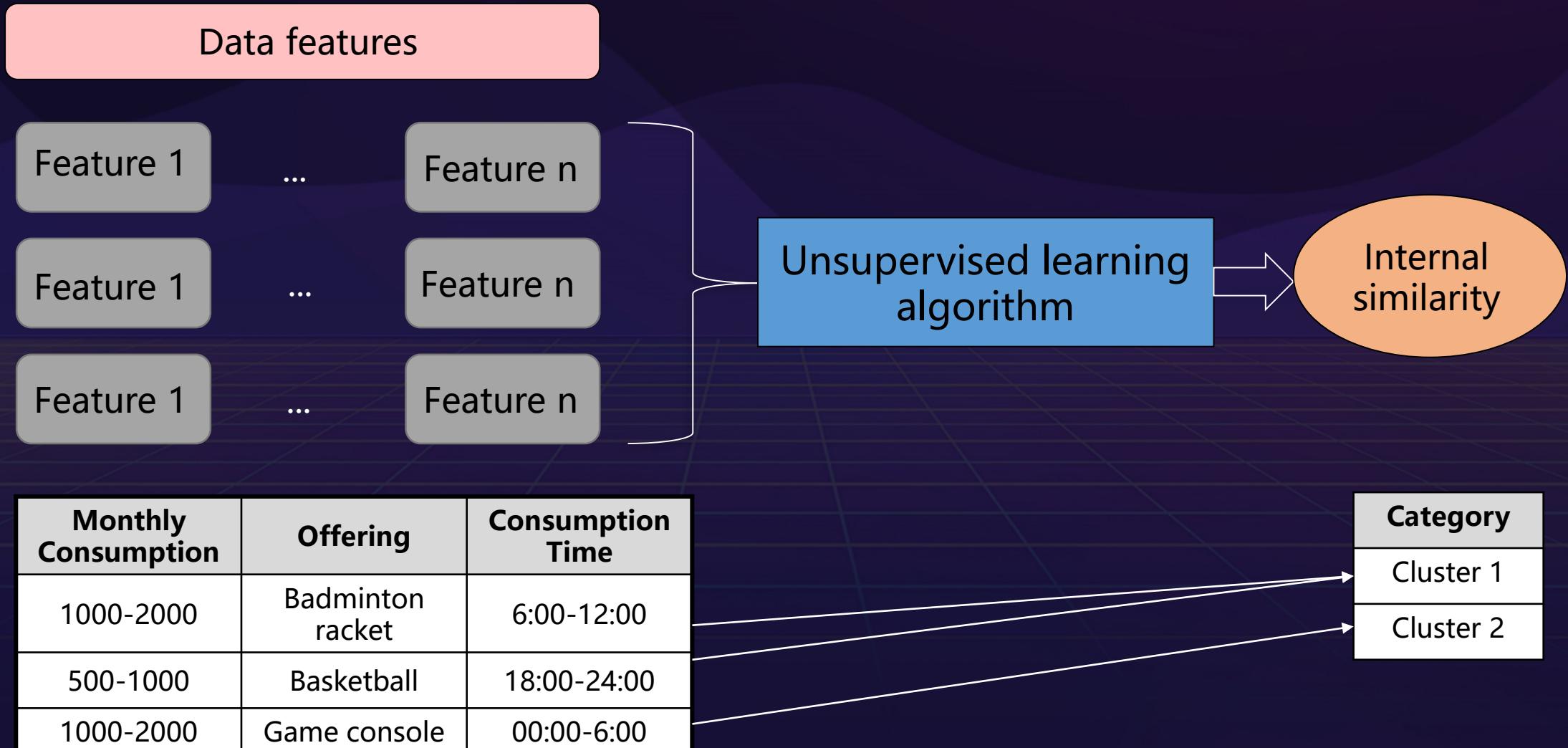
Reinforcement
learning

Supervised Learning

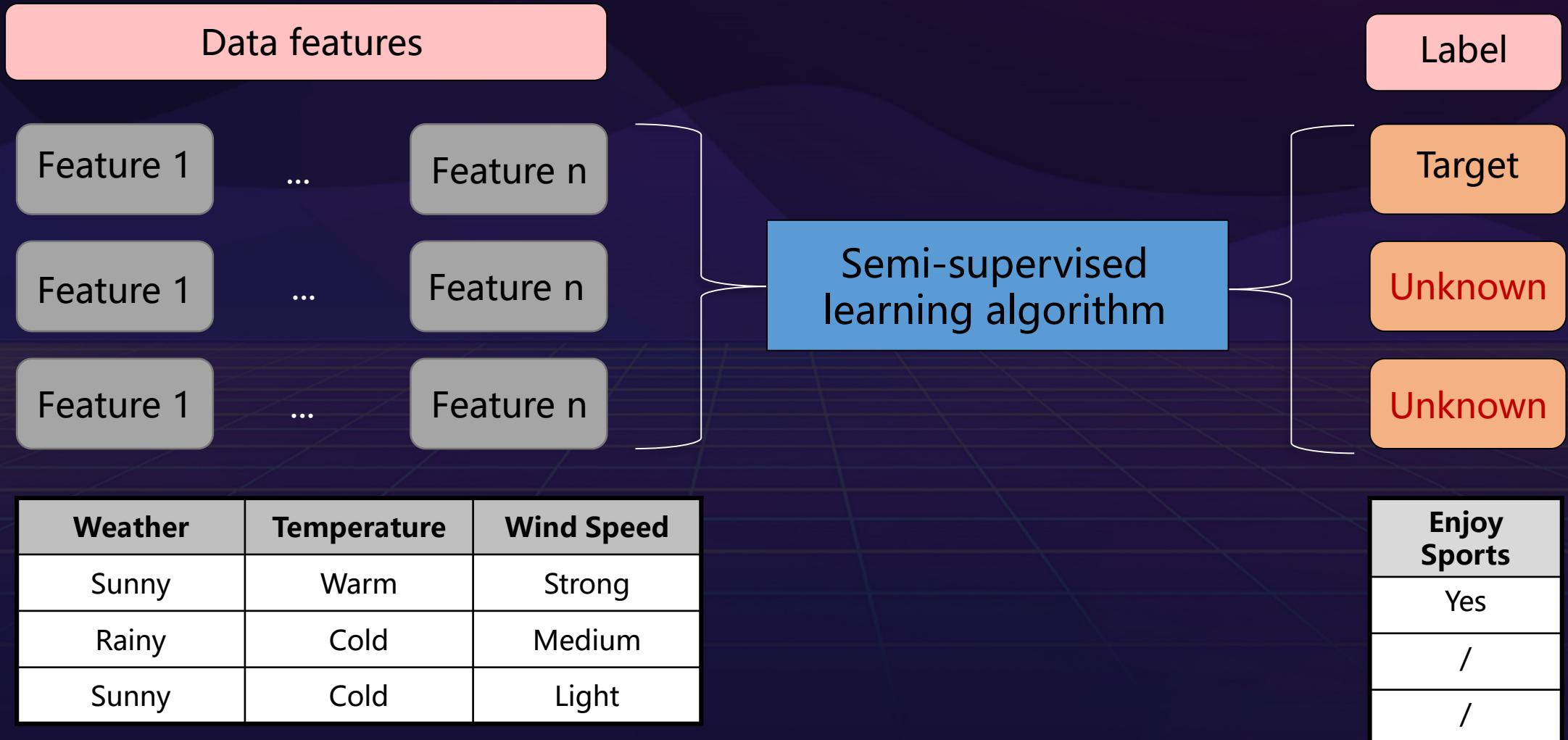




Unsupervised Learning



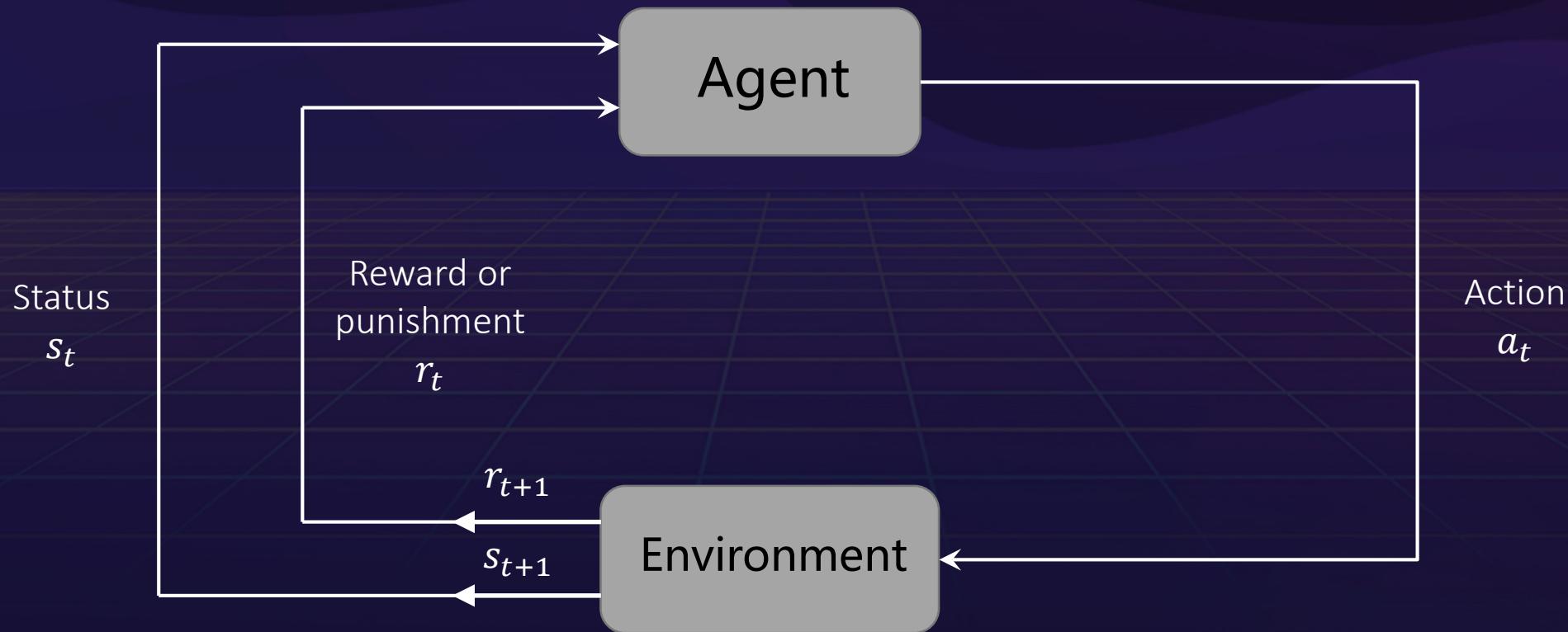
Semi-supervised Learning





Reinforcement Learning

The model perceives the environment, takes actions, and makes adjustments and choices based on the status and reward or punishment.



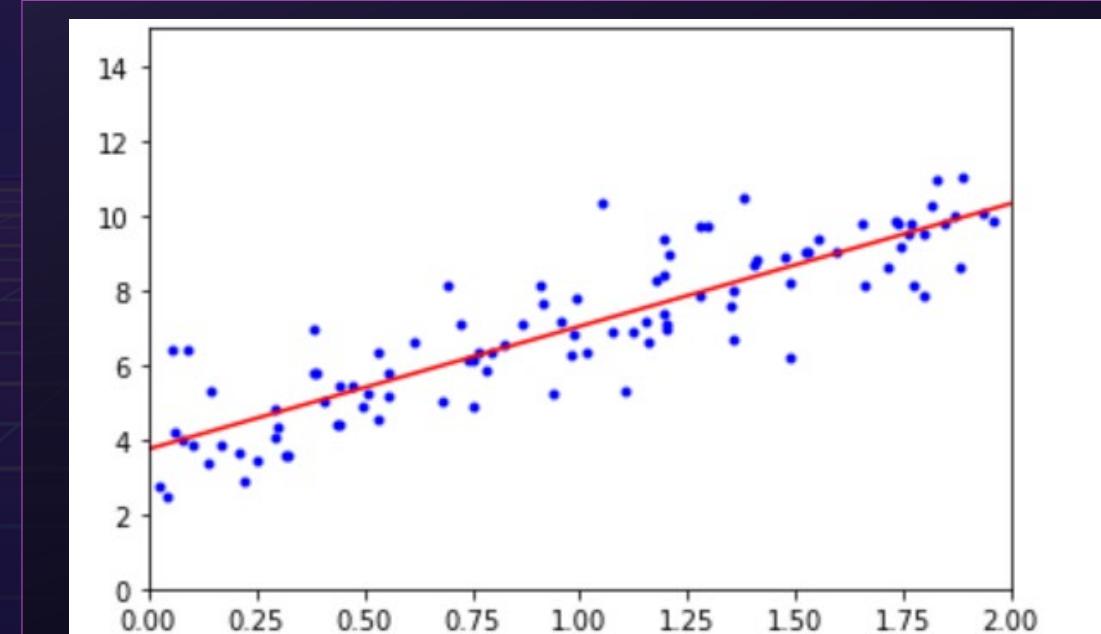


Quiz

1. (Single choice) What learning methods are most suitable for board games agent? ()
- A. Supervised learning
 - B. Unsupervised learning
 - C. Semi-supervised learning
 - D. Reinforcement learning

Linear Regression

- ◆ **Linear regression** : a statistical analysis method to determine the quantitative relationships between two or more variables through regression analysis in mathematical statistics.

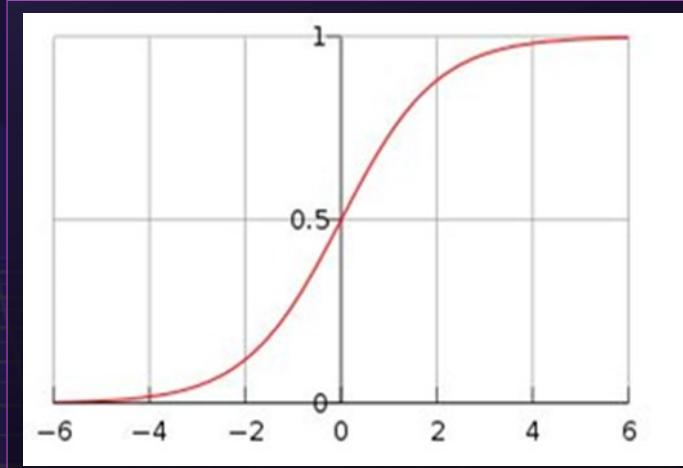


Logistic Regression

- ◆ **Logistic regression :** The logistic regression model is used to solve classification problems. Model definition:

$$P(Y = 1|x) = \frac{e^{wx+b}}{1 + e^{wx+b}}$$

$$P(Y = 0|x) = \frac{1}{1 + e^{wx+b}}$$

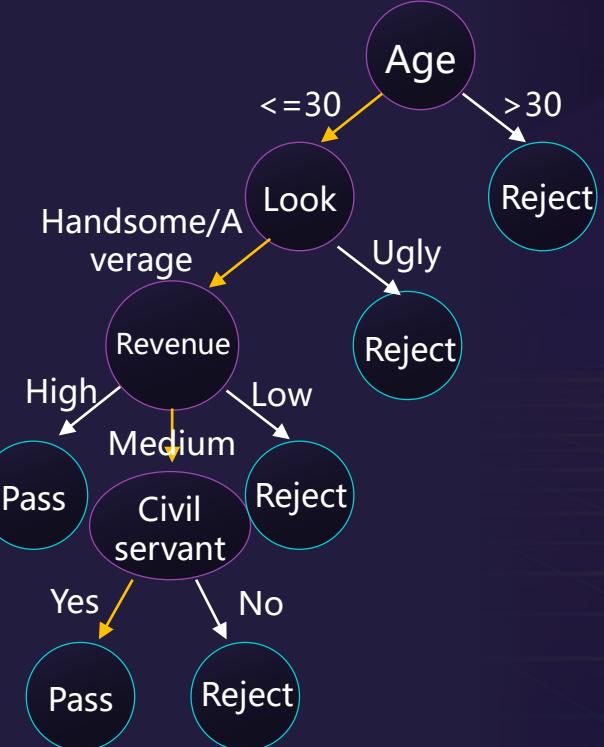


where w is the weight, b is the bias, $wx+b$ is the linear function of x . Compare the two probability values. x belongs to the class with a larger probability value.



Decision Tree

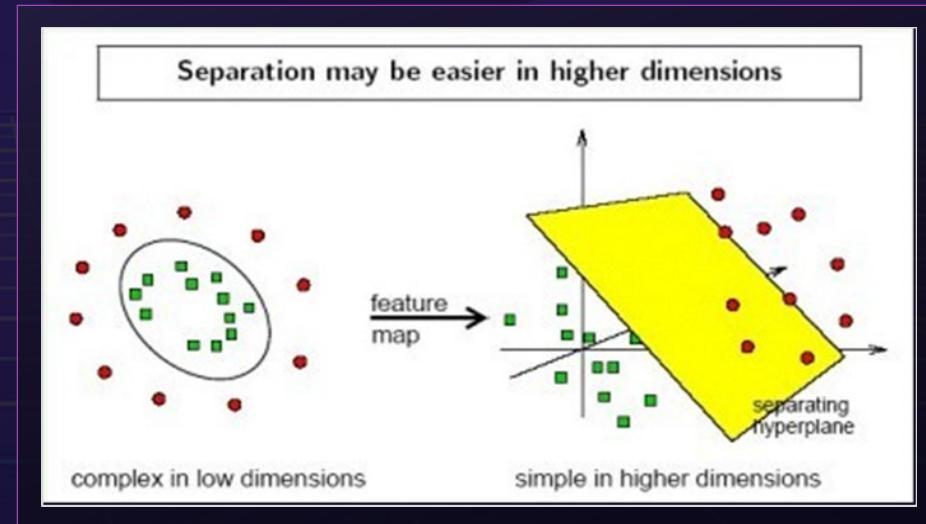
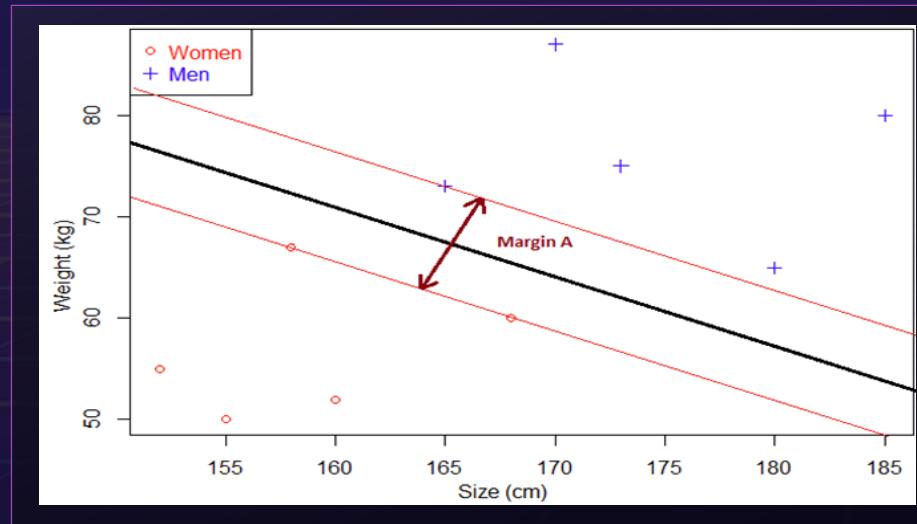
- ◆ **Decision tree:** A decision tree is a tree-like structure (a binary tree or a non-binary tree). Each non-leaf node represents a test on a feature attribute. Each branch represents the output of a feature attribute at a certain value range, and each leaf node stores a class. To use the decision tree, start from the root node, test the feature attributes of the items to be classified, select the output branches, and use the class stored on the leaf node as the final result.





Support Vector Machine

◆ A support vector machine (SVM) is a two-class classification model, and its basic model is a linear classifier defined in the feature space with the largest interval. SVM also includes kernel tricks that make it a non-linear classifier. The SVM learning algorithm is the optimum solution to convex quadratic programming.





Naive Bayes

◆ **Naive Bayes algorithm** : A classification method based on Bayes' theorem and attribute conditional independence assumption. According to Bayes' theorem, in a classification problem, for a given sample feature X , the probability that the sample belongs to the class H is:

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}$$

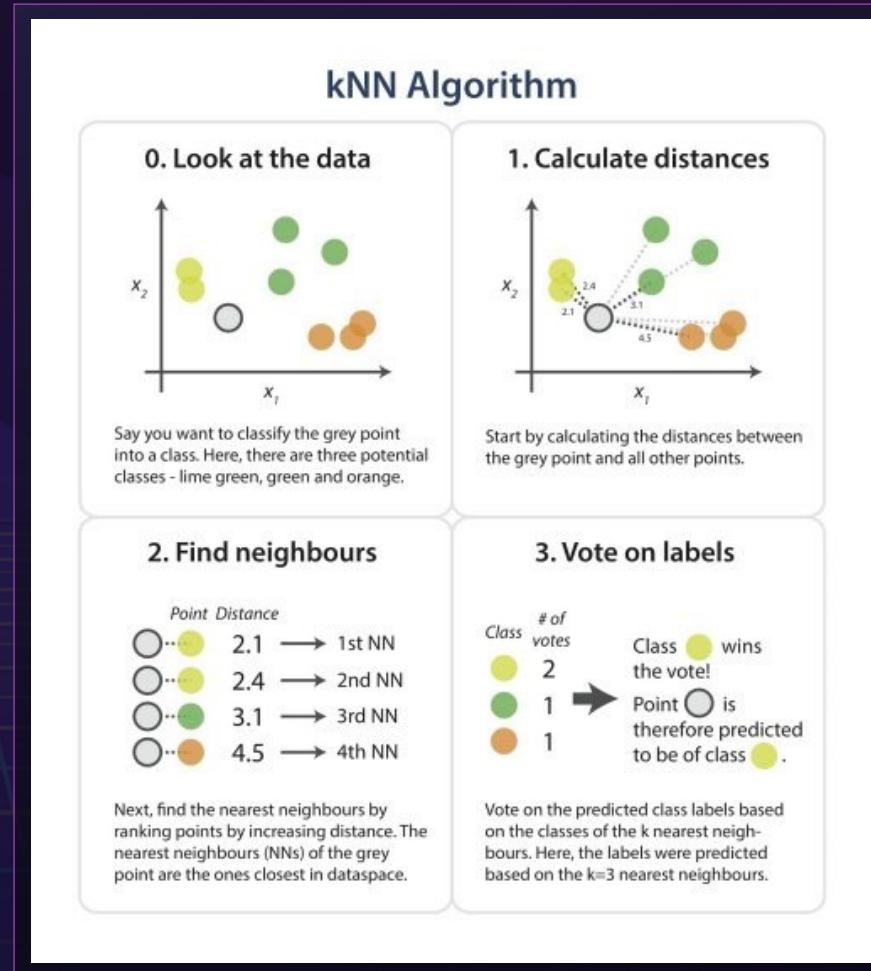
X is the data sample and generally described with the measurement value of n attribute sets. H is a hypothesis, for example, data sample X belonging to one certain class C . $P(H | X)$ is the posterior probability or H 's posterior probability under the condition X . $P(H)$ is the prior probability or H 's prior probability and is independent of X . $P(X)$ is the prior probability of X .

- Based on the naive assumption, namely attribute conditional independence assumption, and the law of total probability, the basic formula of Naive Bayes classification is as follows:

$$P(C_k|X) = \frac{P(X|C_k)P(C_k)}{P(X)} = \frac{\prod_{k=1}^M P(X_i|C_k)P(C_k)}{\sum_k P(C_k) \prod_{k=1}^M P(X_i|C_k)}$$

◆ The k-nearest neighbors (KNN)

classification algorithm is a theoretically mature method and one of the simplest machine learning algorithms. If the majority of k samples that are most similar to one sample (nearest neighbors in the feature space) belongs to one class, the sample also belongs to this class.





Quiz

1. Logistic regression model is used to deal with regression problem. ()

- A. True
- B. False

Contents

1. Propaedeutics of Deep Learning

- Learning algorithms
- Common Machine Learning Algorithms
- Hyperparameter and Validation Set
- Parameter Estimation
- Maximum Likelihood Estimation
- Bayes Estimation

2. Overview of Deep Learning



Hyperparameter (1)

There are two types of parameters in the learning model. One can be obtained from learning and the other cannot be obtained from data but is set based on human's experience. The latter is called hyperparameter.

◆ The features of model hyperparameter:

- It is often used in processes to help estimate model parameters.
- It is often specified by the practitioner.
- It can often be set using heuristics.
- It is often tuned for a given predictive modeling problem.

◆ Examples:

- Learning rate for training a neural network, number of iterations, batch size, activation function, number of neurons
- C and σ hyperparameters of support vector machines
- K in KNN

Hyperparameter (2)

◆ The procedure for searching for hyperparameters:

- Divide a dataset into a training set, verification set, and test set.
- Optimize the model parameters in the training set based on the performance indicators of the model.
- Search for the model hyperparameters in the training set based on the performance indicators of the model.
- Step 2 and step 3 are performed alternately. Finally, obtain parameters and hyperparameters of the model and assess the model in the test set.

◆ The common search algorithms:



Grid search



Random search



Heuristic intelligent search



Bayesian search

Validation Set

- ◆ **Cross validation :** It is a statistical analysis method used to validate the performance of a classifier. The basic idea is to divide the original dataset into two parts: training set and validation set. Train the classifier on the training set and test the model on the validation set. Then obtain a performance indicator of the classifier.
- ◆ **K-fold cross validation (K-CV):**
 - Partition the original training data set into k (equal) subsets.
 - Each time, one subset is used as a validation set and the remaining $k-1$ subsets as the training set. Then we obtain k models.
 - Use the mean classification accuracy of the validation set of k models as the performance indicator of the K-CV classifier.

Validation Set



- ◆ Note: The value **k** in **k-fold cross-validation** is also a hyperparameter.



Quiz

1. The hyper-parameters can not be changed in the model training. ()

- A. True
- B. False

Contents

1. Propaedeutics of Deep Learning

- Learning algorithms
- Common Machine Learning Algorithms
- Hyperparameter and Validation Set
- Parameter Estimation
 - Maximum Likelihood Estimation
 - Bayes Estimation

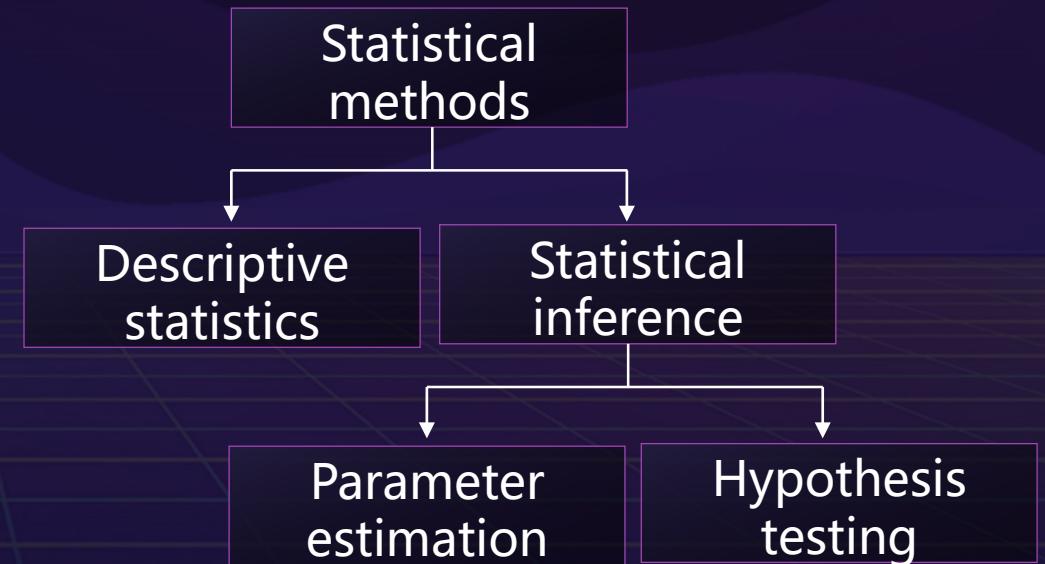
2. Overview of Deep Learning



Parameter Estimation

◆ **Parameter estimation** : Suppose that there is a statistical population and its distribution function is $F(x, \theta)$, where θ is an unknown parameter. Now we sample the population and obtain the sample X_1, X_2, \dots, X_n . We need to find an estimator $\hat{\theta}$ of the parameter θ or estimate θ 's function $g(\theta)$. This problem is called **parameter estimation**, including **point estimation** and **interval estimation**.

◆ **The position of parameter estimation in statistical methods:**



Point Estimator

- ◆ In one area, the fetal weight is $X \sim N(\mu, \sigma^2)$ (μ and σ^2 are unknown). Randomly select 100 babies and obtain their weight data

10,7,5,6,6,5,2, ...

We have 100 figures and how do we estimate μ and σ ?

- ◆ **Point estimator** : We need to construct a function of an appropriate sample: $T(X_1, X_2, \dots, X_n)$.

Once we have a sample, we can input the value into the function to obtain a result which can be used as the estimator of μ . $T(X_1, X_2, \dots, X_n)$ is the parameter's point estimator.

- ◆ **Problems:** Which estimator can be used to estimate μ ? How do we asset the estimators?

➤ Mean of the samples

➤ Median of the samples

➤ Other statistics



Statistics Assessment Standard: Unbiased

- ◆ **Unbiased** : The bias of an estimator is defined as:

$$\text{bias}(\hat{\theta}) = E(\hat{\theta}) - \theta$$

An estimator $\hat{\theta}$ is said to be unbiased if $\text{bias}(\hat{\theta}) = 0$, which implies that $E(\hat{\theta}) = \theta$.

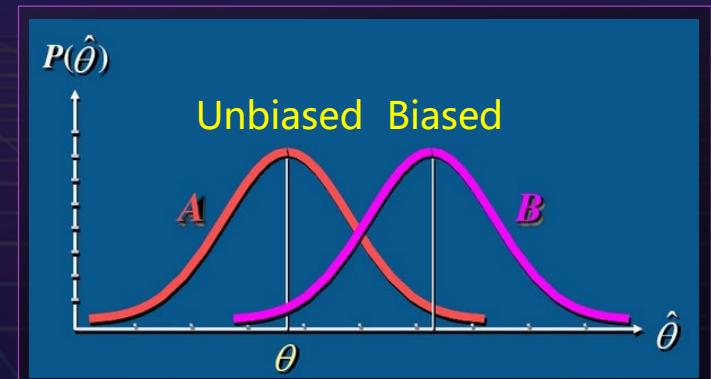
An estimator $\hat{\theta}$ is said to be asymptotically unbiased if $\lim \text{bias}(\hat{\theta}) = 0$, which implies that $\lim E(\hat{\theta}) = \theta$.

- ◆ Example: Suppose (X_1, X_2, \dots, X_n) is a sample from the population X .

$EX = \mu$, $DX = S^2$. Prove:

(1) $\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$ is an unbiased estimator of μ .

(2) $\sigma^2 = \frac{1}{n-1} (\sum_{i=1}^n X_i^2 - n\bar{X}^2)$ is an unbiased estimator of S^2 .



- ◆ Unbiased: samples are the all population.
- ◆ Biased: samples are part of the population.



Statistics Assessment Standard: Valid

◆ **Example:** X_1, X_2 , and X_3 is an sample of the population X . Are the following statistics the unbiased estimators of the mean μ ? Which one is optimal?

$$\widehat{\mu}_1 = \frac{2}{5}X_1 + \frac{1}{10}X_2 + \frac{1}{2}X_3, \quad \widehat{\mu}_2 = \frac{1}{3}X_1 + \frac{3}{4}X_2 - \frac{1}{12}X_3,$$

$$\widehat{\mu}_3 = \frac{1}{2}X_1 + \frac{1}{3}X_2 + \frac{1}{6}X_3, \quad \widehat{\mu}_4 = \frac{1}{5}X_1 + \frac{1}{10}X_2 + \frac{7}{10}X_3.$$

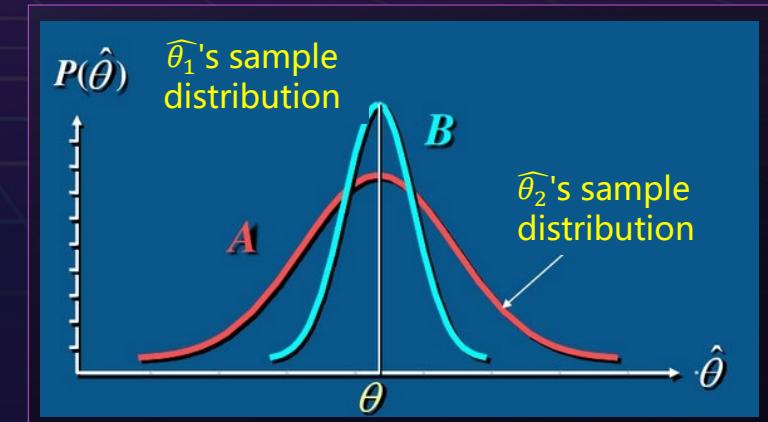
◆ **Valid:** Suppose $\widehat{\theta}_1$ and $\widehat{\theta}_2$ are two unbiased estimators of θ . If $D(\widehat{\theta}_1) < D(\widehat{\theta}_2)$, $\widehat{\theta}_1$ is more valid than $\widehat{\theta}_2$.

Proof: $E(\widehat{\mu}_1) = E\left(\frac{1}{n}\sum_{i=1}^n X_i\right) = \frac{1}{n} \times n\mu = \mu$.

$$E(\widehat{\mu}_2) = E\left(\frac{1}{k}\sum_{i=1}^k X_i\right) = \frac{1}{k} \times k\mu = \mu.$$

$$D(\widehat{\mu}_1) = D\left(\frac{1}{n}\sum_{i=1}^n X_i\right) = \frac{1}{n^2}\sum_{i=1}^n D(X_i) = \frac{\sigma^2}{n}, \quad D(\widehat{\mu}_2) = \frac{\sigma^2}{k}.$$

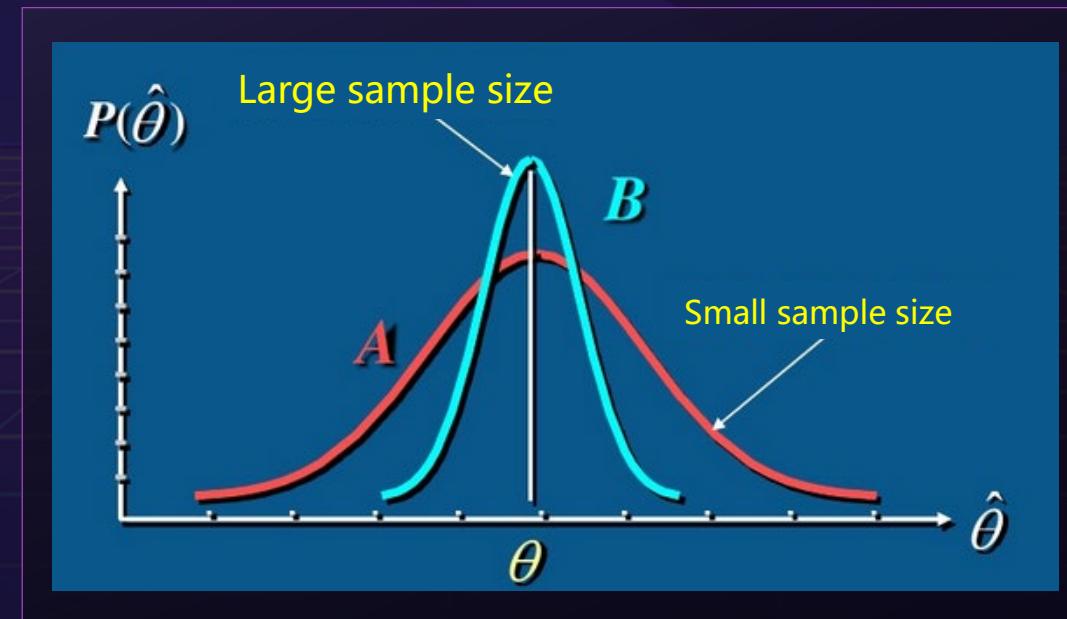
Since $n > k$, $D(\widehat{\mu}_1) < D(\widehat{\mu}_2)$ and $\widehat{\mu}_1$ is more valid.





Statistics Assessment Standard: Consistent

- ◆ **Consistent:** With probability P , $\hat{\theta} \rightarrow \theta(n \rightarrow \infty)$, i.e. for $\forall \epsilon > 0$, $\lim_{n \rightarrow \infty} P(|\hat{\theta} - \theta| > \epsilon) = 0$.
- ◆ **Example:** $EX = \mu$, $DX = \sigma^2$, X_1, X_2, \dots, X_n is a sample from X , $\hat{\mu}_1 = \frac{1}{n} \sum_{i=1}^n X_i$, $\hat{\mu}_2 = \frac{1}{k} \sum_{i=1}^k X_i$, $k < n$. Prove that $\hat{\mu}_1$ and $\hat{\mu}_2$ are unbiased estimators of μ and $\hat{\mu}_1$ is more valid than $\hat{\mu}_2$.





Quiz

- 1. We want to know the age distribution of the world's population. When we do a sample survey of the world's population, is the standard deviation biased or unbiased? ()**
- A. Biased**
 - B. Unbiased**

Contents

1. Propaedeutics of Deep Learning

- Learning algorithms
- Common Machine Learning Algorithms
- Hyperparameter and Validation Set
- Parameter Estimation
- Maximum Likelihood Estimation
- Bayes Estimation

2. Overview of Deep Learning

Maximum Likelihood Estimation (1)

- ◆ Suppose that an urn contains a number of black balls and a number of white balls. It is known that the ratio is 1:3. However, it is not known whether the urn has more black balls or more white balls. If sampling is performed for n times, the probability of the number of black balls as x is:

$$p(x; p) = \binom{n}{x} p^x q^{n-x}$$

where $q=1-p$. According to the assumption, $p=1/4$ or $3/4$. If $n=3$, x 's probability under p is:

x	0	1	2	3
$P(x, \frac{3}{4})$	$\frac{1}{64}$	$\frac{9}{64}$	$\frac{27}{64}$	$\frac{27}{64}$
$P(x, \frac{1}{4})$	$\frac{27}{64}$	$\frac{27}{64}$	$\frac{9}{64}$	$\frac{1}{64}$

$$\hat{p}(x) = \begin{cases} \frac{1}{4}, & x = 0, 1 \\ \frac{3}{4}, & x = 2, 3 \end{cases}$$



Maximum Likelihood Estimation (2)

- ◆ **Maximum likelihood estimation principle:** Suppose that X_1, X_2, \dots, X_n is a sample from the population X . Its joint density or joint distribution function is $f(x_1, x_2, \dots, x_n; \theta)$. Define the likelihood function as:

$$L(\theta) = f(x_1, x_2, \dots, x_n; \theta)$$

where x_1, x_2, \dots, x_n is the observed value of the sample. $L(\theta)$ is a function of parameter θ and can be used to assess the possibility of θ generating x_1, x_2, \dots, x_n .

- ◆ **Maximum likelihood estimation** uses $\hat{\theta}$ that maximizes $L(\theta)$ to estimate θ .

$$L(\hat{\theta}) = \max_{\theta} L(\theta)$$

$\hat{\theta}$ is the maximum likelihood estimate of θ . The statistic $\hat{\theta}(X_1, X_2, \dots, X_n)$ is the maximum likelihood estimator of θ .



Maximum Likelihood Estimation Example

- ◆ **Example:** Suppose that X_1, X_2, \dots, X_n is a sample from the population $X \sim B(1, p)$. Find the maximum likelihood estimator of the parameter p .

Solution:

Likelihood
function:

$$L(p) = f(x_1, x_2, \dots, x_n; p) = \prod_{i=1}^n p^{x_i} (1-p)^{1-x_i} = p^{\sum x_i} (1-p)^{n-\sum x_i}.$$

Log likelihood
function:

$$\ln L(p) = \sum_{i=1}^n x_i \ln(p) + (n - \sum_{i=1}^n x_i) \ln(1-p).$$

Derive the function with
respect to p and let it be 0:

$$\frac{d \ln L(p)}{dp} = \frac{1}{p} \sum_{i=1}^n x_i - \frac{1}{1-p} (n - \sum_{i=1}^n x_i) = 0.$$

$\hat{p} = \bar{x}$. Therefore the maximum
likelihood estimator of p is:

$$\hat{p}(X_1, X_2, \dots, X_n) = \frac{1}{n} \sum_{i=1}^n X_i = \bar{X}.$$



Quiz

- 1. Maximum likelihood estimation is one of the point estimation methods? ()**
- A. True**
 - B. False**

Contents

1. Propaedeutics of Deep Learning

- Learning algorithms
- Common Machine Learning Algorithms
- Hyperparameter and Validation Set
- Parameter Estimation
- Maximum Likelihood Estimation
- Bayes Estimation

2. Overview of Deep Learning

Bayes' Theorem

- ◆ Bayes' theorem calculates the posterior probability $P(h|D)$ with $P(D)$, $P(h)$, and $P(D|h)$. The formula is as follows:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

D is the data sample and generally described using measurement values of n attribute sets. h is a hypothesis, for example, data sample D belonging to one certain class C . $P(h|D)$ is the posterior probability or h 's posterior probability given that D is true. $P(h)$ is the prior probability or h 's prior probability and is independent of D . $P(D)$ is D 's prior probability.



Maximum a Posteriori Hypothesis

- ◆ $P(h|D)$ is the posterior probability of the hypothesis h . The hypothesis h that maximizes $P(h|D)$ is the maximum a posteriori (MAP) hypothesis. Formally, MAP supposes that h_{MAP} satisfies:

$$h_{MAP} = \operatorname{argmax}_{h \in H} P(h|D) = \operatorname{argmax}_{h \in H} \frac{P(D|h)P(h)}{P(D)} = \operatorname{argmax}_{h \in H} P(D|h)P(h)$$

- ◆ In practice, we usually cannot obtain the prior probability of each hypothesis. We only assume that all hypotheses in the hypothesis space are equally likely. $P(h)$ is a constant and $h_{MAP} = \operatorname{argmax}_{h \in H} P(D|h)P(h) = \operatorname{argmax}_{h \in H} P(D|h) = h_{ML}$ where $P(D | h)$ is the likelihood of data D given h . Therefore the hypothesis that maximizes $P(D|h)$ is the maximum likelihood hypothesis h_{ML} .
- ◆ If hypotheses in the hypothesis space are equally likely, $h_{MAP} = h_{ML}$.



Quiz

1. Maximum posterior estimation is one of the point estimation methods? ()

- A. True**
- B. False**

Challenges Motivating Deep Learning



Curse of dimensionality



Local constancy and
smoothness regularization



Manifold learning

Contents

1. Propaedeutics of Deep Learning

2. Deep Learning Overview

- Definition and Development of Neural Networks
 - Perceptron and Training Rules
 - Activation Functions
 - Types of Neural Networks
 - Regularization in Deep Learning
 - Optimizer
- Applications of Deep Learning

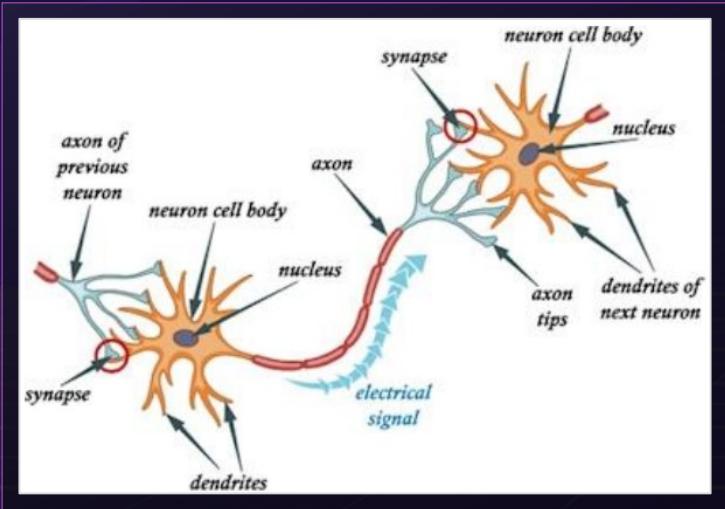


Neural Network Definition

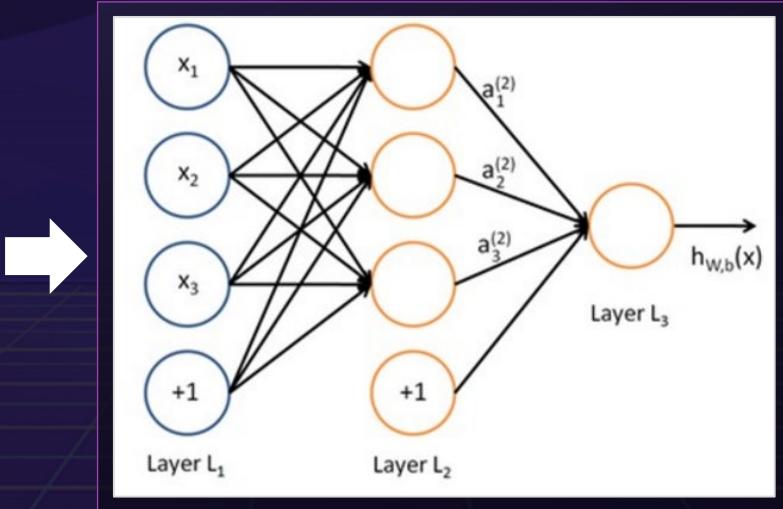
- ◆ Hecht Nielsen, a neural network researcher in the US, defines a neural network as "a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs."
- ◆ Based on the origin, features, and interpretations of the neural network, it can be simply defined as **an information processing system designed to simulate human brain's structure and functions.**
- ◆ **Artificial neural network (neural network for short):** refers to a network composed of artificial neurons. It abstracts and simplifies a human brain based on its microscopic structure and functions. It is an important way to simulate human intelligence and reflects some basic features of human brain functions, such as parallel information processing, learning, association, pattern classification, and memory.

Deep Learning

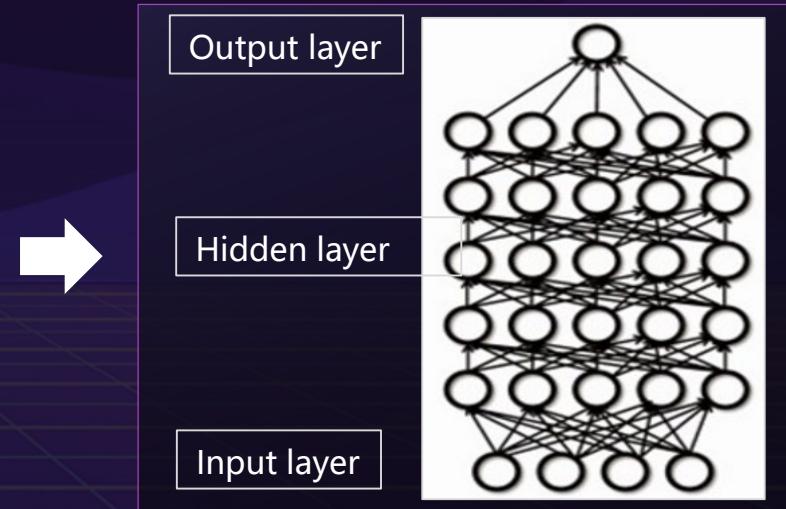
- ◆ Deep learning generally involves a deep neural network, where the depth refers to the number of layers of the neural network.



Artificial neural network



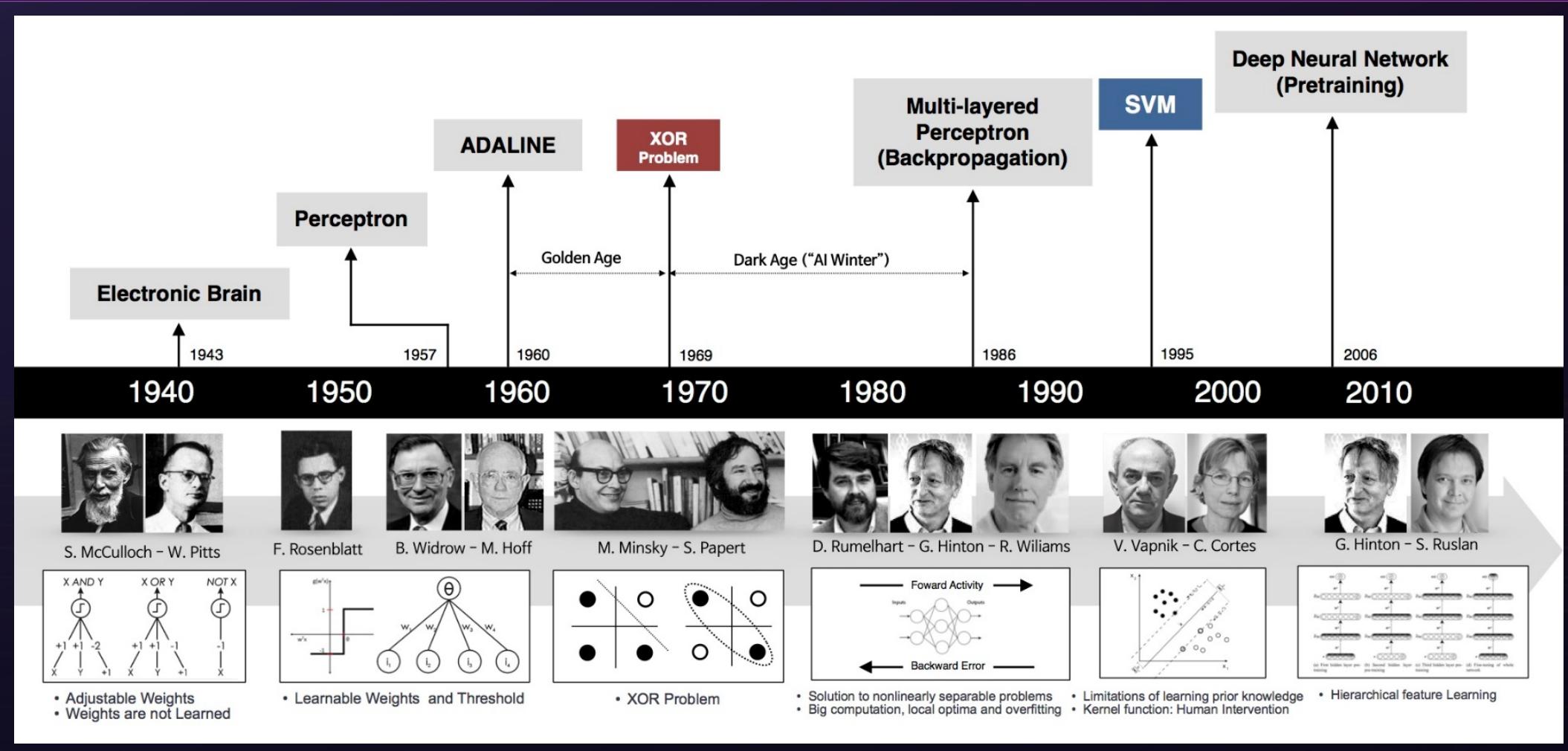
Perceptron



Deep neural network



Deep Learning Milestones





Quiz

1. Perceptron can solve the XOR problem. ()

- A. True
- B. False

Contents

1. Propaedeutics of Deep Learning

2. Deep Learning Overview

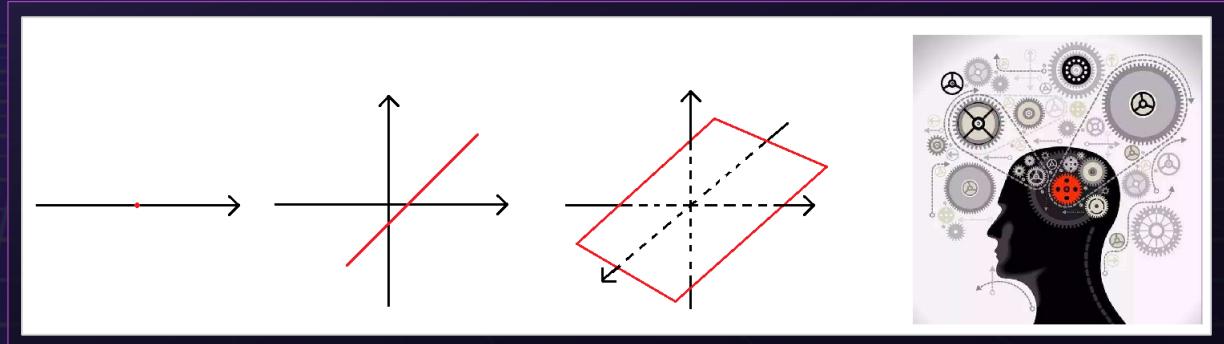
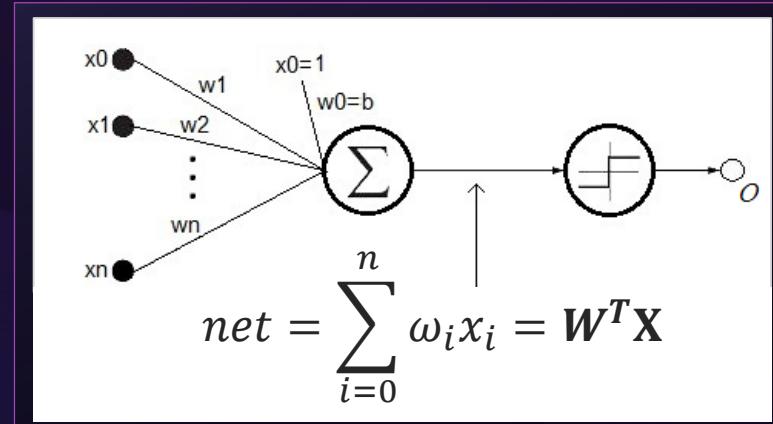
- Definition and Development of Neural Networks
 - Perceptron and Training Rules
 - Activation Functions
 - Types of Neural Networks
 - Regularization in Deep Learning
 - Optimizer
- Applications of Deep Learning

Perceptron

- ◆ **Input vector:** $X = [x_0, x_1, \dots, x_n]^T$.
- ◆ **Weight:** $W = [\omega_0, \omega_1, \dots, \omega_n]^T$, where ω_0 is bias.
- ◆ **Activation function:**

$$O = sign(net) = \begin{cases} 1, & net > 0, \\ -1, & otherwise. \end{cases}$$

- ◆ The perceptron is equivalent to a classifier. Its input is the high-dimensional vector X and it performs binary classification on input samples in the high-dimensional space.



Segmentation
Point
 $Ax + B = 0$

Segmentation
line
 $Ax + By + C = 0$

Segmentation
surface
 $Ax + By + Cz + D = 0$

Segmentation
hyperplane
 $W^T X + b = 0$



Perceptron Training Rules

- ◆ Perceptron training rules: For each training sample $\langle X, t \rangle$
 - Use the current weights to calculate the perceptron output o ;
 - Each weight is updated as follows :

$$\omega_i \leftarrow \omega_i + \Delta\omega_i$$

$$\Delta\omega_i = \eta[t - o]x_i$$

where X is the input vector, t is the target value, o is the output under the current weights, η is the learning rate, x_i and ω_i are the i -th elements of vectors X and W .

Gradient Descent and Loss Function

- ◆ For the multivariable function $o = f(x) = f(x_0, x_1, \dots, x_n)$, its gradient at $X' = [x_0', x_1', \dots, x_n']^T$:

$$\nabla f(x_0', x_1', \dots, x_n') = [\frac{\partial f}{\partial x_0}, \frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n}]^T|_{X=x'}$$

Direction of the gradient vector is the steepest ascent direction of the function. Therefore, the negative gradient vector $-\nabla f$ points to the steepest descent direction of the function.

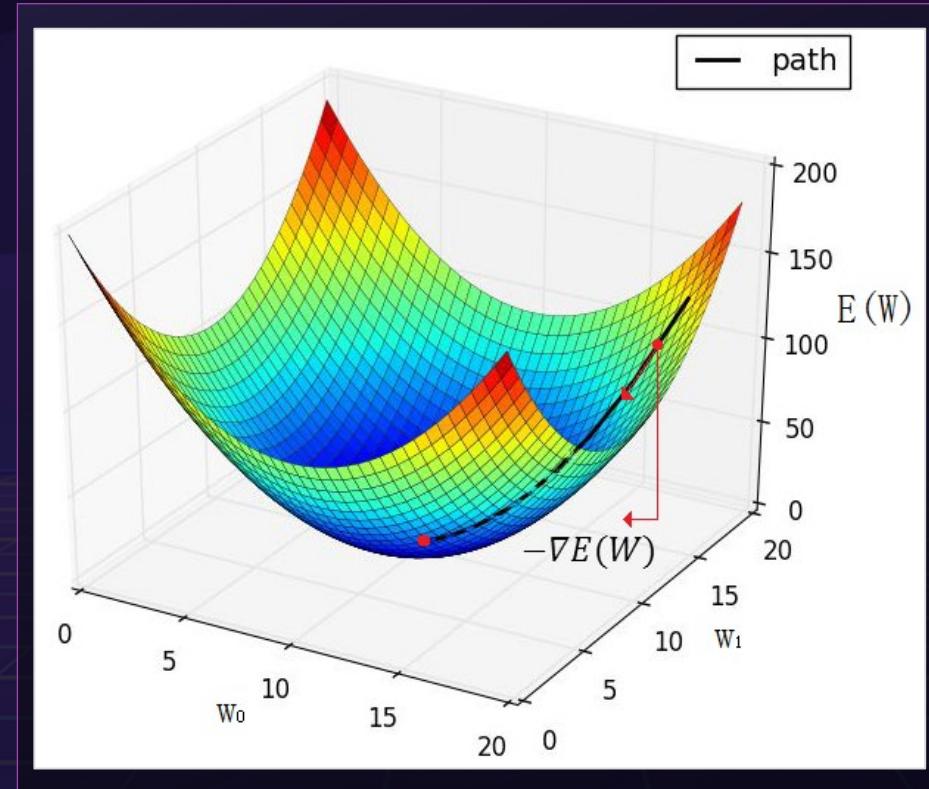
- ◆ When training samples are not linearly separable, we cannot find a hyperplane to enable the perceptron to perfectly classify training samples. However, we can classify them approximately and allow some minor classification errors. To minimize the errors, we need to first parameterize the error using the loss function (error function). The function reflects the error between the target output and the actual output of the perceptron. The most common error function is L2 loss function:

$$E(w) = \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2,$$

where d is the training example, D is the training example set, t_d is the target output, and o_d is the actual output.



Extrema of the Loss Function



Gradient descent on the
binary paraboloid

Global Gradient Descent Algorithm for Linear Units

- ◆ Each sample in the training sample set D is denoted as $\langle X, t \rangle$, X is the input vector, t is the target output, and η is the learning rate.
 - Initialize each w_i to a random value with a small absolute value.
 - Before the termination condition is met , do:
 - Initialize each Δw_i to 0.
 - For each $\langle X, t \rangle$ in D , do:
 - Enter X in the unit and calculate the output o .
 - For each w_i in the unit, do $\Delta w_i += \eta(t-o) x_i$
 - For each w_i in the unit, do $w_i += \Delta w_i$

SGD Algorithm and Online Learning

- ◆ To address the defects of the original gradient descent algorithm, a common variant, incremental gradient descent algorithm, is used, which is also called stochastic gradient descent (SGD) algorithm. One implementation is called online learning, which updates the gradient based on each sample:

$$\Delta w_i = \eta \sum_{d \in D} (t_d - o_d) x_{id} \Rightarrow \Delta w_i = \eta (t_d - o_d) x_{id}.$$

- ◆ ONLINE-GRADIENT-DESCENT(D, η)
 - Initialize each w_i to a random value with a small absolute value.
 - Before the termination condition is met, do:

For each $\langle X, t \rangle$ in D, do:

- Enter X in the unit and calculate the output o .
- For each w_i in the unit, do $\Delta w_i += \eta(t-o) x_i$.



Mini-Batch Gradient Descent

- Initialize each w_i to a random value with a small absolute value.
- Before the termination condition is met, do:
 - Initialize each Δw_i to 0.
 - Select samples (BS) from D . For each $\langle X, t \rangle$ among these samples, do:
 - Enter X in the unit and calculate the output o .
 - For each w_i in the unit, do $\Delta w_i += \eta(t-o) x_i$.
 - For each w_i in the unit, do $w_i += \Delta w_i$
 - If it is the last batch, disrupt the training sample sequence.



Quiz

1. Gradient Descent Algorithm can always get the global optimal solution. ()

- A. True
- B. False

Contents

1. Propaedeutics of Deep Learning

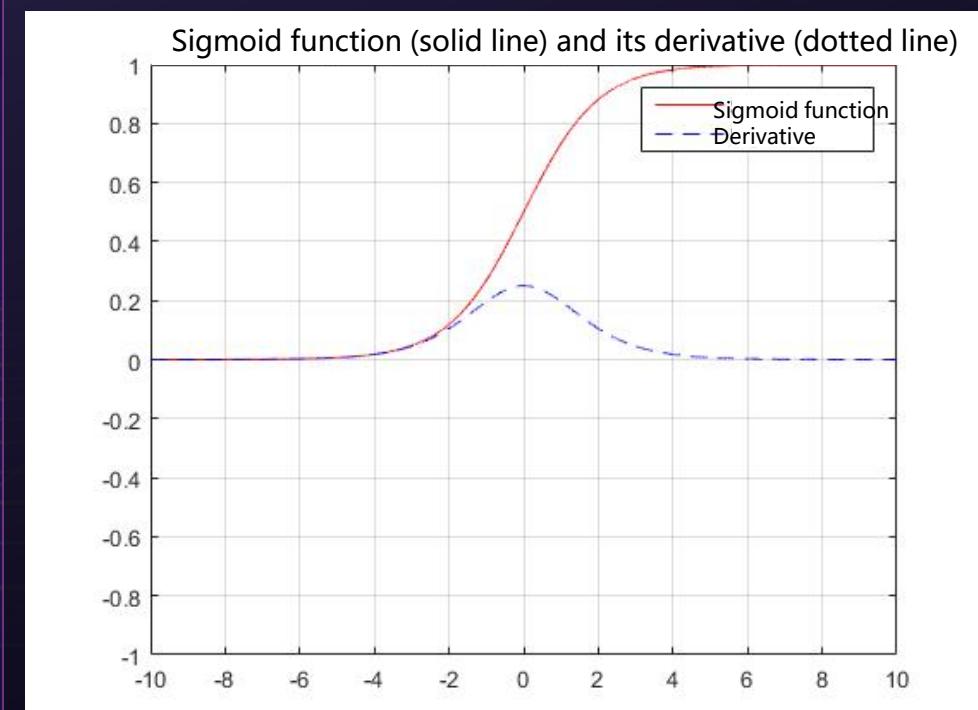
2. Deep Learning Overview

- Definition and Development of Neural Networks
 - Perceptron and Training Rules
 - Activation Functions
 - Types of Neural Networks
 - Regularization in Deep Learning
 - Optimizer
- Applications of Deep Learning



Sigmoid Function

$$f(x) = \frac{1}{1 + e^{-x}}$$

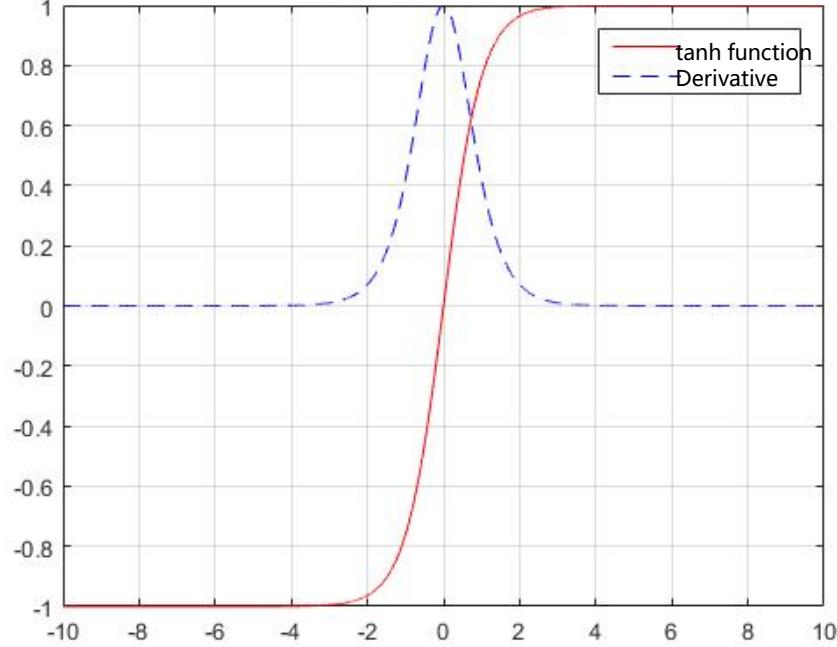




tanh function

$$\tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

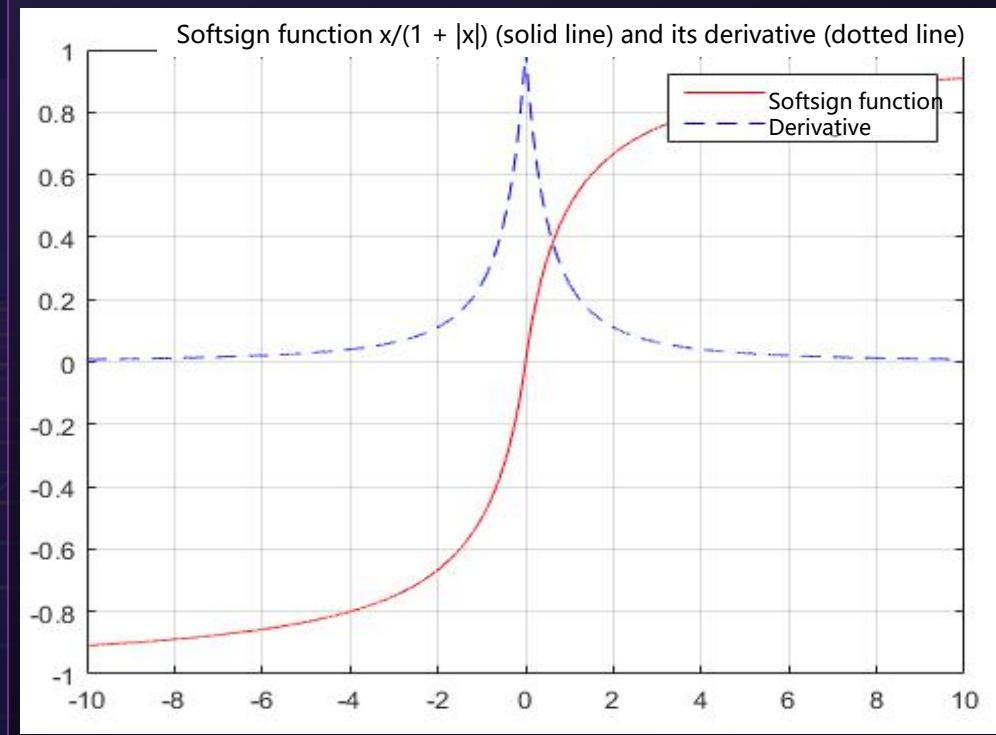
tanh function (solid line) and its derivative (dotted line)





Softsign Function

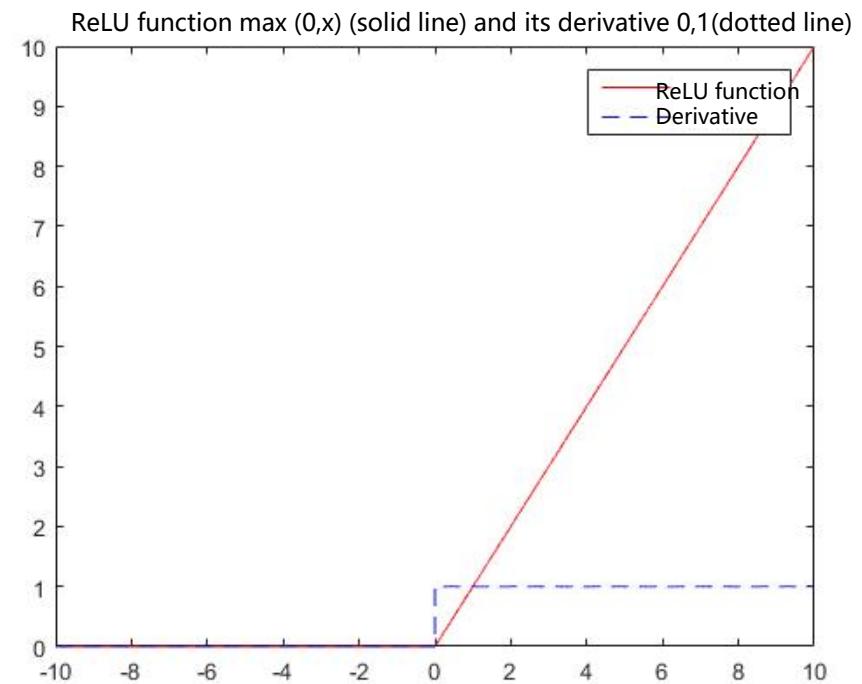
$$f(x) = \frac{x}{|x| + 1}$$





ReLU Function

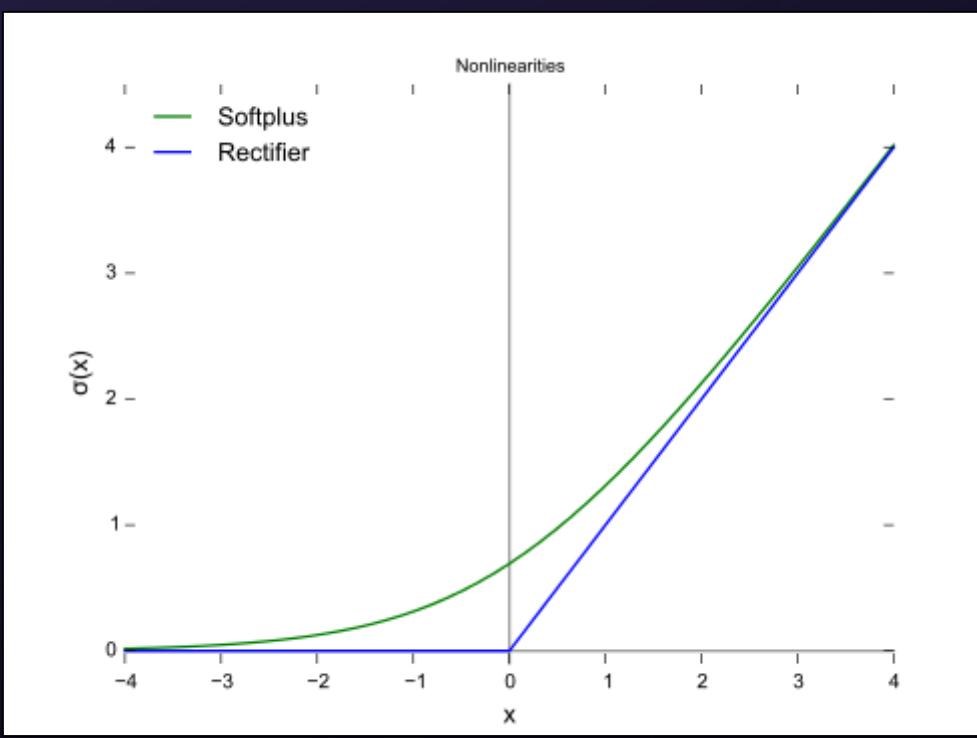
$$y = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$$





Softplus Function

$$f(x) = \ln(e^x + 1)$$



Factors

Approximating
identity near the origin

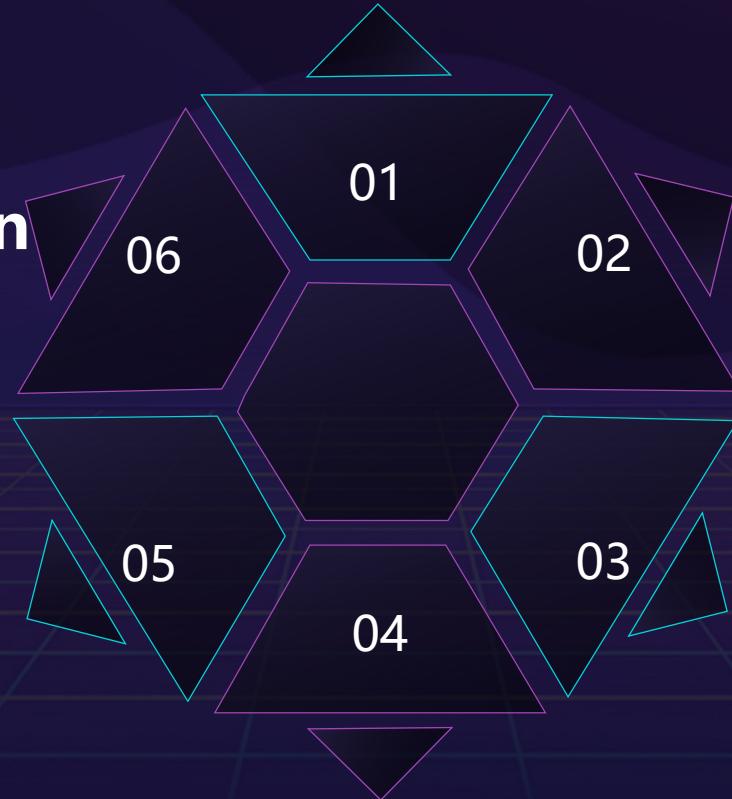
Non-linear

Smooth

Continuously
differentiable

Range

Monotonicity





Quiz

- 1. Comparing with the Sigmoid activation function, the ReLU activation function can mitigate the vanishing gradient problem. ()**
- A. True**
 - B. False**

Contents

1. Propaedeutics of Deep Learning

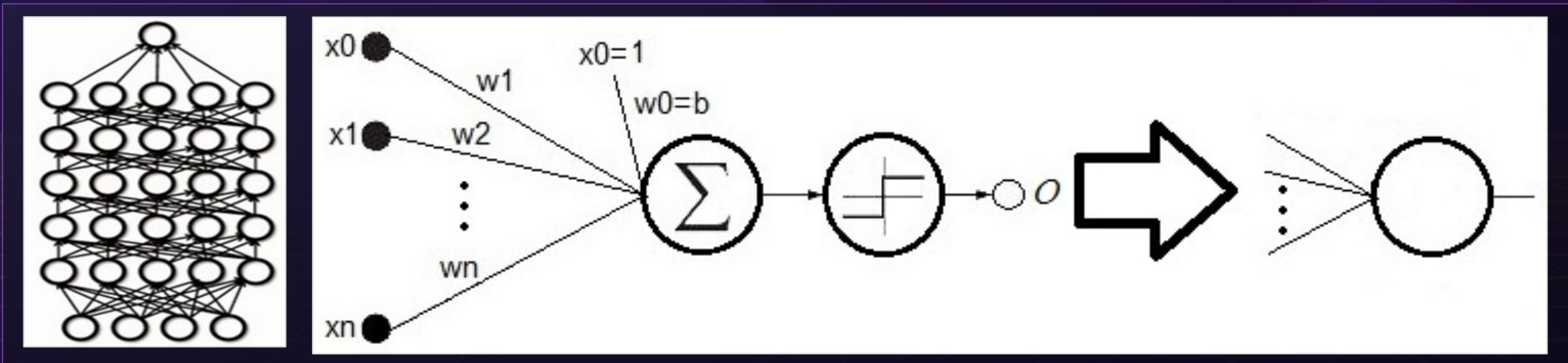
2. Deep Learning Overview

- Definition and Development of Neural Networks
 - Perceptron and Training Rules
 - Activation Functions
 - Types of Neural Networks
- Regularization in Deep Learning
- Optimizer
- Applications of Deep Learning



Multi-layer Fully Connected Artificial Neural Network

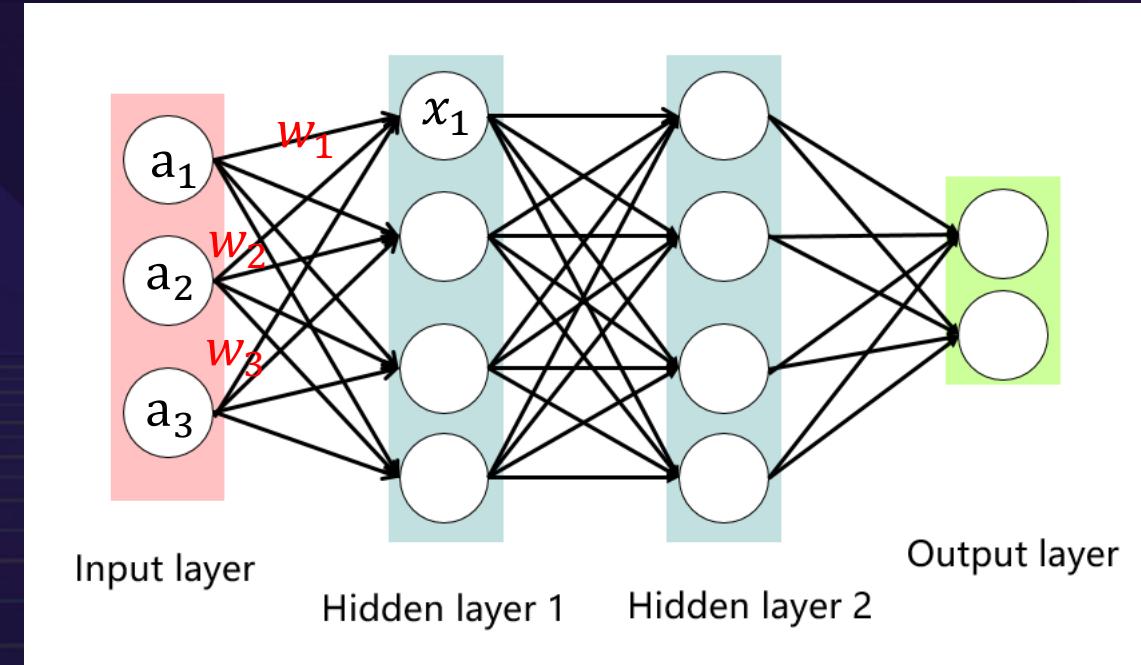
- ◆ A single perceptron has limited representation capability and can only represent the linear decision surface (hyperplane).





Feedforward Neural Network

- ◆ **The feedforward neural network** is one of the simplest neural networks in which neurons are arranged hierarchically. It is the most widely used neural network with the fastest development.



$$x_1 = a_1 \times w_1 + a_2 \times w_2 + a_3 \times w_3$$



Backpropagation Algorithm (1)

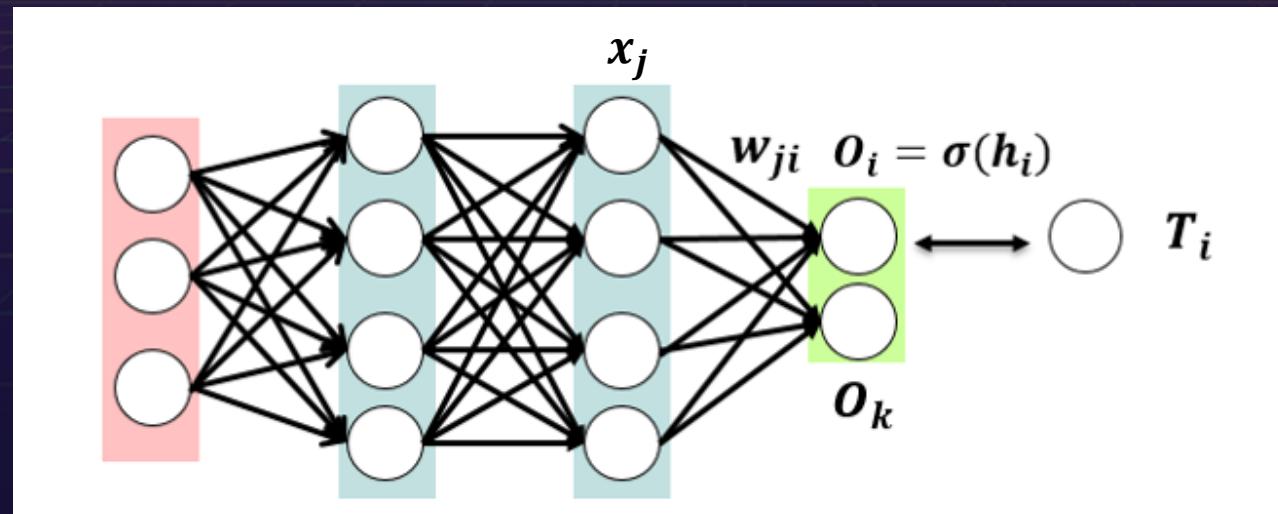
$$E = E_i + E_k$$

$$E_i = \frac{1}{2} (T_i - O_i)^2$$

$$O_i = \text{sigmoid}(h_i)$$

$$h_i = x_j \times w_{ji} + \text{L} + b$$

$$\begin{aligned}\Delta w_{ji} &= -\eta \frac{\partial E}{\partial w_{ji}} \\ &= -\eta \frac{\partial E}{\partial E_i} \frac{\partial E_i}{\partial O_i} \frac{\partial O_i}{\partial h_i} \frac{\partial h_i}{\partial w_{ji}} \\ &= -\eta \times 1 \times (T_i - O_i) \times [O_i \times (1 - O_i)] \times x_j\end{aligned}$$

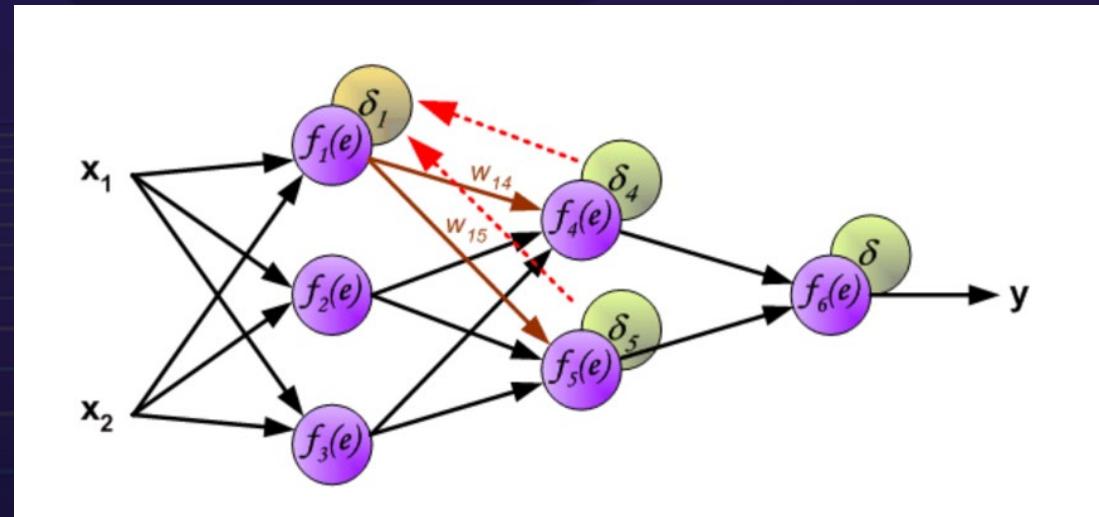




Backpropagation Algorithm (2)

The weights' coefficients w_{ji} used to propagate errors back are equal to this used during computing output value. Only the direction of data flow is changed (signals are propagated from output to inputs one after the other). This technique is used for all network layers. If propagated errors came from few neurons they are added.

$$\delta_1 = (\delta_4 \times w_{14} + \delta_5 \times w_{15}) \times f'_1(x_1)$$





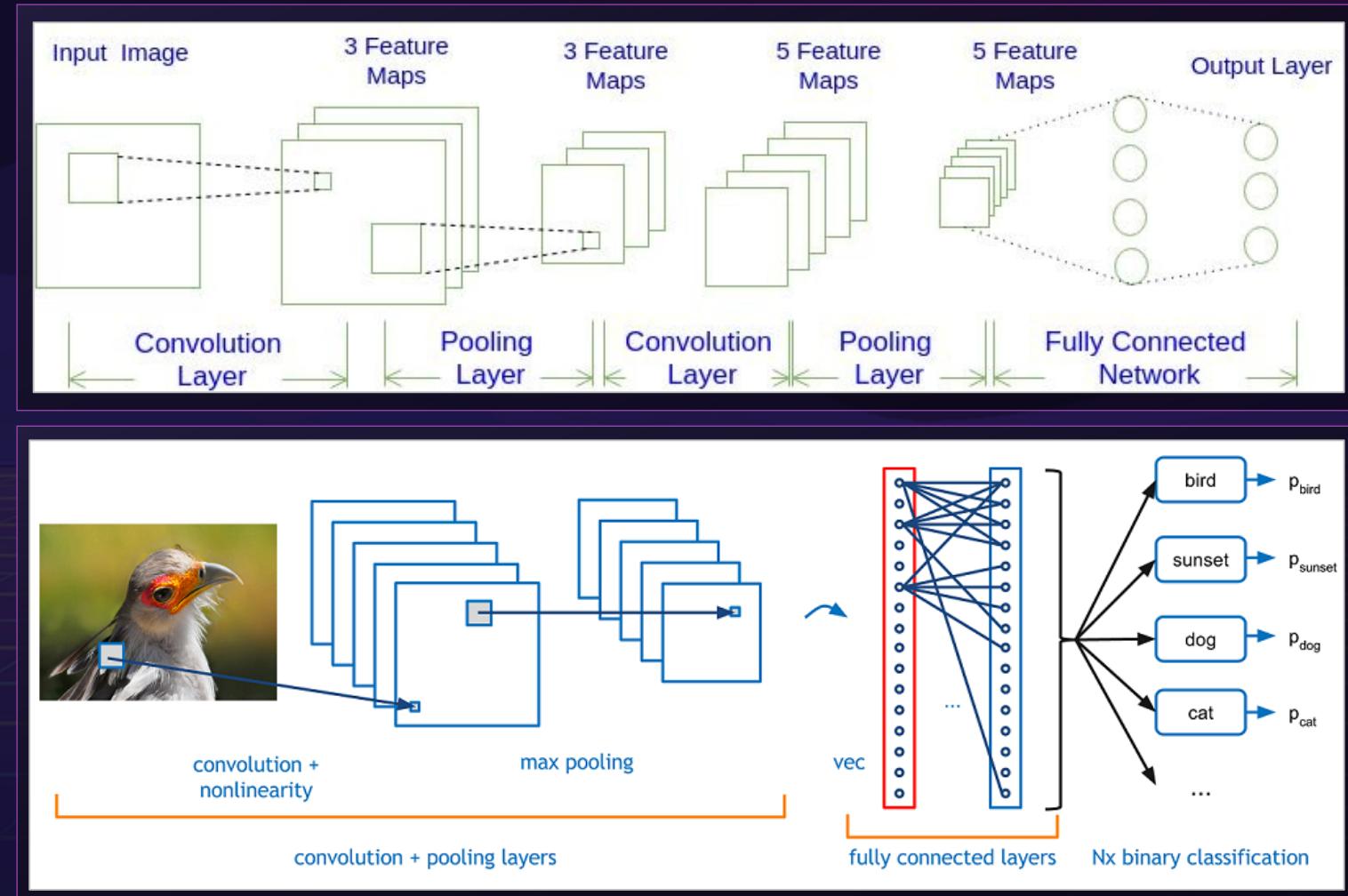
Quiz

1. We usually use gradient descent-based algorithms to train neural network models. ()

- A. True
- B. False



Convolutional Neural Network





Single-Kernel Convolution Calculation (1)

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

image 5*5

1	0	1
0	1	0
1	0	1

bias=0

filter 3*3

red	red	red
red	red	red
red	red	red

feature map 3*3



Single-Kernel Convolution Calculation (1)

1 <small>$\times 1$</small>	1 <small>$\times 0$</small>	1 <small>$\times 1$</small>	0	0
0 <small>$\times 0$</small>	1 <small>$\times 1$</small>	1 <small>$\times 0$</small>	1	0
0 <small>$\times 1$</small>	0 <small>$\times 0$</small>	1 <small>$\times 1$</small>	1	1
0	0	1	1	0
0	1	1	0	0

Image

4		

Convolved Feature

Multi-Kernel Convolution Calculation

Input image

7*7*3

Pad = 1

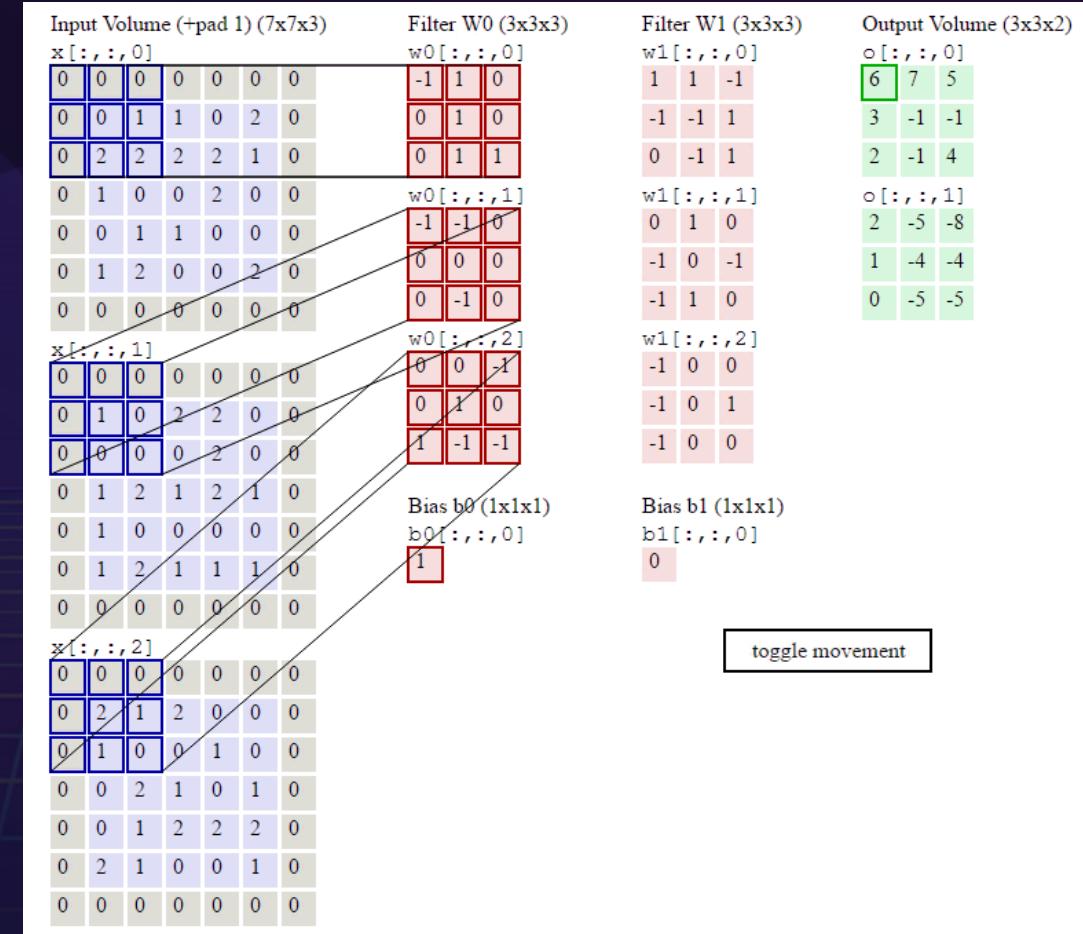
2 convolution kernels

3*3*3

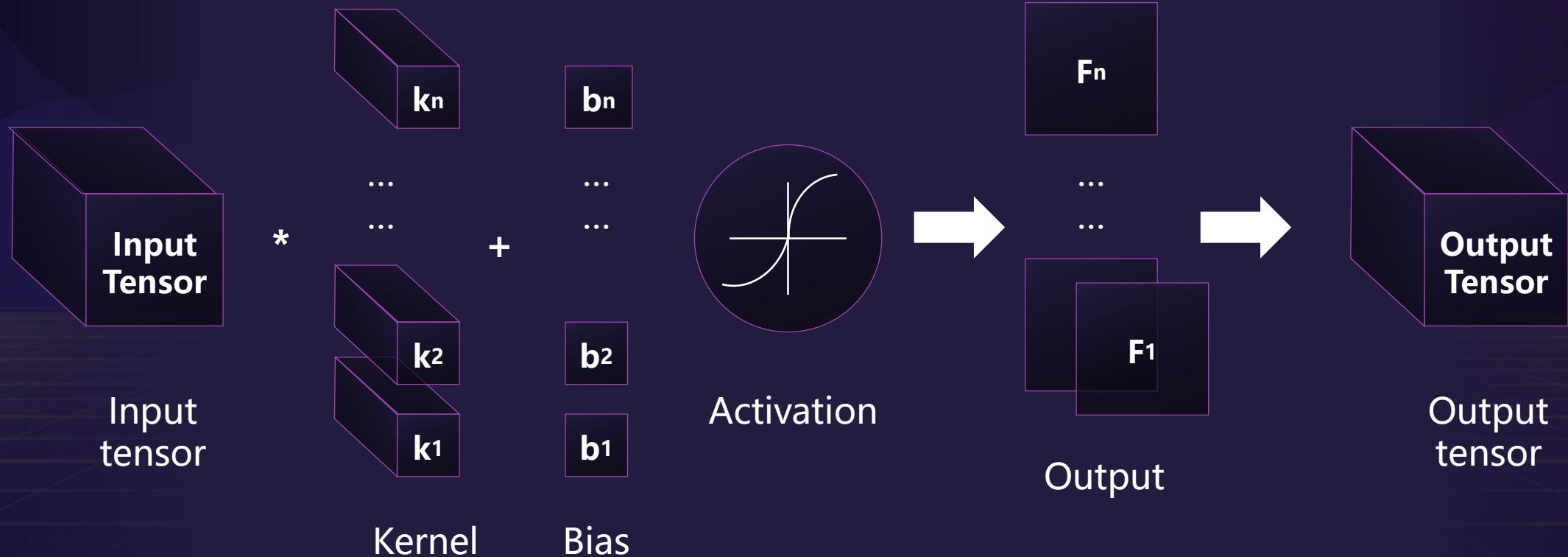
stride = 2

Feature Map

3*3*2



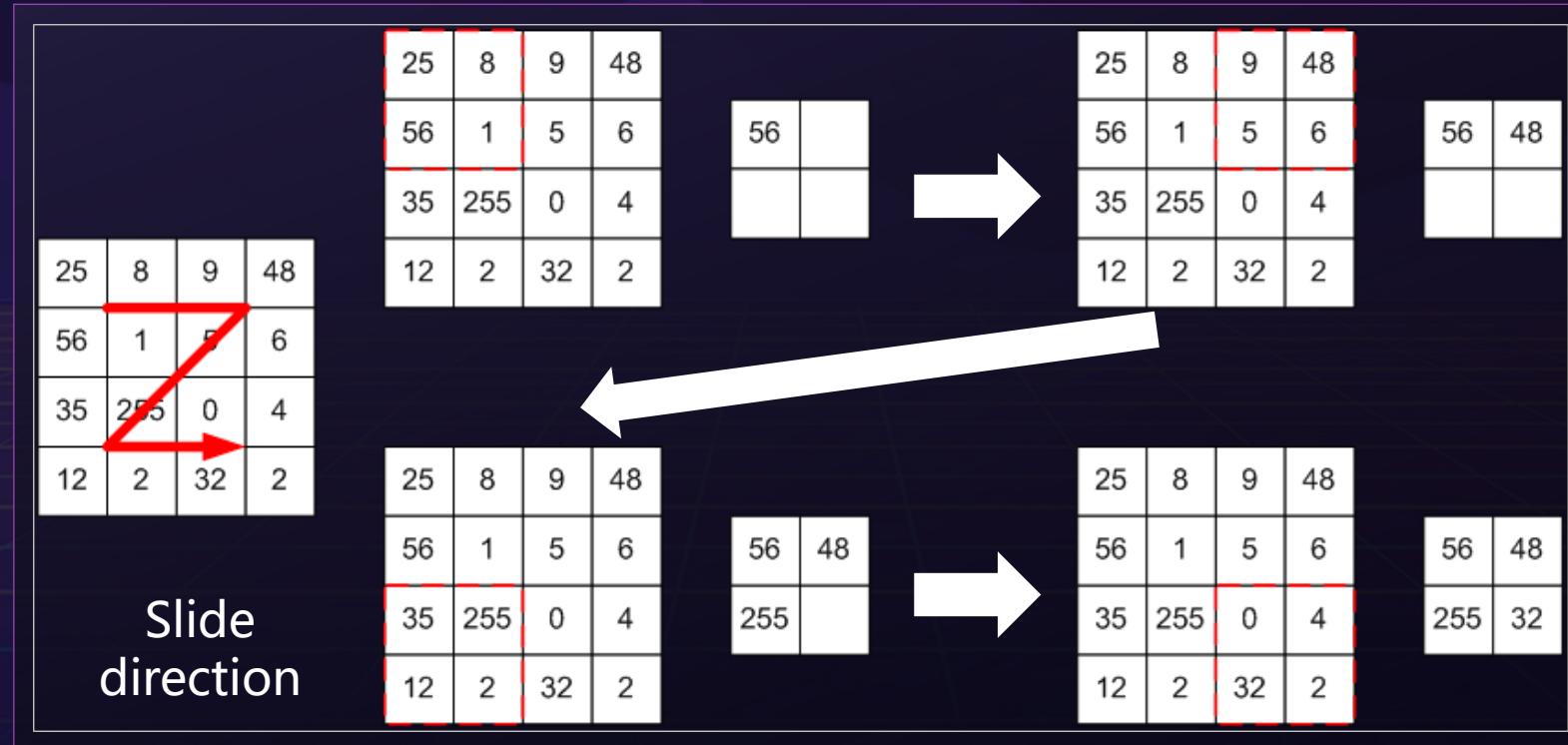
Convolutional Layer





Pooling Layers

- ◆ Pooling combines the nearby units to reduce the size of the input for the next layer. It includes **max pooling** and **average pooling**.





Quiz

- 1. Convolutional neural networks are commonly used to process images. ()**
A. True
B. False

- 2. When the inputs are the same, the feedforward neural networks (FCN and CNN) will always have the same outputs. ()**
A. True
B. False

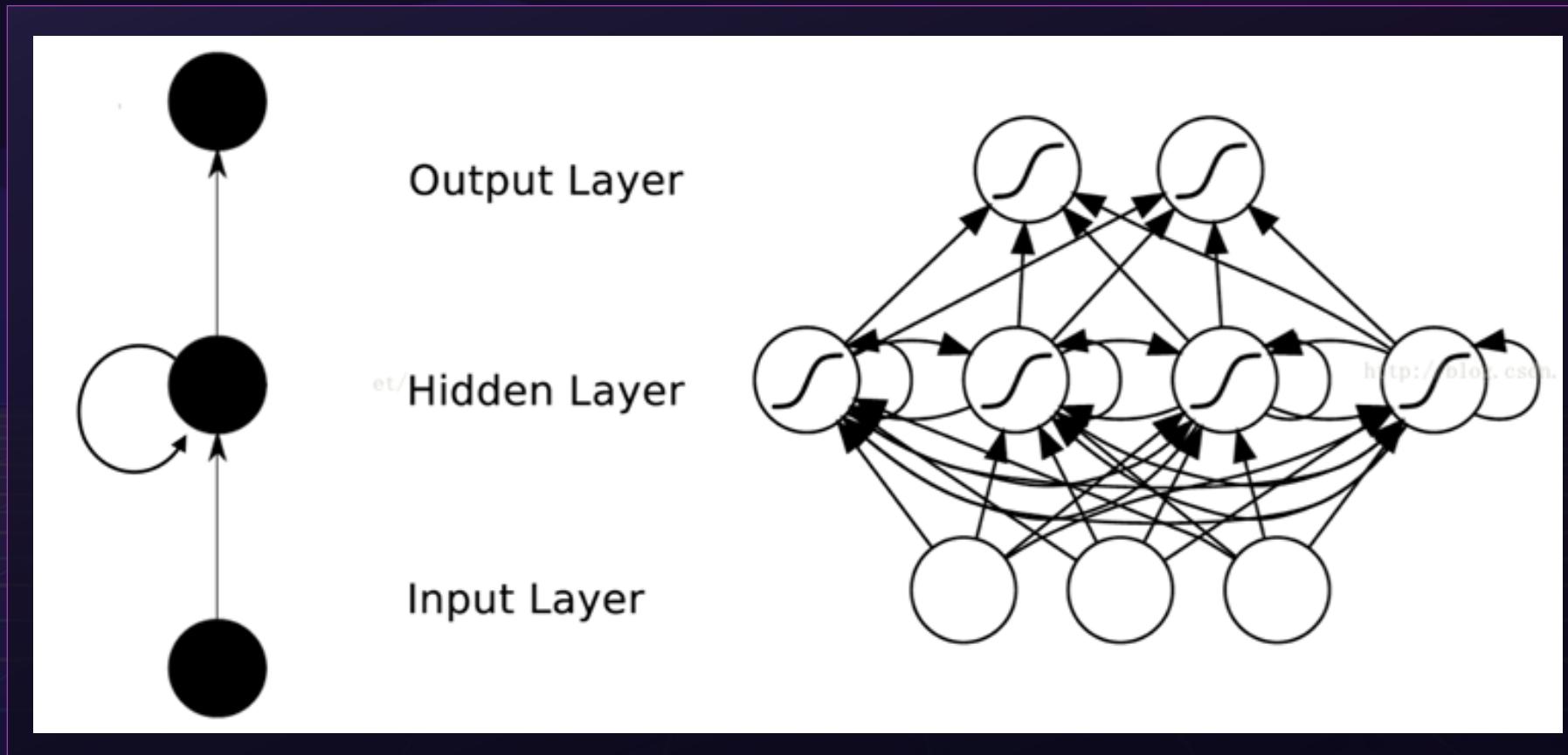
Recurrent Neural Network (1)

- ◆ The recurrent neural network (RNN) is a neural network that captures dynamic information in sequential data through periodical connections of hidden layer nodes. It can classify sequential data.
- ◆ Unlike other forward neural networks, the RNN can keep a context state and even store, learn, and express related information in context windows of any length. Different from traditional neural networks, it extends in space and time sequences. In other words, the hidden layers of the current moment and the next moment are related.
- ◆ The RNN is widely used in scenarios related to sequences, such as videos consisting of image frames, audio consisting of clips, and sentences consisting of words.



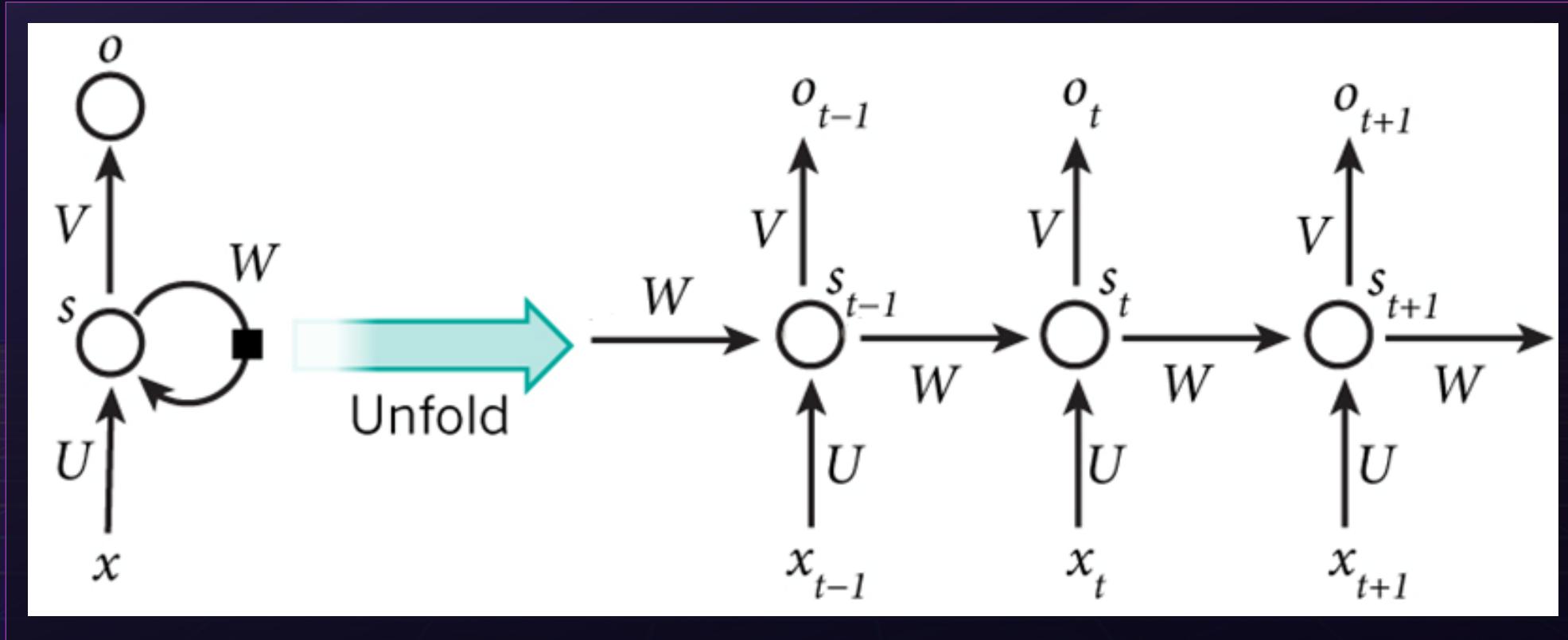


Recurrent Neural Network (2)





Unfolded Recurrent Neural Network





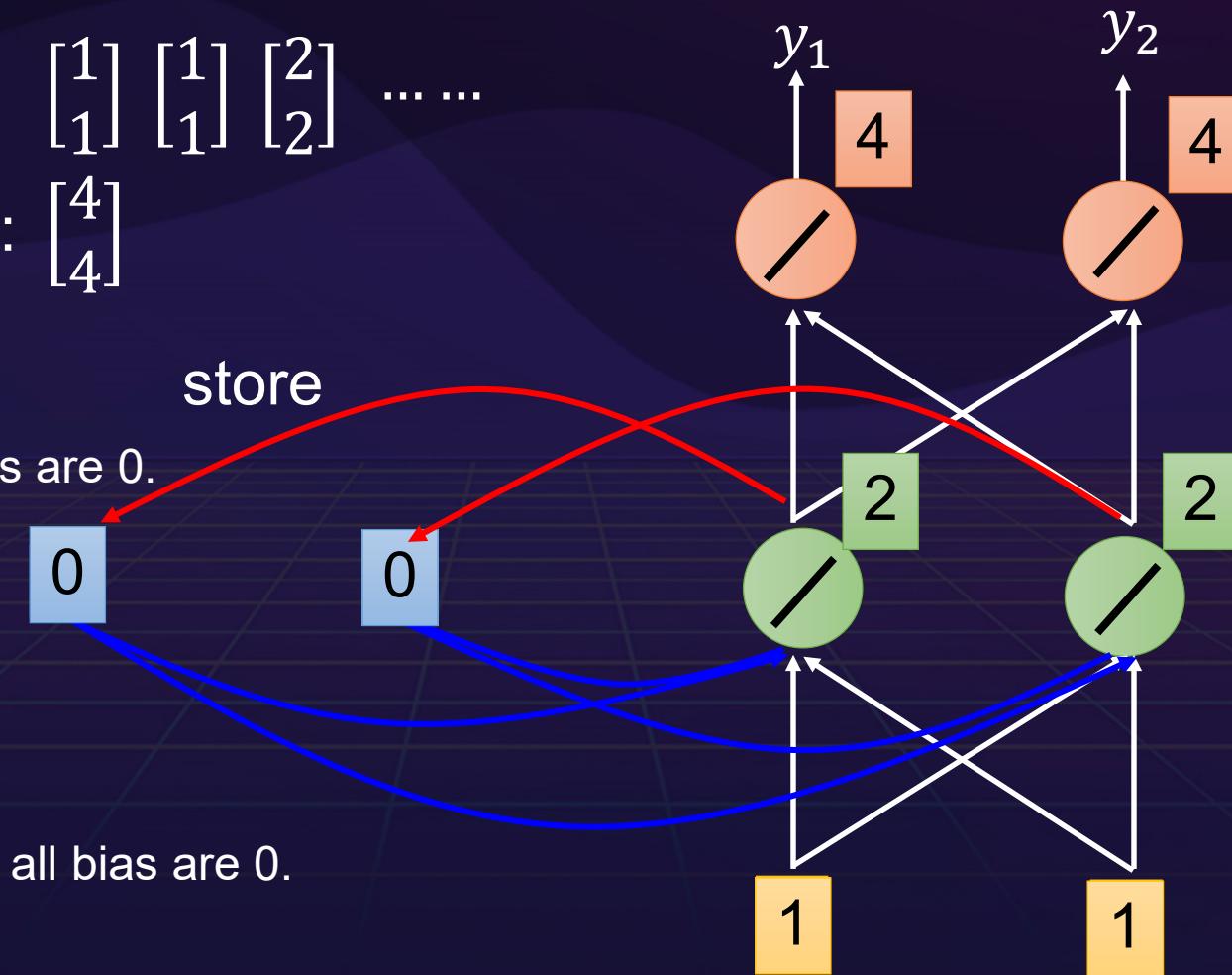
Example of Recurrent Neural Network (1)

Input sequence: $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \dots \dots$

output sequence: $\begin{bmatrix} 4 \\ 4 \end{bmatrix}$

The initialized memory cells are 0.

All weights are 1 and all bias are 0.

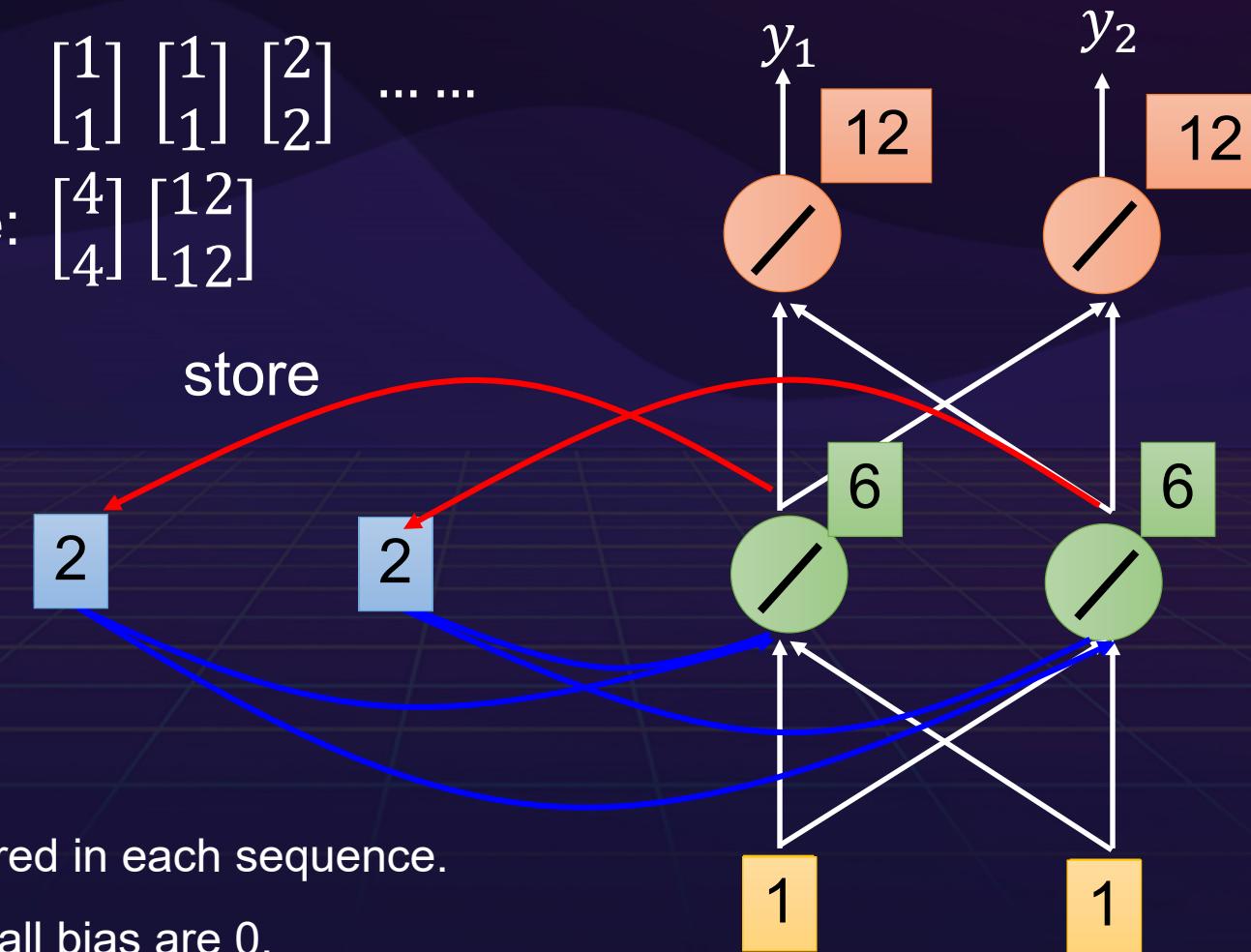




Example of Recurrent Neural Network (2)

Input sequence: $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \dots \dots$

output sequence: $\begin{bmatrix} 4 \\ 4 \end{bmatrix} \begin{bmatrix} 12 \\ 12 \end{bmatrix}$



The weights and bias are shared in each sequence.

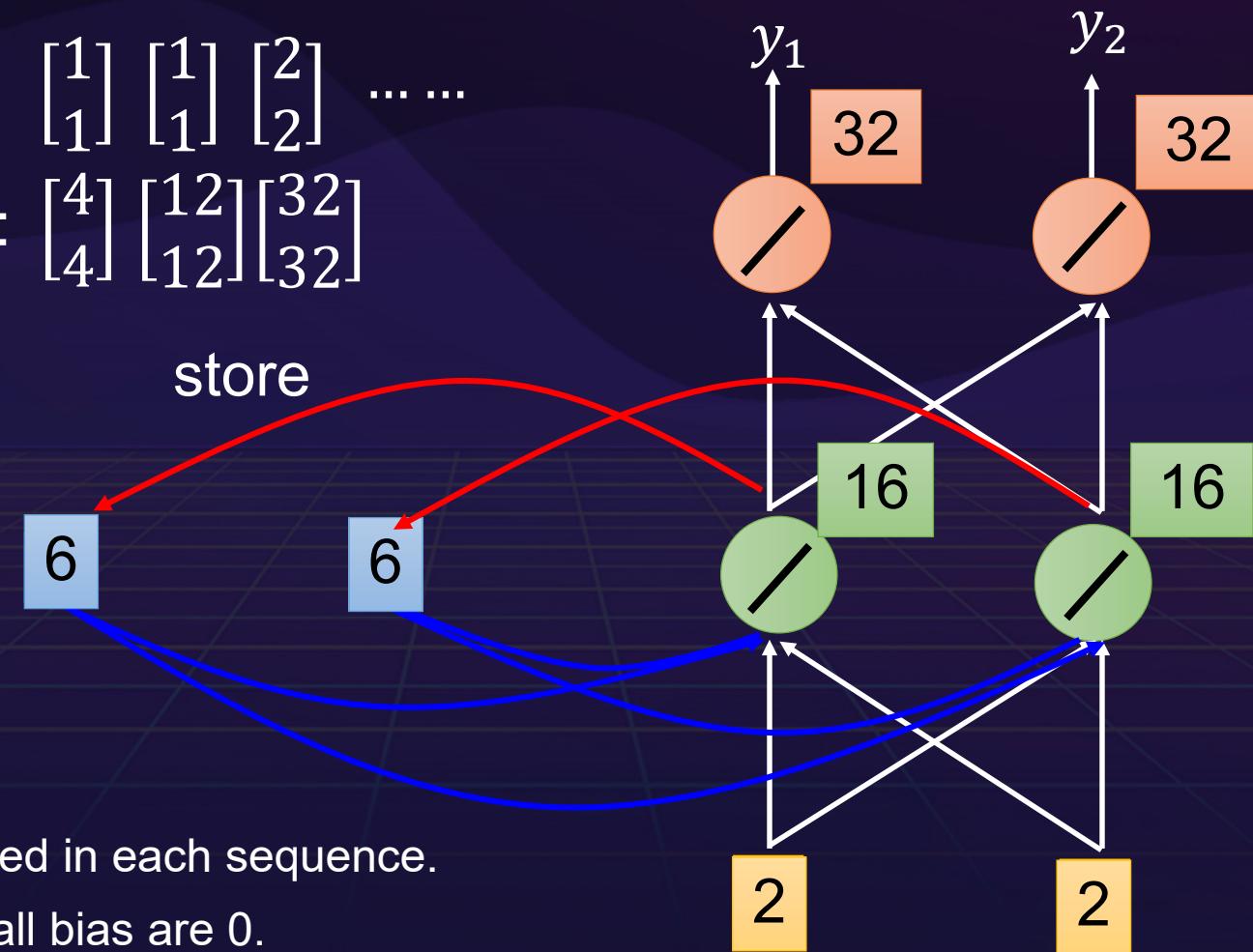
All weights are 1 and all bias are 0.



Example of Recurrent Neural Network (3)

Input sequence: $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \dots \dots$

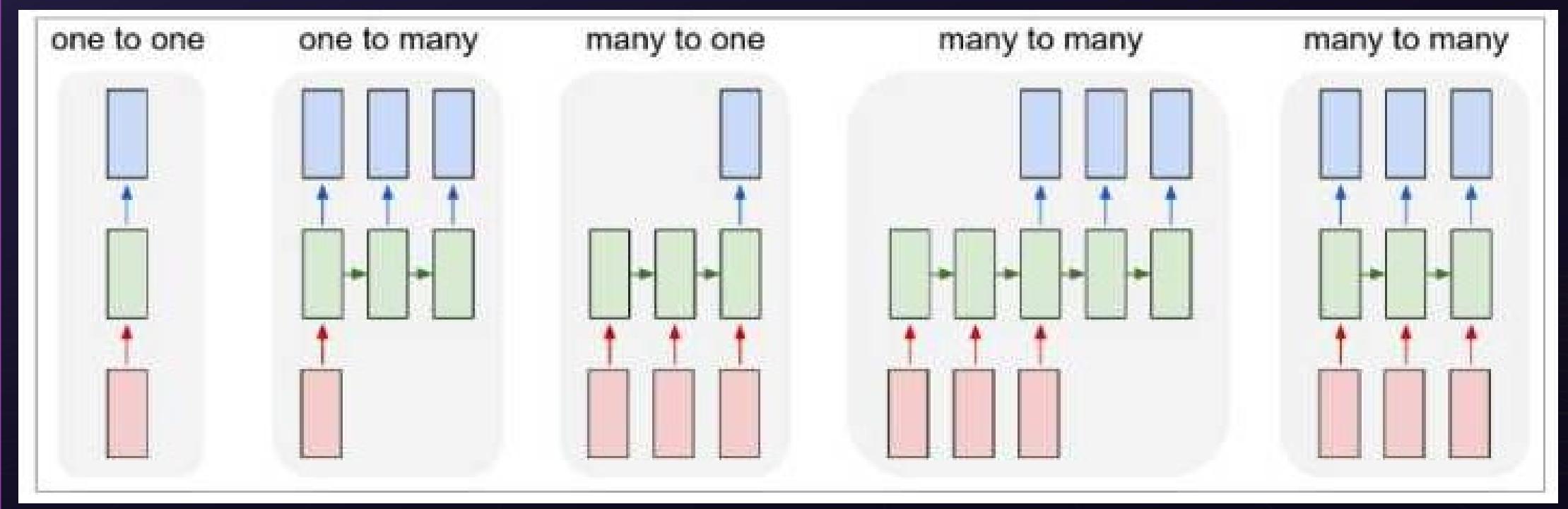
output sequence: $\begin{bmatrix} 4 \\ 4 \end{bmatrix} \begin{bmatrix} 12 \\ 12 \end{bmatrix} \begin{bmatrix} 32 \\ 32 \end{bmatrix}$



The weights and bias are shared in each sequence.

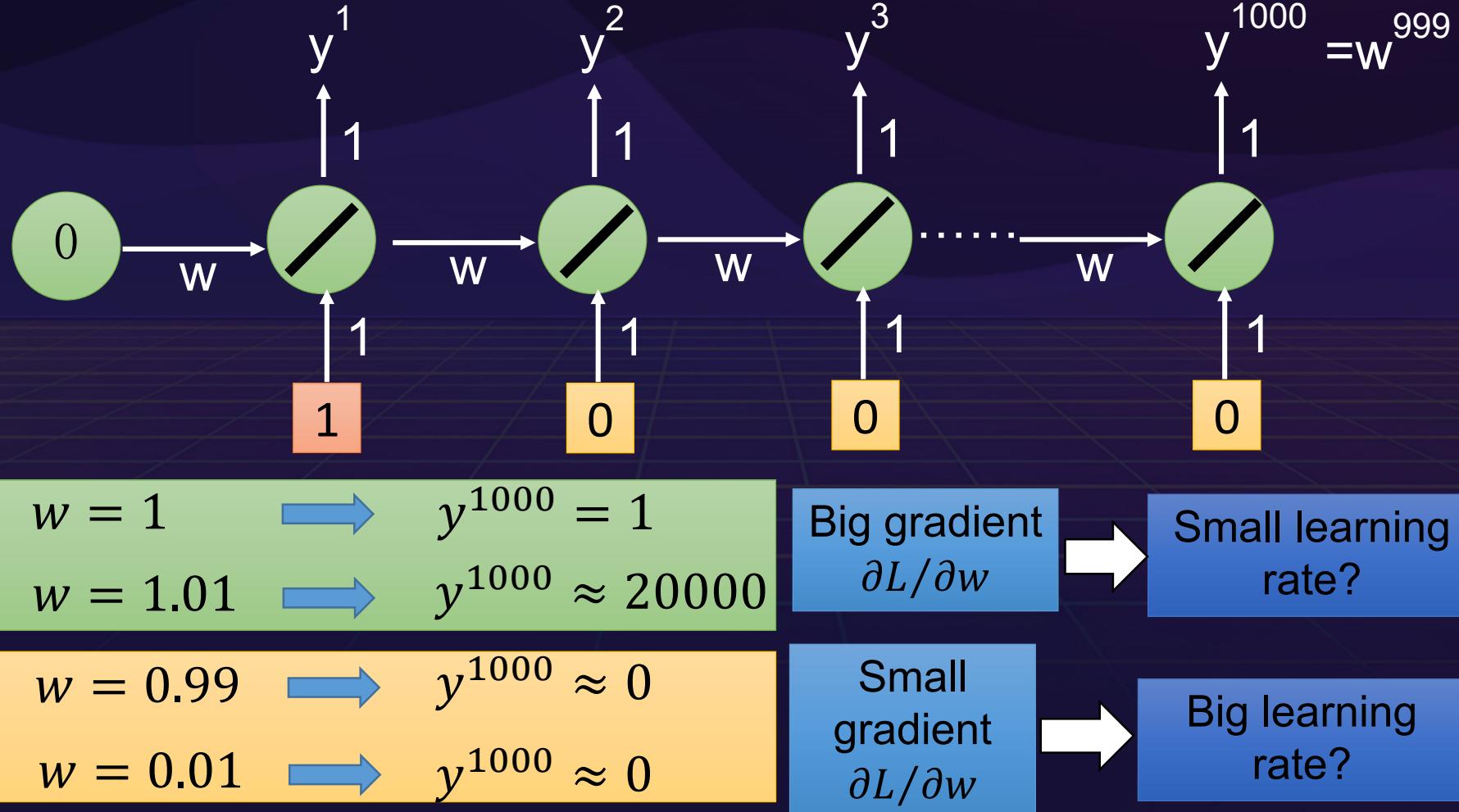
All weights are 1 and all bias are 0.

Types of Recurrent Neural Networks



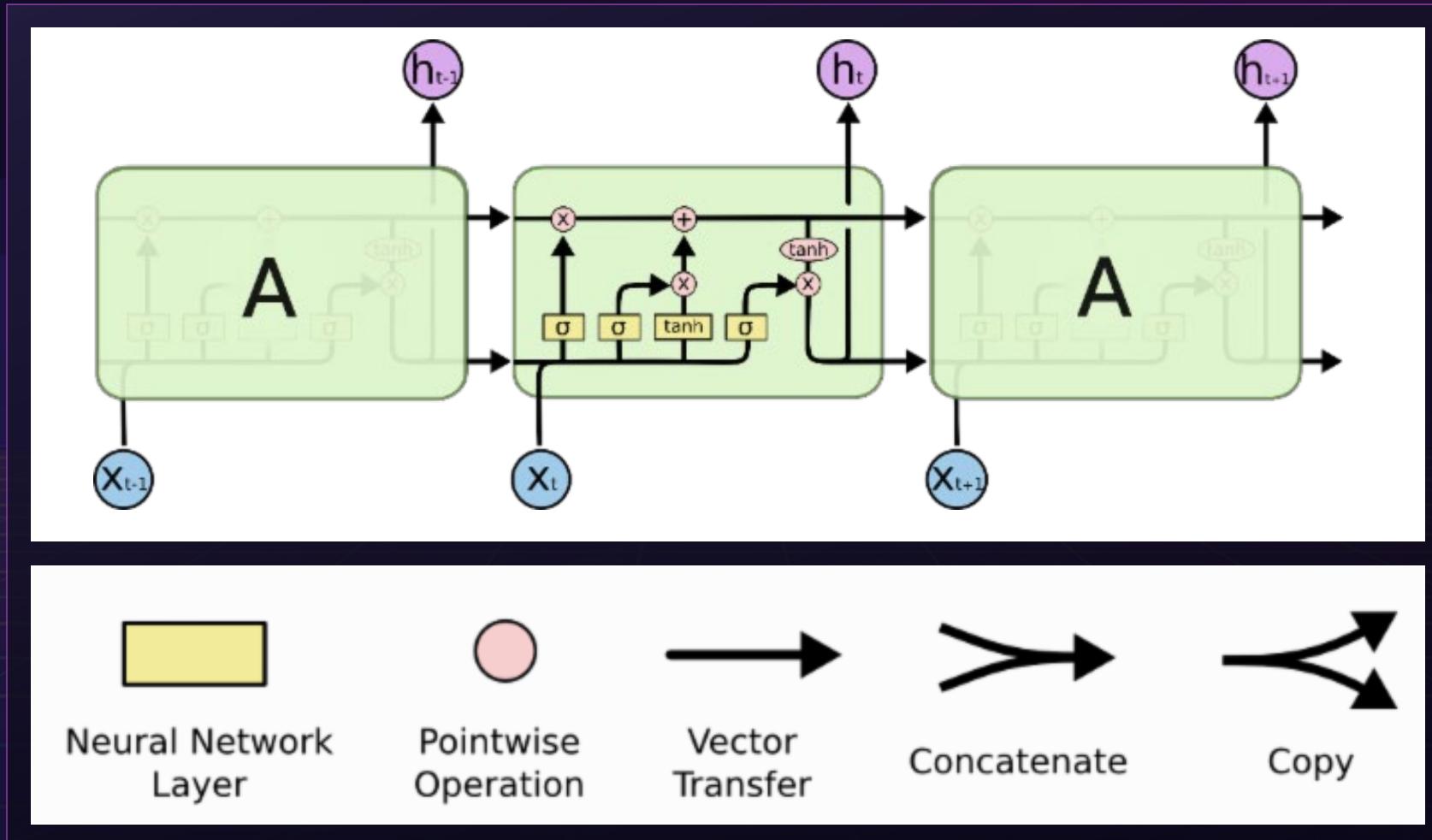


Vanishing gradient and gradient explosion problems in RNN



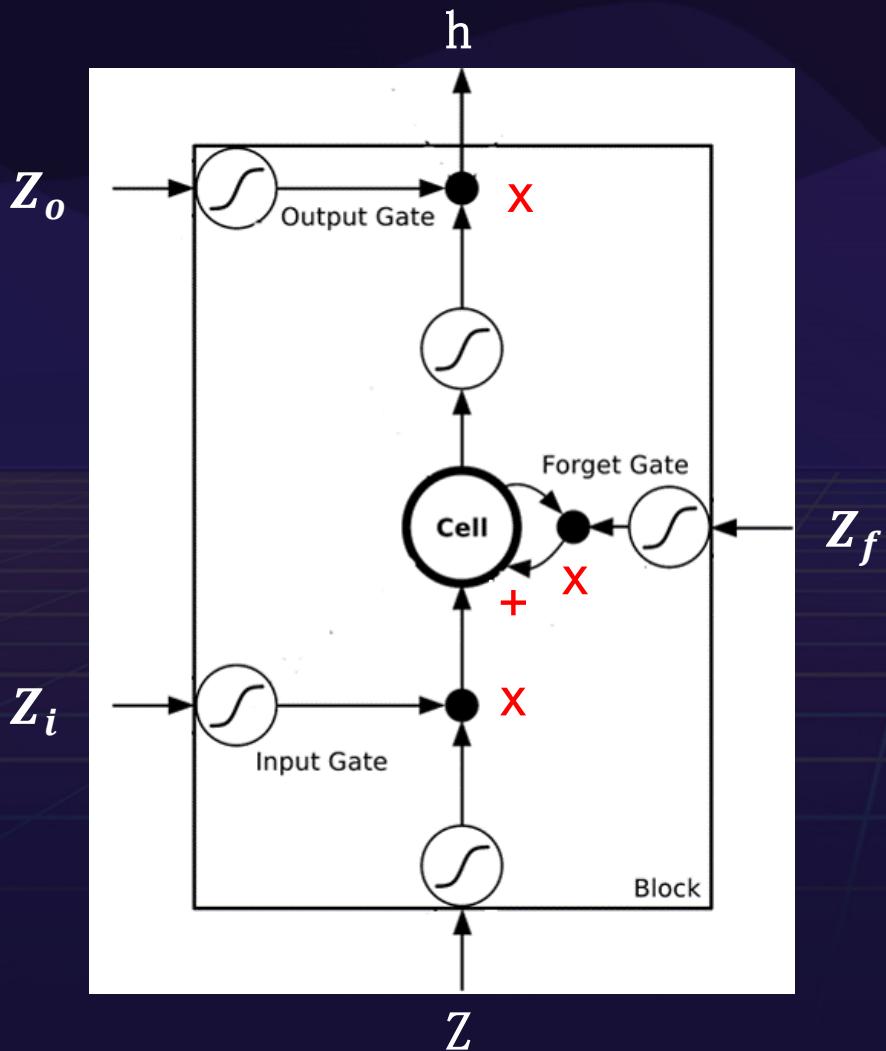


Standard LSTM Structure





LSTM Unit



$$Z = -100$$

$$Z_i = 100$$

$$Z_f = 100$$

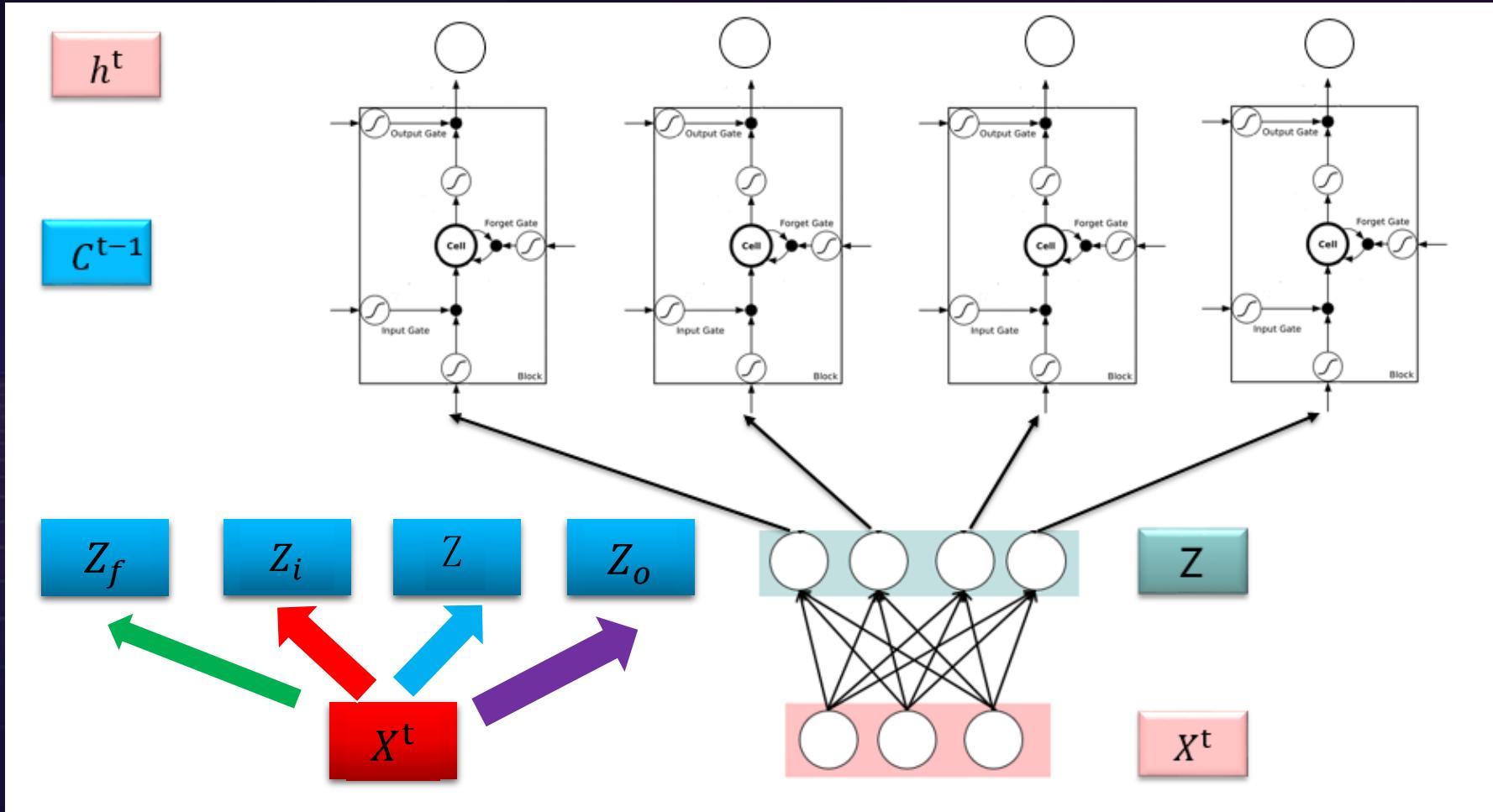
$$Z_o = 100$$

If the value of old cell is 1.

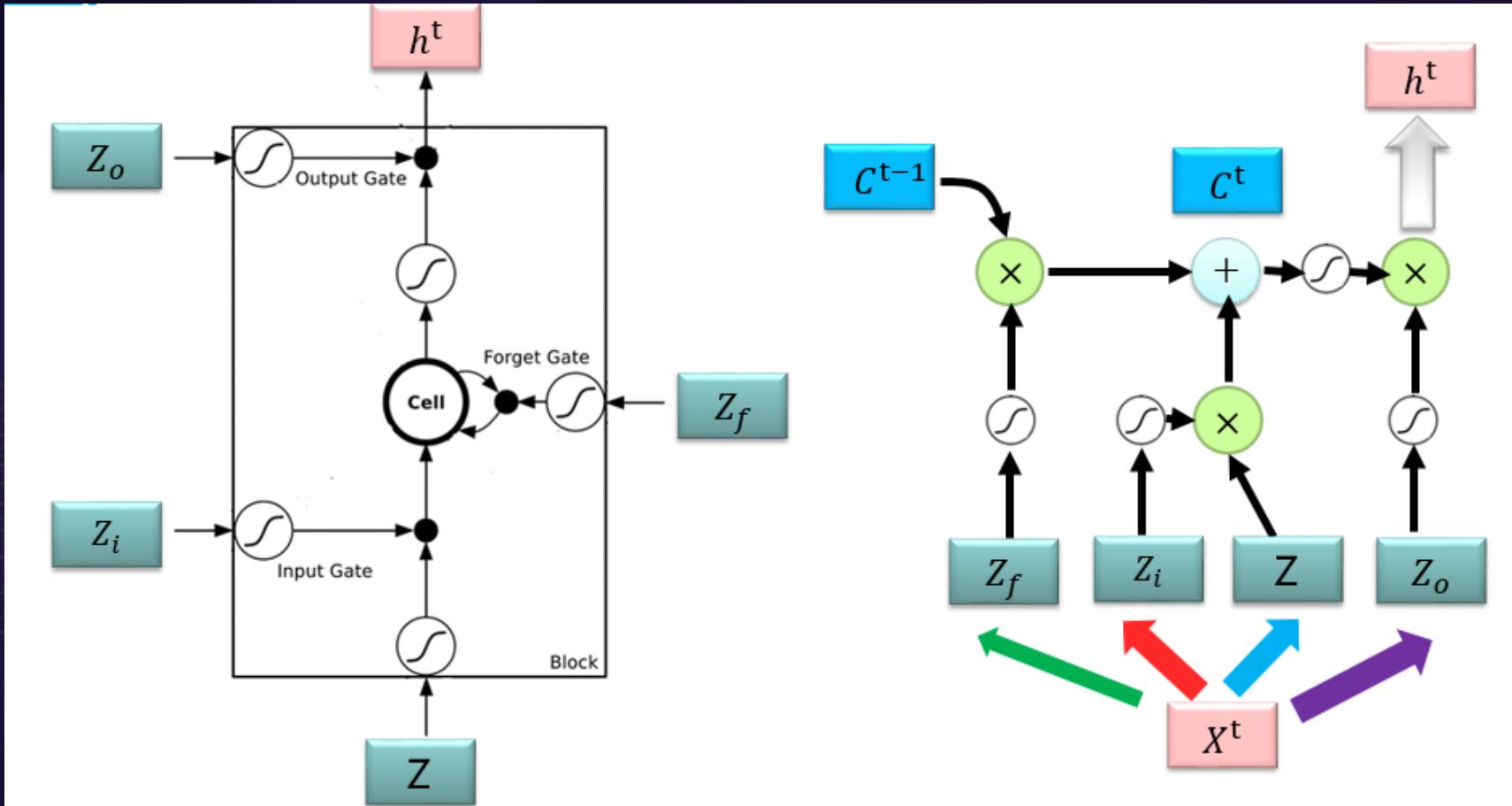
What is the value of the cell now?

What is the value of h ?

LSTM Unit

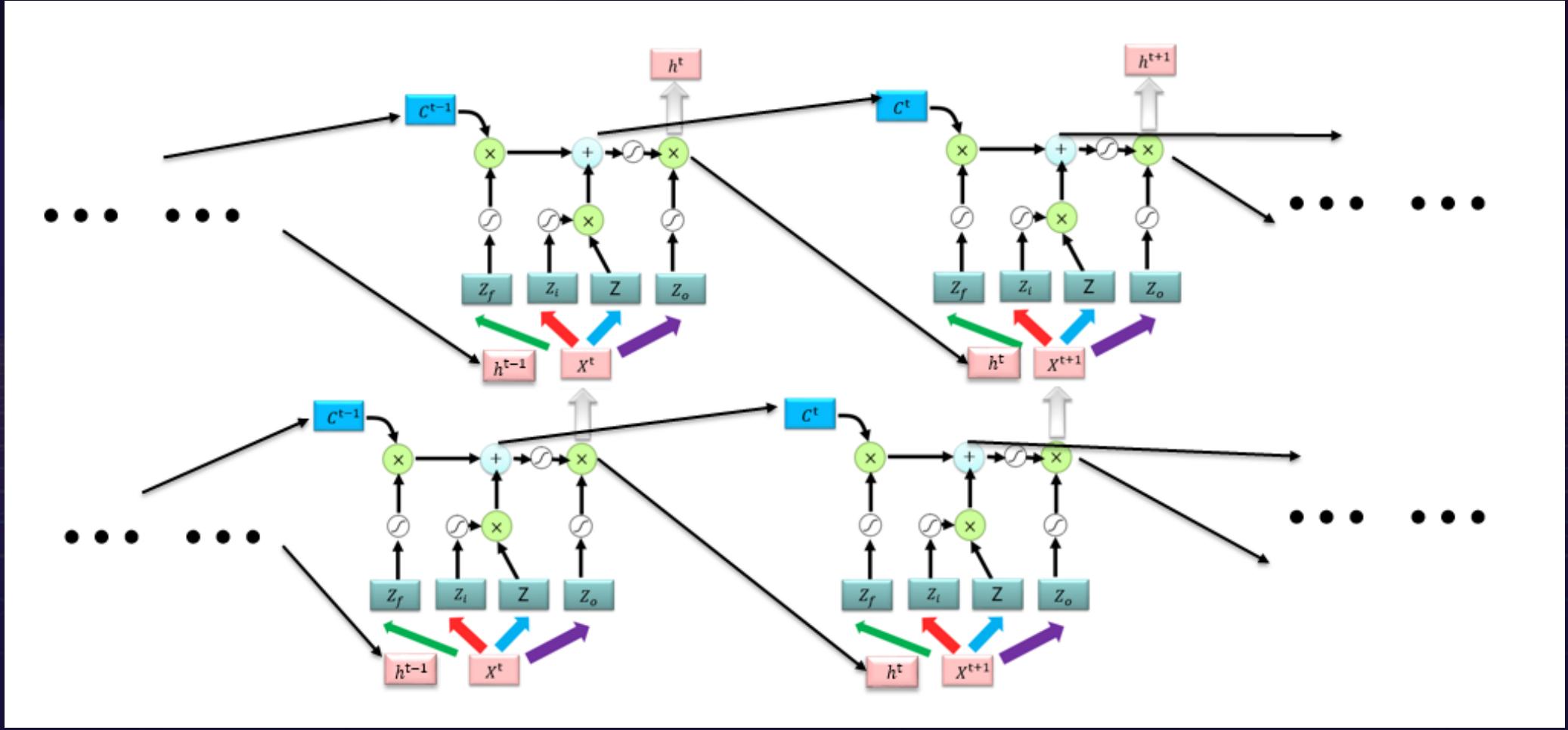


LSTM Unit





LSTM Structure





Quiz

- 1. LSTM model can alleviate the vanishing gradient problem in RNN. ()**
A. True
B. False

- 2. The gradient explosion problem can be solved with simple gradient clipping method. ()**
A. True
B. False

Generative Adversarial Network

Someone asked a question on the Internet: What are some recent and potentially upcoming breakthroughs in unsupervised learning?



A screenshot of a Quora post by Yann LeCun. The post is titled "What are some recent and potentially upcoming breakthroughs in unsupervised learning?". Yann LeCun is identified as "Yann LeCun, Director of AI Research at Facebook and Professor at NYU" and has a blue checkmark next to his name. The post was written on July 29 and upvoted by Joaquin Quiñonero Candela, Director Applied Machine Learning at Facebook, and Huang Xiao. The quote from Yann LeCun reads: "Adversarial training is the coolest thing since sliced bread."

Generative Adversarial Networks (GAN) were proposed in 2014 by Ian Goodfellow. GAN is a one kind of Neural Network, and all the layers are the fully connected layers.



What does GAN use for?

Machine learning is to find a function f:

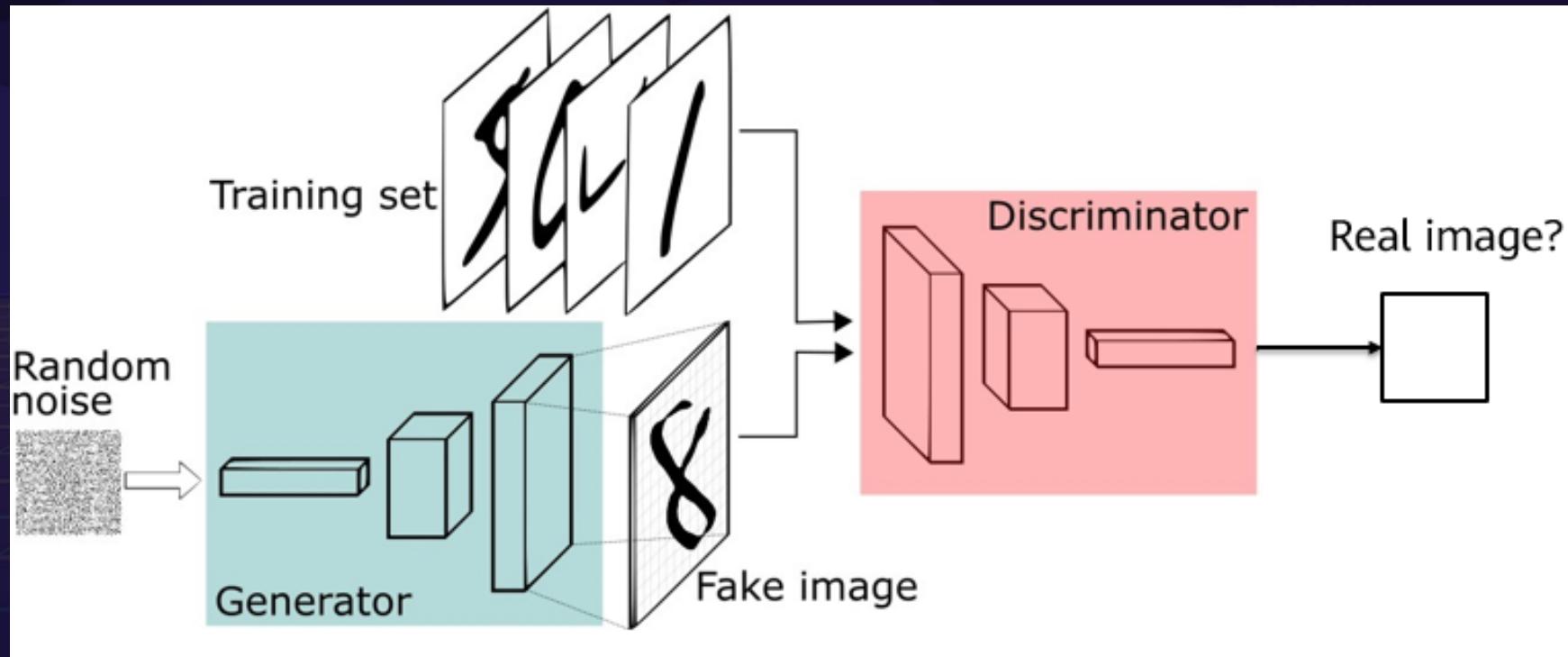
$$f: x \rightarrow y$$

GAN can use the **unsupervised learning** to create matrix, sequences and so on.

So, the input of GAN can be the data without labels, and the output of GAN can be images, sentences or voice.



The structure of the GAN



GAN Training Rules

- Optimization target:

- Value function:

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_{z(z)}}[\log (1 - D(G(z)))]$$

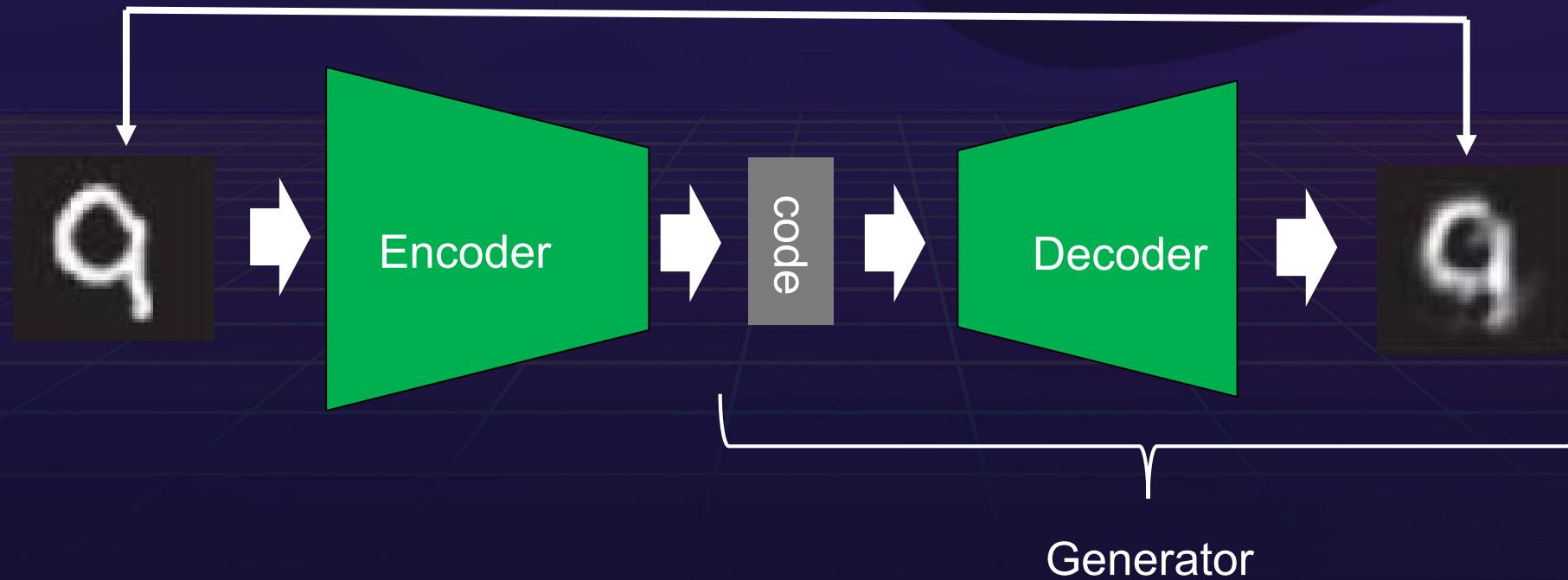
- In the early training stage, when the generation effect of G is very poor, D determines that the generated sample is fake with high confidence, because the sample is obviously different from training data. In this case, $\log(1-D(G(z)))$ is saturated (that is, a constant, with the gradient being 0 and iteration being unavailable). Therefore, minimize **$[-\log(D(G(z)))]$** instead of minimizing **$\log(1-D(G(z)))$** when training G.



Why the generator cannot leave the discriminator

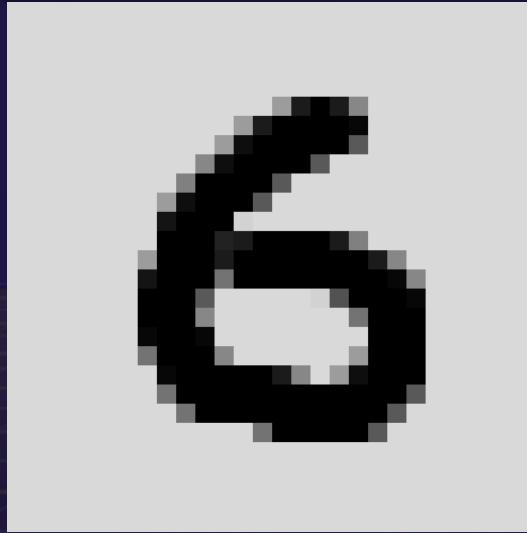
Autoencoder:

As close as possible

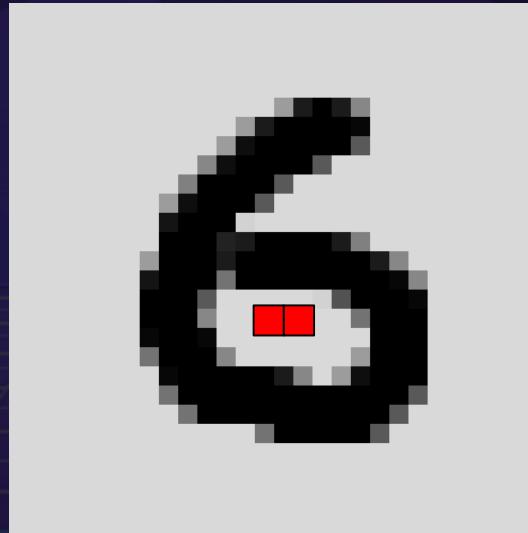




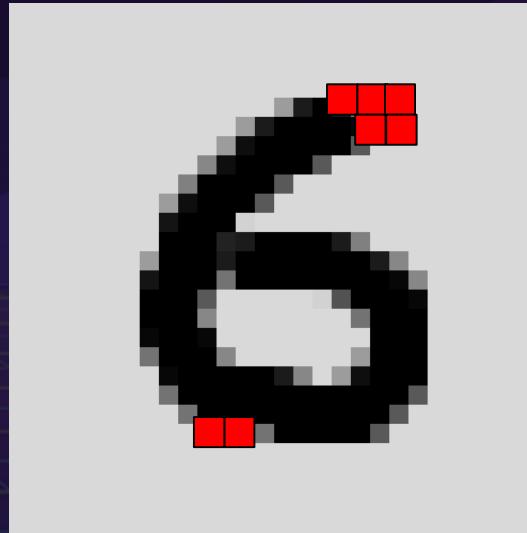
Why the generator cannot leave the discriminator



Original



Bad



Good



Quiz

1. GAN can use the unsupervised learning to generate the images. ()

- A. True**
- B. False**

2. The autoencoder also belongs to the feedforward neural network. ()

- A. True**
- B. False**

Contents

1. Propaedeutics of Deep Learning

2. Deep Learning Overview

- Definition and Development of Neural Networks
 - Perceptron and Training Rules
 - Activation Functions
 - Types of Neural Networks
 - Regularization in Deep Learning
- Optimizer
- Applications of Deep Learning



Regularization in Deep Learning

- Adding constraints to parameters, such as L_1 and L_2 norms
- Expanding the training set, such as adding noise and transforming data
- Dropout



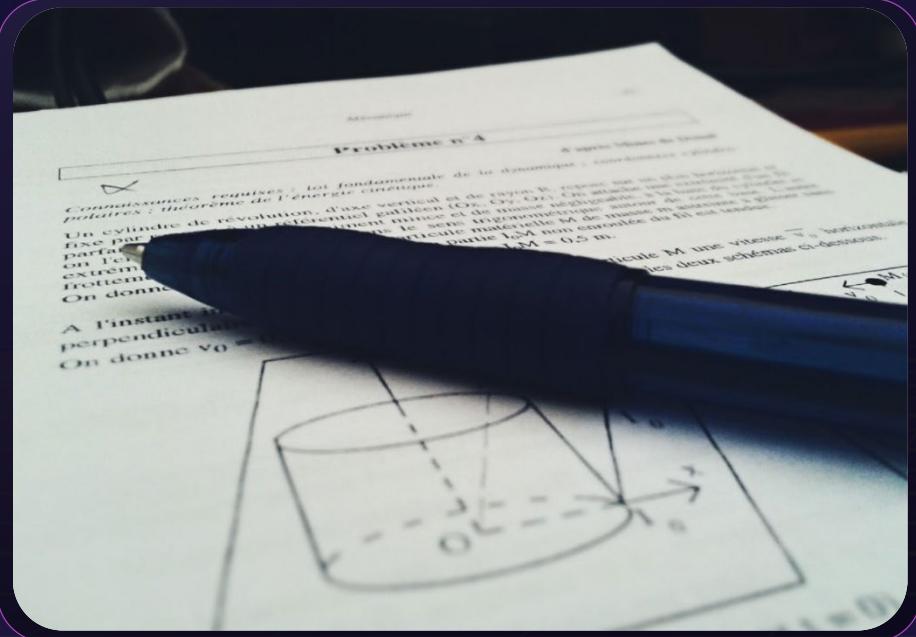


Parameter Penalties

- ◆ Many regularization approaches add a parameter penalty $\Omega(\theta)$ to the objective function J , limiting the learning capability of models. We denote the regularized objective function as \tilde{J} :

$$\tilde{J}(\theta; X, y) = J(\theta; X, y) + \alpha\Omega(\theta)$$

where $\alpha \in [0, \infty)$ is a hyperparameter that weights the relative contribution of the norm penalty term, Ω , relative to the standard objective function $J(X; \theta)$. Setting α to 0 results in no regularization. Larger values of α correspond to more regularization.





L_2 Regularization

- ◆ Add L_2 parameter norm penalty to prevent overfitting.

$$\tilde{J}(w; X, y) = J(w; X, y) + \frac{1}{2} \alpha \|w\|^2$$

- ◆ The parameter optimization method can be inferred using the optimization technology (such as gradient correlation method):

$$w = (1 - \varepsilon\alpha)\omega - \varepsilon\nabla J(w)$$

- ◆ where ε is the learning rate. This multiplies parameters by a reduction factor compared with the common gradient optimization function.



L_1 Regularization

- ◆ Add L_1 norm constraint to model parameters:

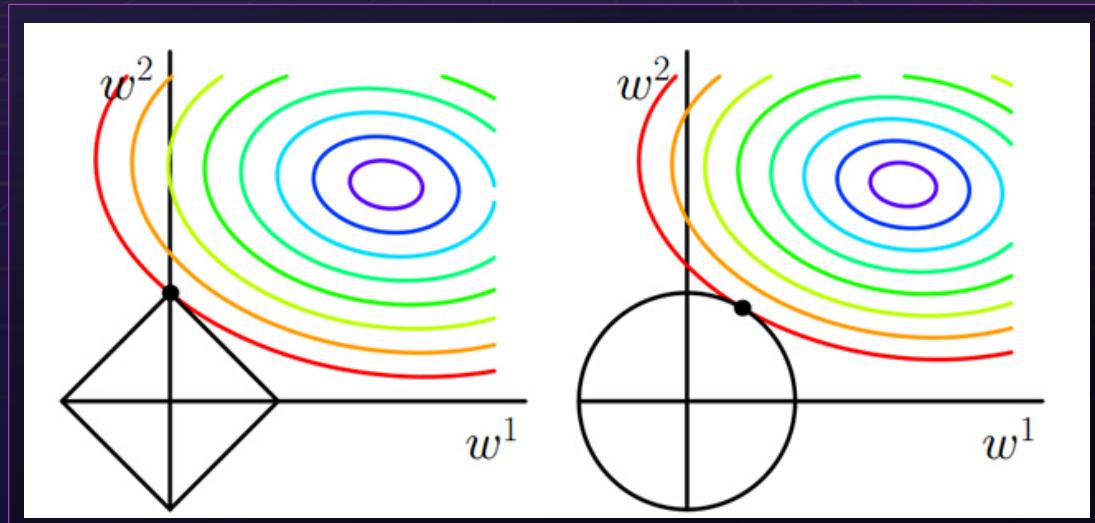
$$\tilde{J}(w; X, y) = J(w; X, y) + \alpha \|w\|_1$$

- ◆ If the gradient method is used to solve the problem, the parameter gradient is

$$\nabla \tilde{J}(w) = \alpha sign(w) + \nabla J(w)$$

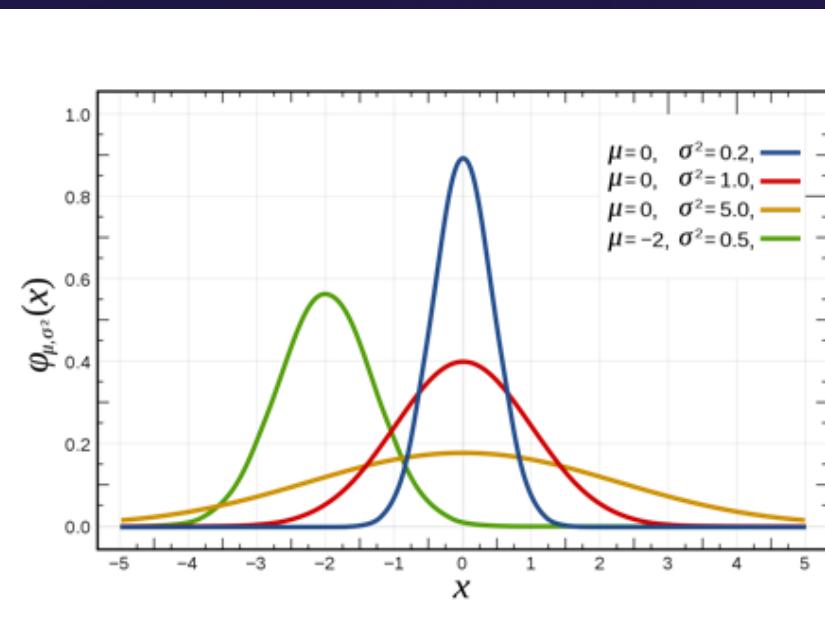
L_2 VS L_1

- ◆ Major differences between L_2 and L_1 :
 - According to the preceding analysis, L_1 can generate a more sparse model than L_2 . When the parameter w is small, L_1 regularization can directly reduce the parameter to 0, which can be used for feature selection.
 - From the perspective of probability, many norm constraints are equivalent to adding prior distribution to parameters. L_2 norm equals to Gaussian distribution while L_1 norm equals to Laplace distribution.

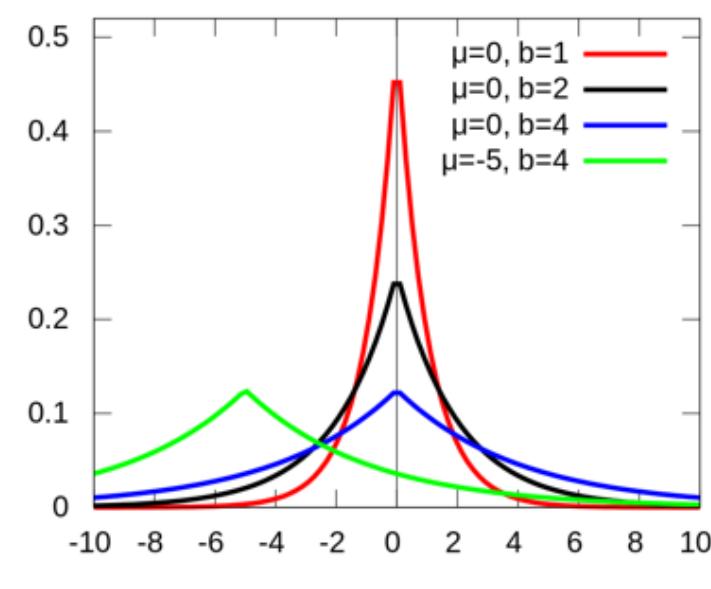


L₂ VS L₁

- ◆ Major differences between L₂ and L₁:
 - From the perspective of probability, many norm constraints are equivalent to adding prior distribution to parameters. L₂ norm equals to Gaussian distribution while L₁ norm equals to Laplace distribution.



Gaussian distribution



Laplace distribution



Dataset Expansion

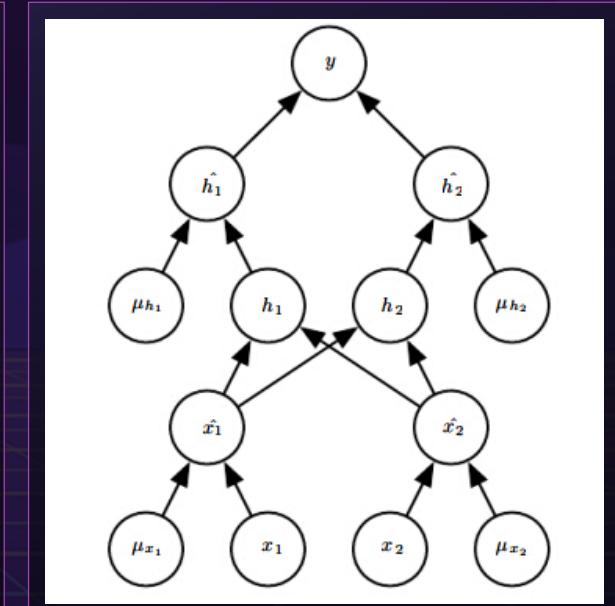
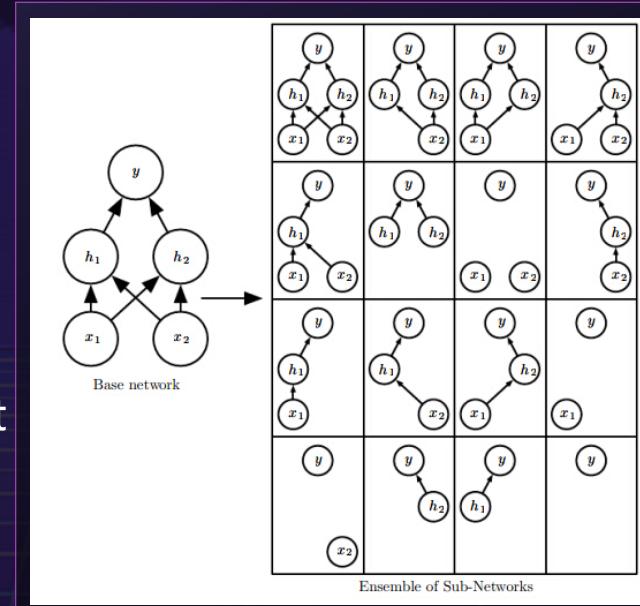


- Random noise is added to the input data in speech recognition.
- The common idea of NLP is to replace words with their synonyms.
- Noise injection can add noise to the input or to the hidden layer or output layer. For example, for the softmax classification problem, noise can be added by using the label smoothing technology. If noise is added to the 0-1 category, the corresponding probability is changed to $\frac{\varepsilon}{k}$ and $1 - \frac{k-1}{k}\varepsilon$.



Dropout

- ◆ Dropout is a common and simple regularization method, which has been widely used since 2014. Simply put, dropout randomly discards some inputs during the training process. In this case, the parameters corresponding to the discarded inputs are not updated. Dropout is an integration method. It combines all sub-network results and obtains sub-networks by randomly dropping inputs.





Quiz

1. (Multiple choice) Which of the following method can alleviate the overfitting problem in deep learning? ()

- A. L1 regularization
- B. L2 regularization
- C. Dataset expansion
- D. Dropout

Contents

1. Propaedeutics of Deep Learning

2. Deep Learning Overview

- Definition and Development of Neural Networks
 - Perceptron and Training Rules
 - Activation Functions
 - Types of Neural Networks
 - Regularization in Deep Learning
- Optimizer
- Applications of Deep Learning

Optimizer

- ◆ There are various improved versions of gradient descent algorithms. In object-oriented language implementation, different gradient descent algorithms are often encapsulated into an object which is called an **optimizer**.
- ◆ The **purpose** of algorithm improvement includes but is not limited to:
 - Accelerates algorithm convergence.
 - Avoids or overshoots local extrema.
 - Simplifies manual parameter setting, especially the learning rate.
- ◆ Common optimizers: common GD optimizer, momentum optimizer, Nesterov, Adagrad, Adadelta, RMSprop, Adam, AdaMax, and Nadam

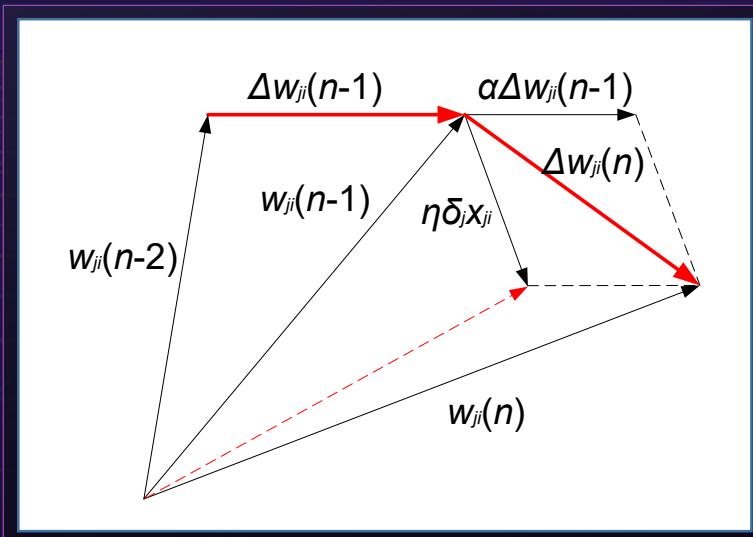


Momentum Optimizer

- ◆ One basic improvement is to add the momentum term for Δw_{ji} . Denote the weight correction of n -th iteration as $\Delta w_{ji}(n)$ and then the weight correction rule is:

$$\Delta w_{ji}(n) = \eta \delta_j x_{ji} + \alpha \Delta w_{ji}(n - 1)$$

where α is a constant ($0 \leq \alpha < 1$) and called momentum and $\alpha \Delta w_{ji}(n - 1)$ is a momentum term.





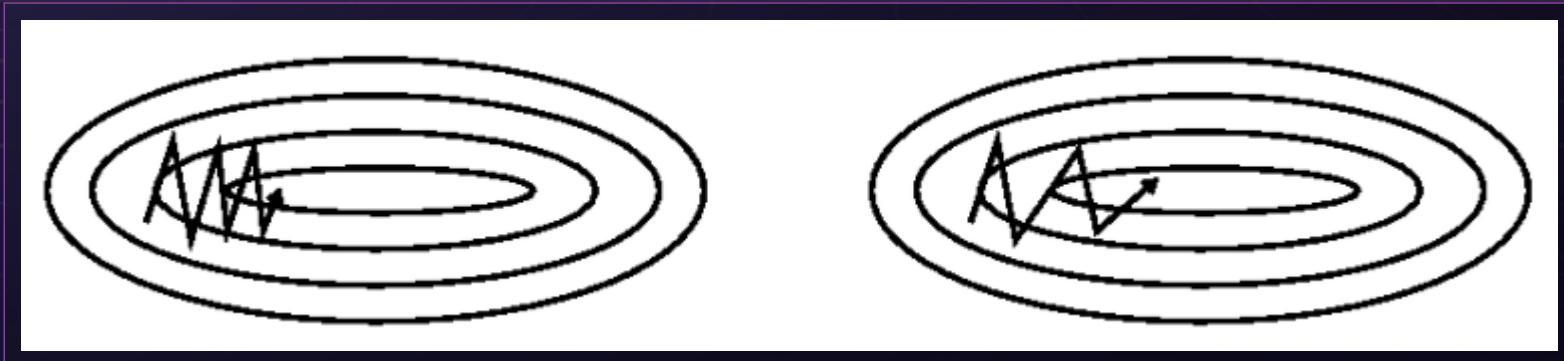
Advantages and Disadvantages

◆ Advantages:

- Enhances the stability of the gradient correction direction and reduces mutations.
- In areas where the gradient direction is stable, the ball rolls faster and faster (there is a speed upper limit because $\alpha < 1$), which helps the ball quickly overshoot the flat area and accelerates convergence.
- A small ball with inertia is more likely to roll over some narrow local extrema.

◆ Disadvantage:

- The learning rate η and momentum α need to be manually set, which often requires more experiments to determine the appropriate value.





Adam Optimizer (1)

- ◆ Adaptive Moment Estimation (Adam): Developed based on Adagrad and Adadelta, Adam maintains two additional variables m_t and v_t for each variable to be trained:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

where t represents t -th iteration and g_t is the calculated gradient. m_t and v_t are moving averages of the gradient and the squared gradient.

Adam Optimizer (2)

- ◆ As m_t and v_t are initialized as vectors of 0's, they are biased towards 0, especially during the initial steps, and especially when β_1 and β_2 are close to 1. To solve this problem, we use \hat{m}_t and \hat{v}_t :

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

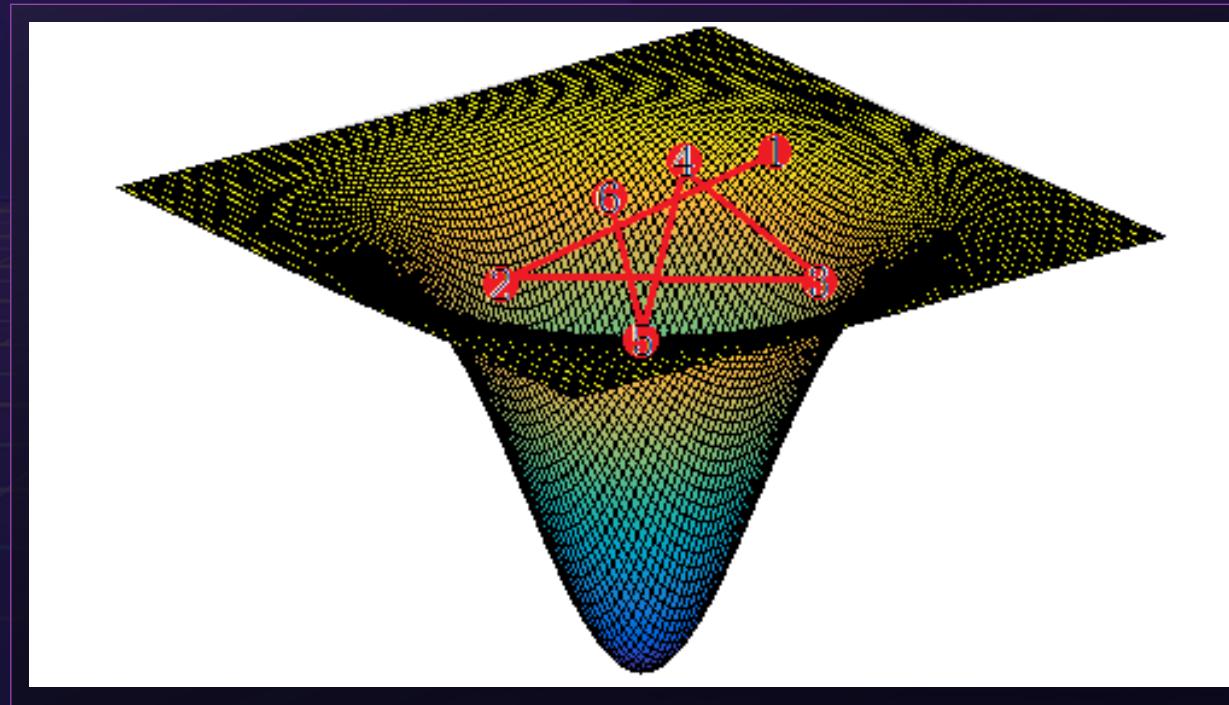
- ◆ Adam's weight update rule:

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

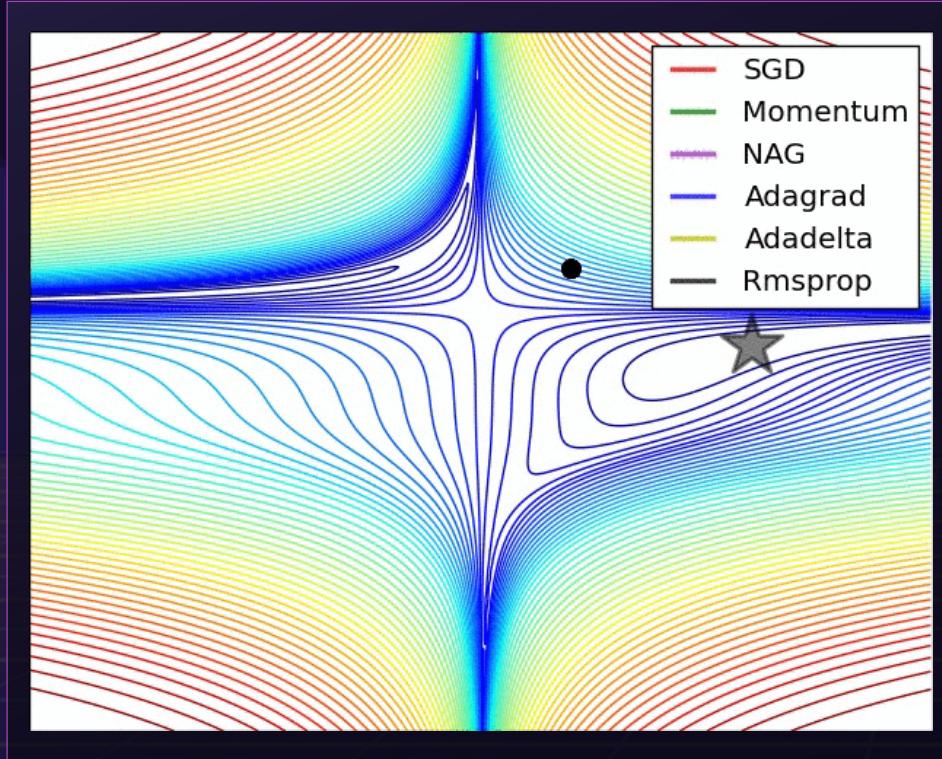
- ◆ Although the rule involves manual setting of η , β_1 , and β_2 , this is much simpler. According to experiments, the default setting are $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and $\eta = 0.001$. In practice, Adam will converge quickly. For convergence saturation, reduce η . After several times of reduction, the satisfying local extrema will be converged. Other parameters do not need adjustment.

Adam Optimizer (3)

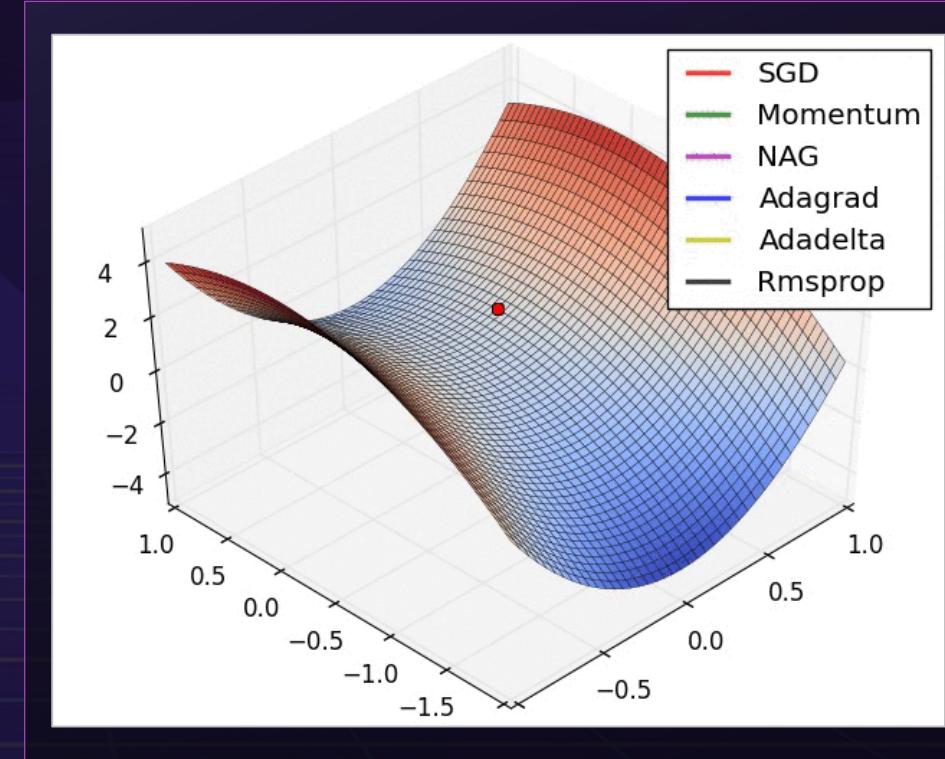
- ◆ \hat{m}_t is the current gradient with a momentum term and $\frac{\eta}{\sqrt{\hat{v}_t + \epsilon}}$ is equivalent to the current learning rate. If the gradient model is too large, the weight cannot jump out of the extrema or converge. Then we need to reduce the learning rate to facilitate convergence.



Visualized Comparison of Optimizers



Comparison of optimization algorithms
in contour maps of loss functions



Comparison of optimization
algorithms at the saddle point



Quiz

1. (Multiple choice) Which of the following optimizer can be used in deep learning? ()

- A. SGD
- B. Momentum
- C. RMSprop
- D. Adam

Contents

1. Propaedeutics of Deep Learning

2. Deep Learning Overview

- Definition and Development of Neural Networks
 - Perceptron and Training Rules
 - Activation Functions
 - Types of Neural Networks
 - Regularization in Deep Learning
- Optimizer
- Applications of Deep Learning



Applications of Deep Learning

Data
mining

Computer
vision



Deep learning

Natural
language
processing

Speech
recognition

Computer vision



Image
classification



Scene
recognition



Object detection
and recognition



OCR



Semantic
segmentation



Stylization



Super-resolution

.....

Image Classification

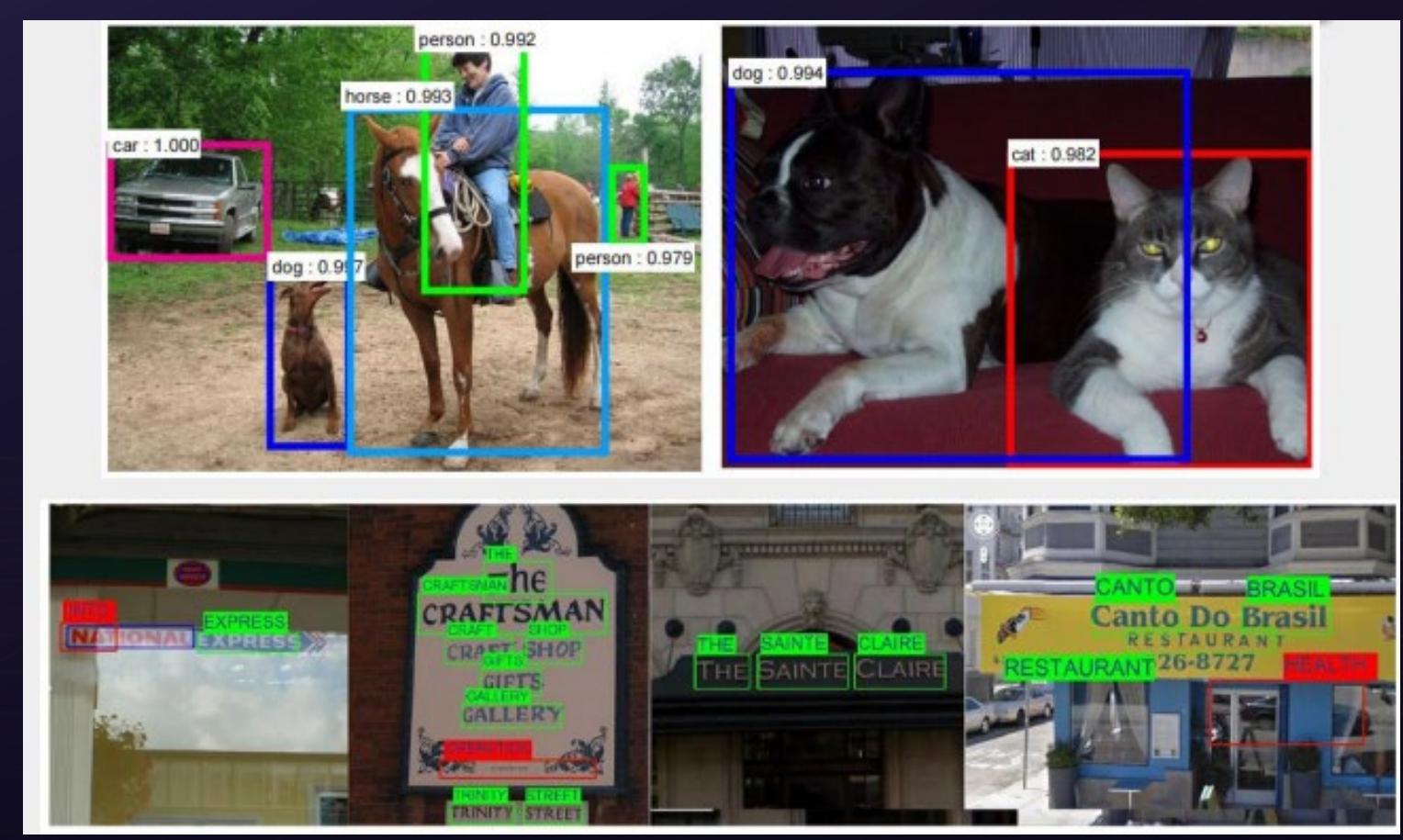


Scene Recognition



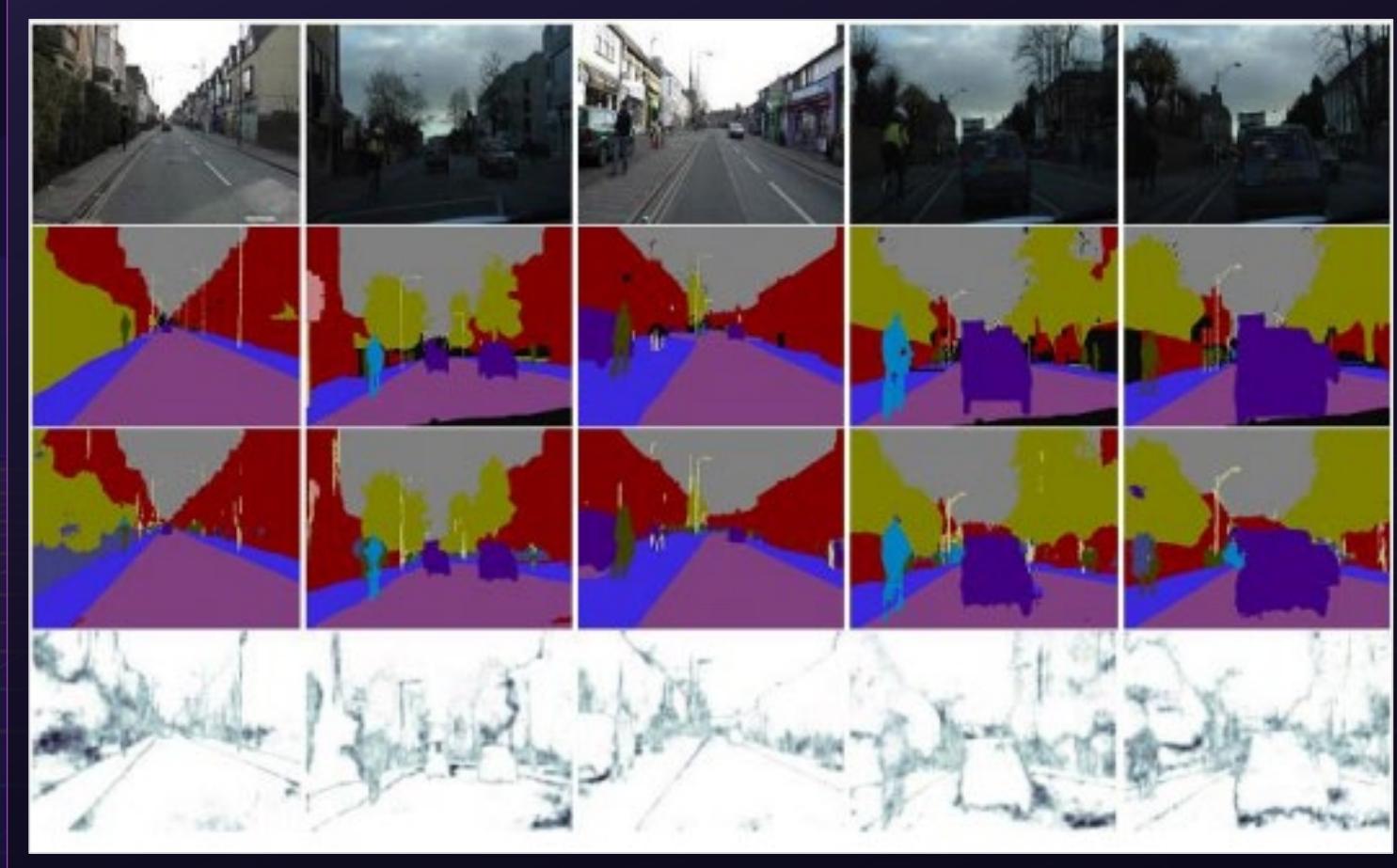


Object Detection and Recognition



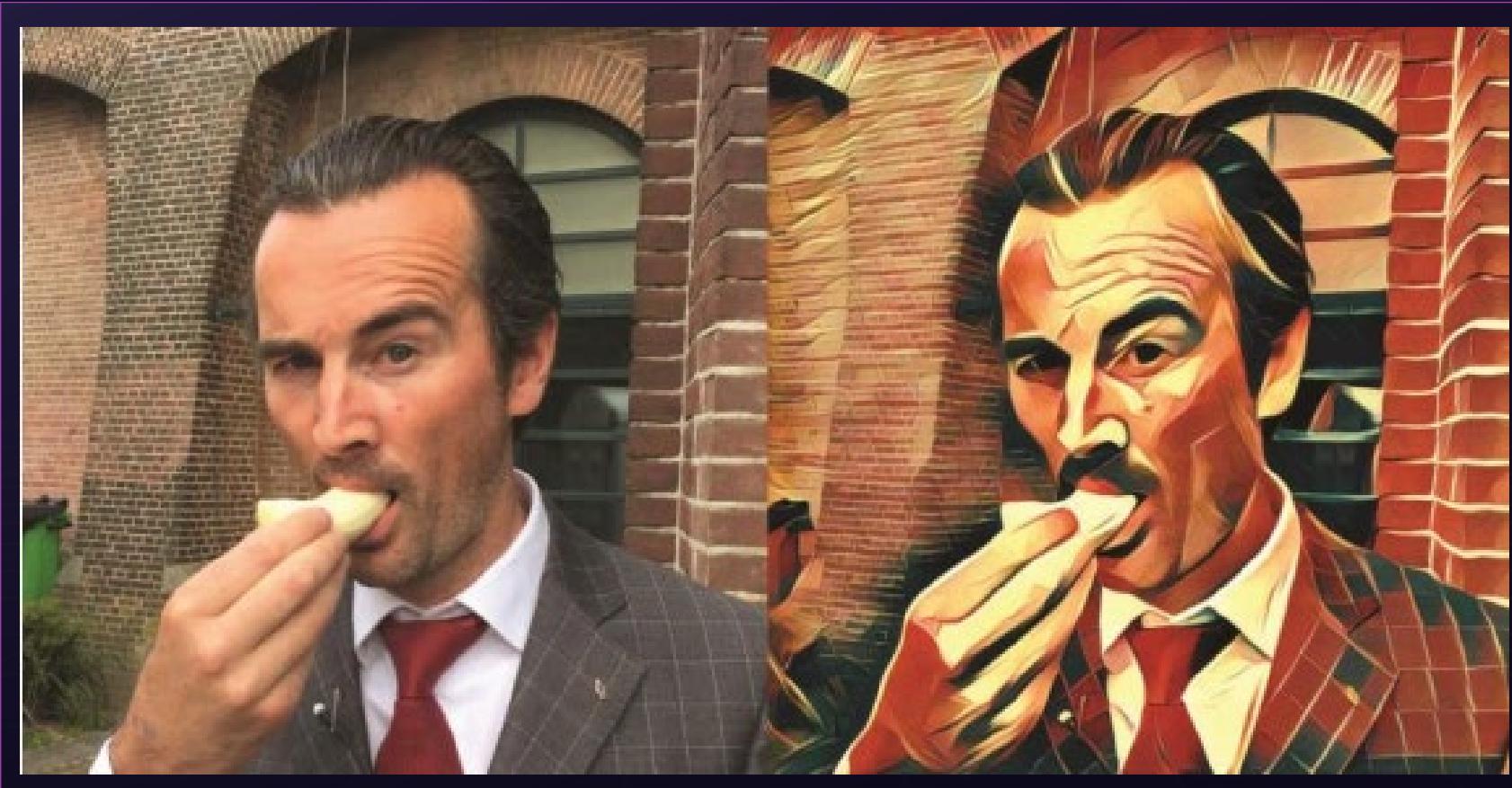


Semantic Segmentation



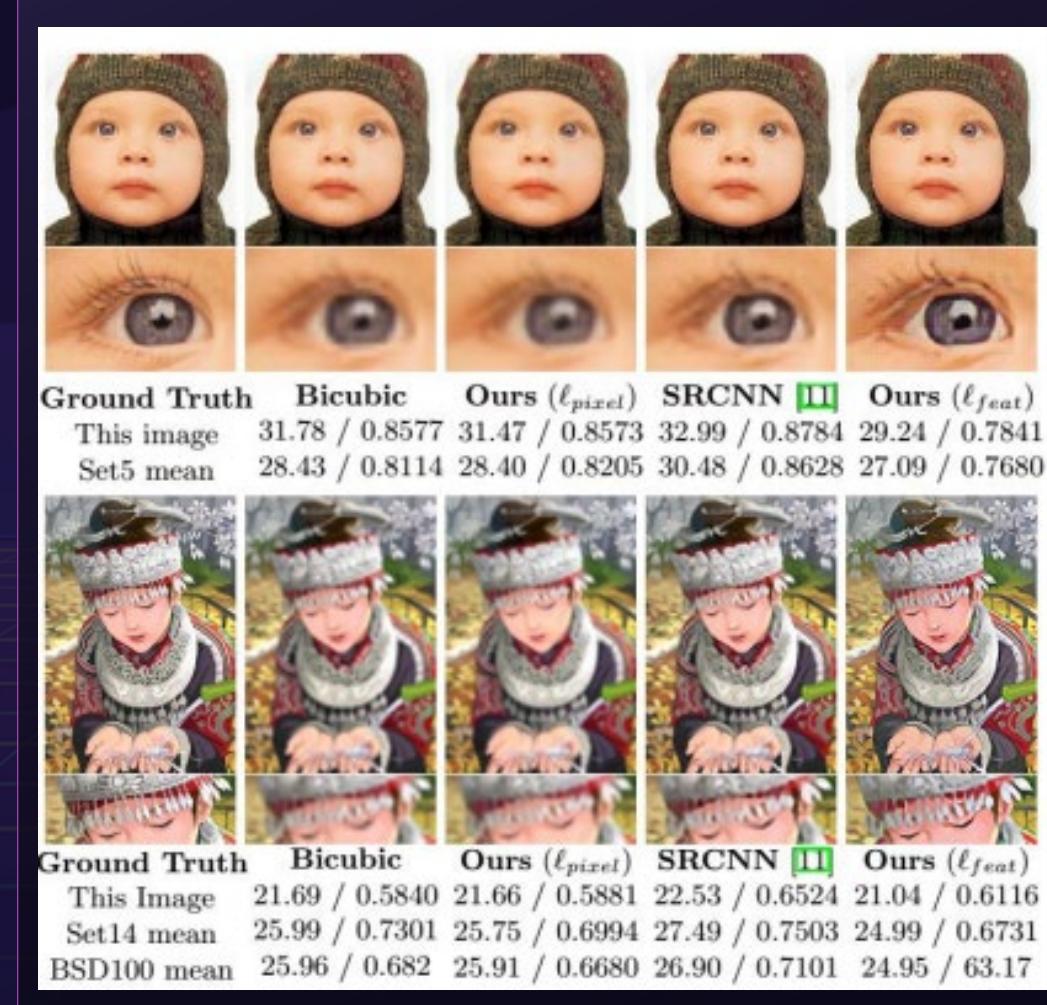


Stylization





Super-resolution





OCR

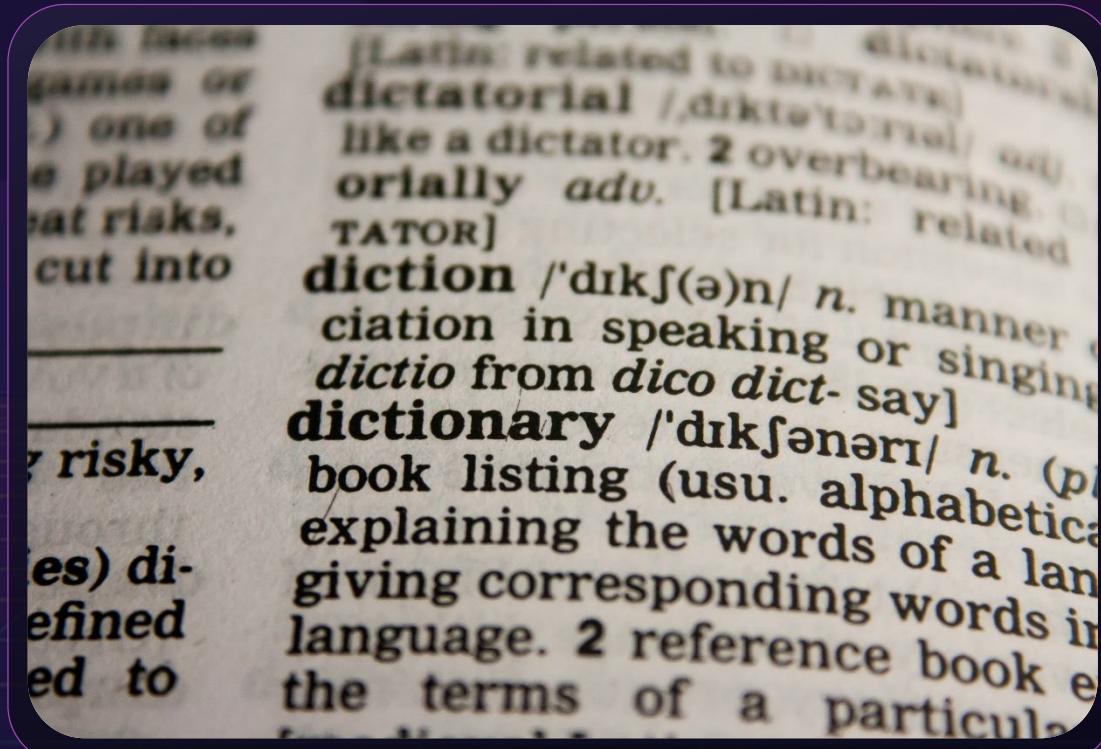




Natural Language Processing

word
segmentation

Knowledge
exploration



Machine
translation

Sentiment
analysis



Knowledge Exploration

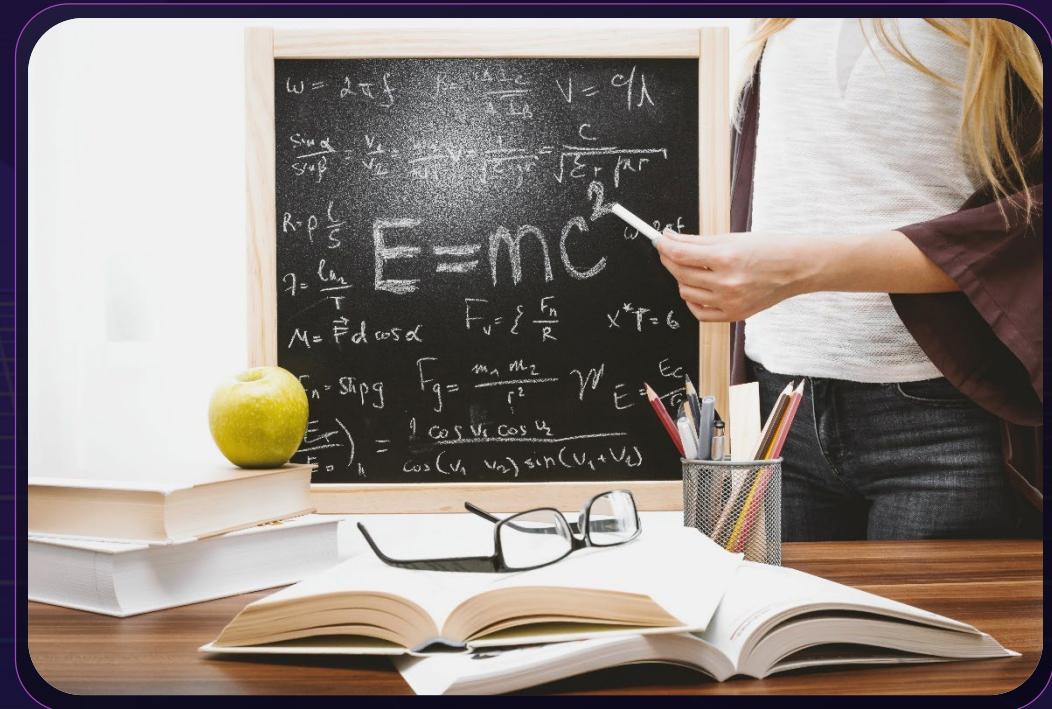
◆ Two types of problems

➤ New knowledge reasoning in the existing knowledge base

- CYC, WordNet, FreeNet, etc.
- Current studies use similar approaches.
 - Known entities are represented by word embeddings.
 - The entity relationship is modeled using tensor networks.
 - Backpropagation + SGD training

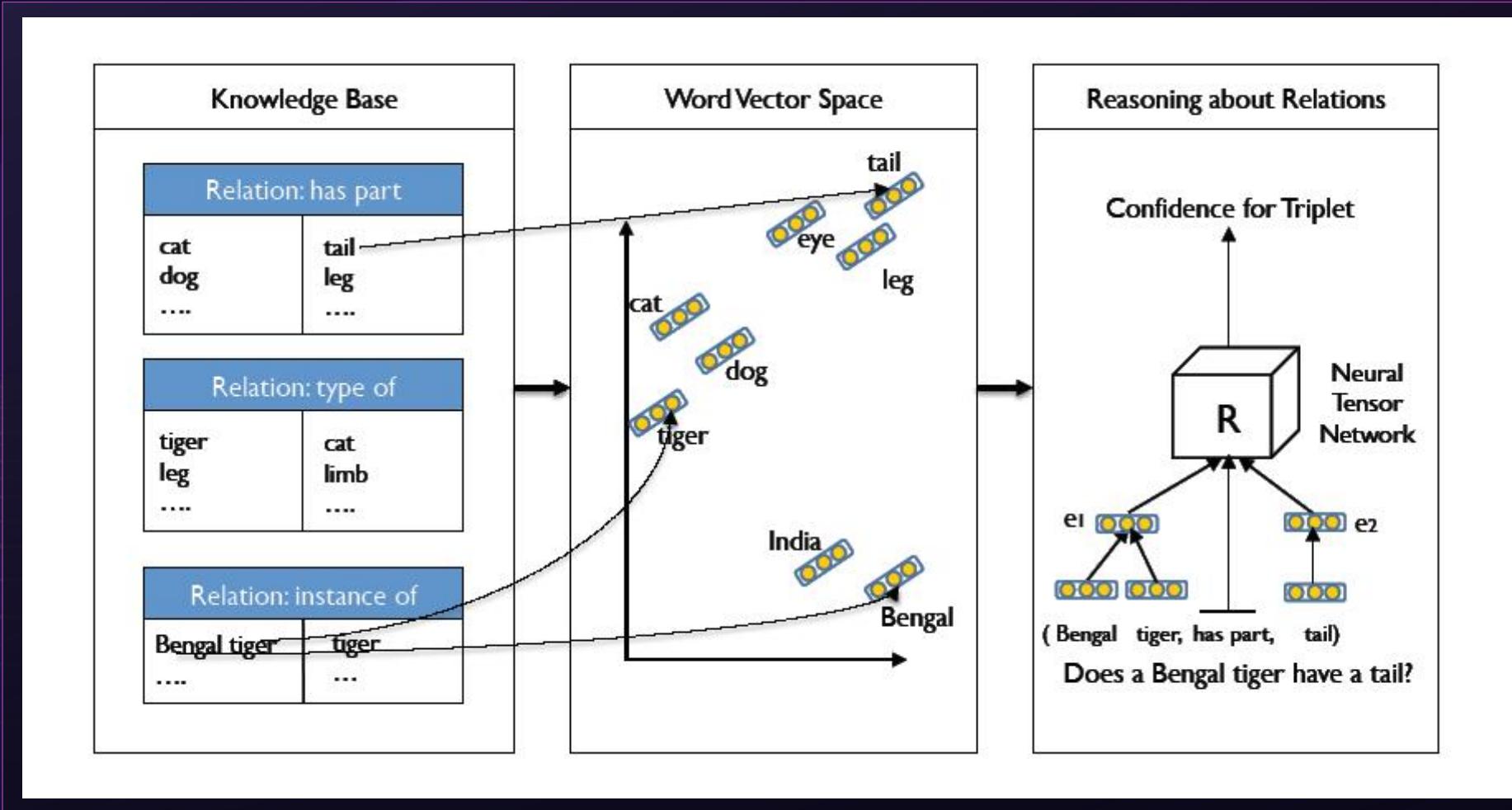
➤ Mining structured knowledge from free text

- Text mining
- TF-IDF

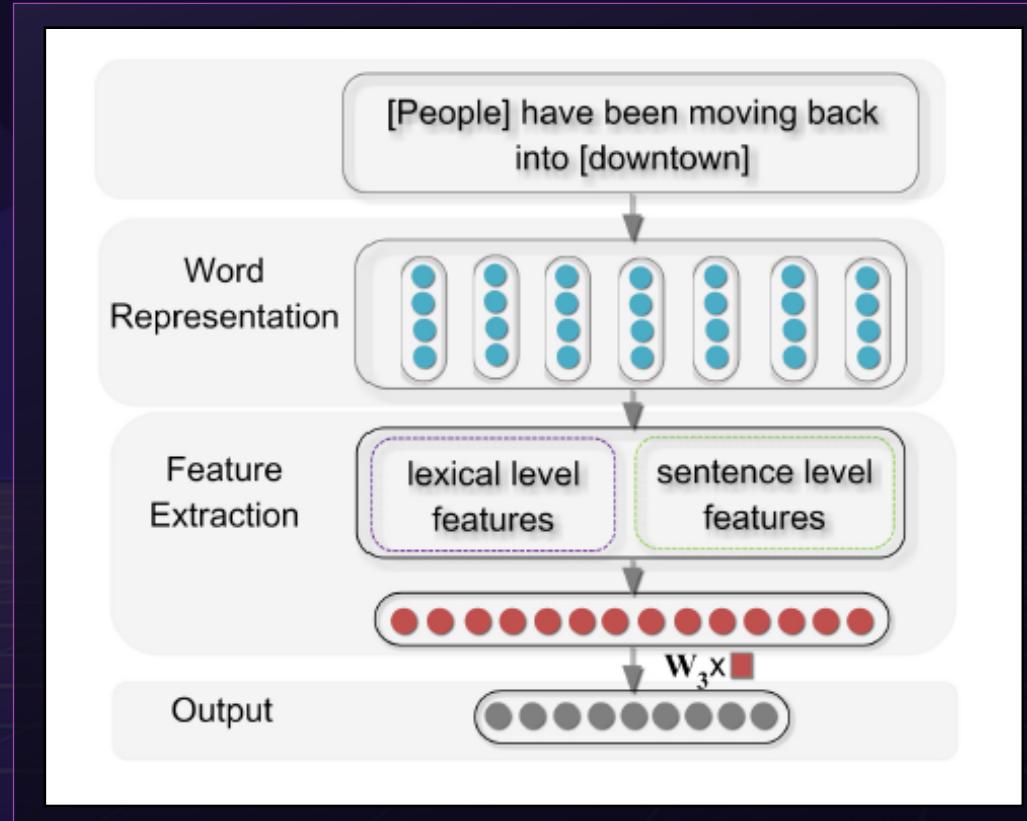




New Knowledge Reasoning in the Existing Knowledge Base



Mining Structured Knowledge from Free Text (1)



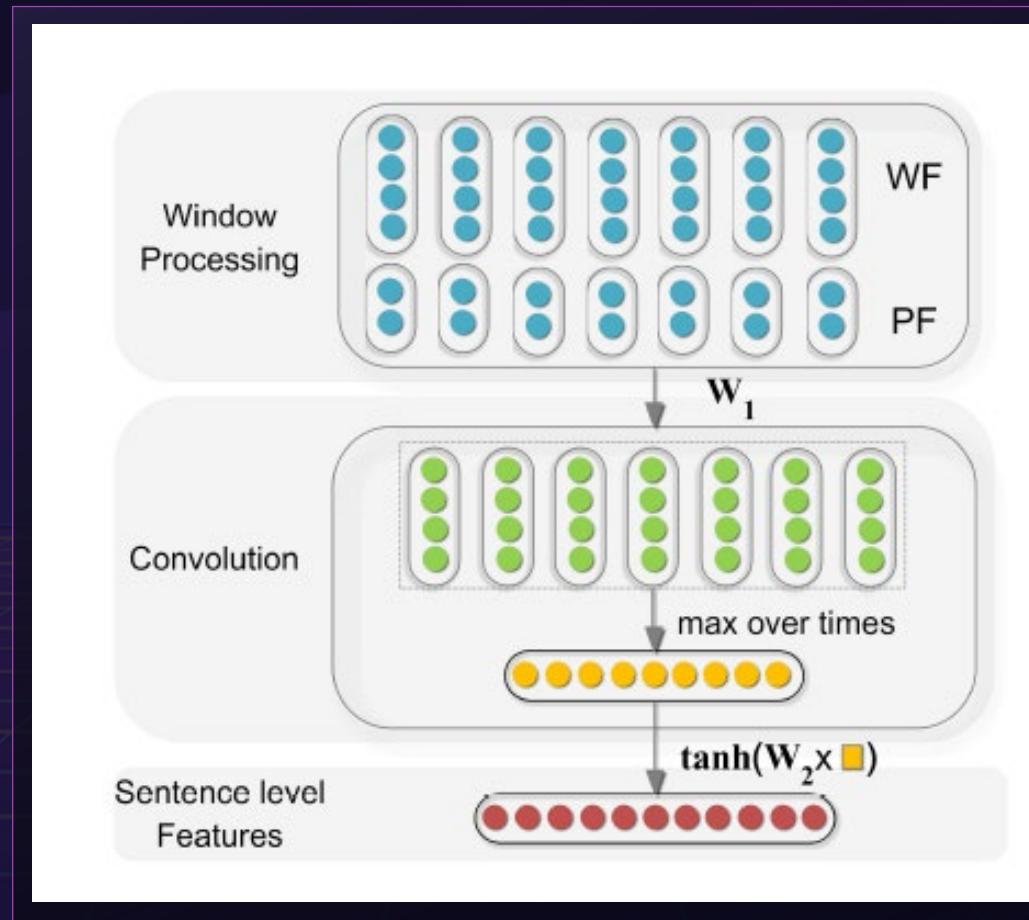
Overall structure

Features	Remark
L1	Noun 1
L2	Noun 2
L3	Left and right tokens of noun 1
L4	Left and right tokens of noun 2
L5	WordNet hypernyms of nouns

Lexical feature

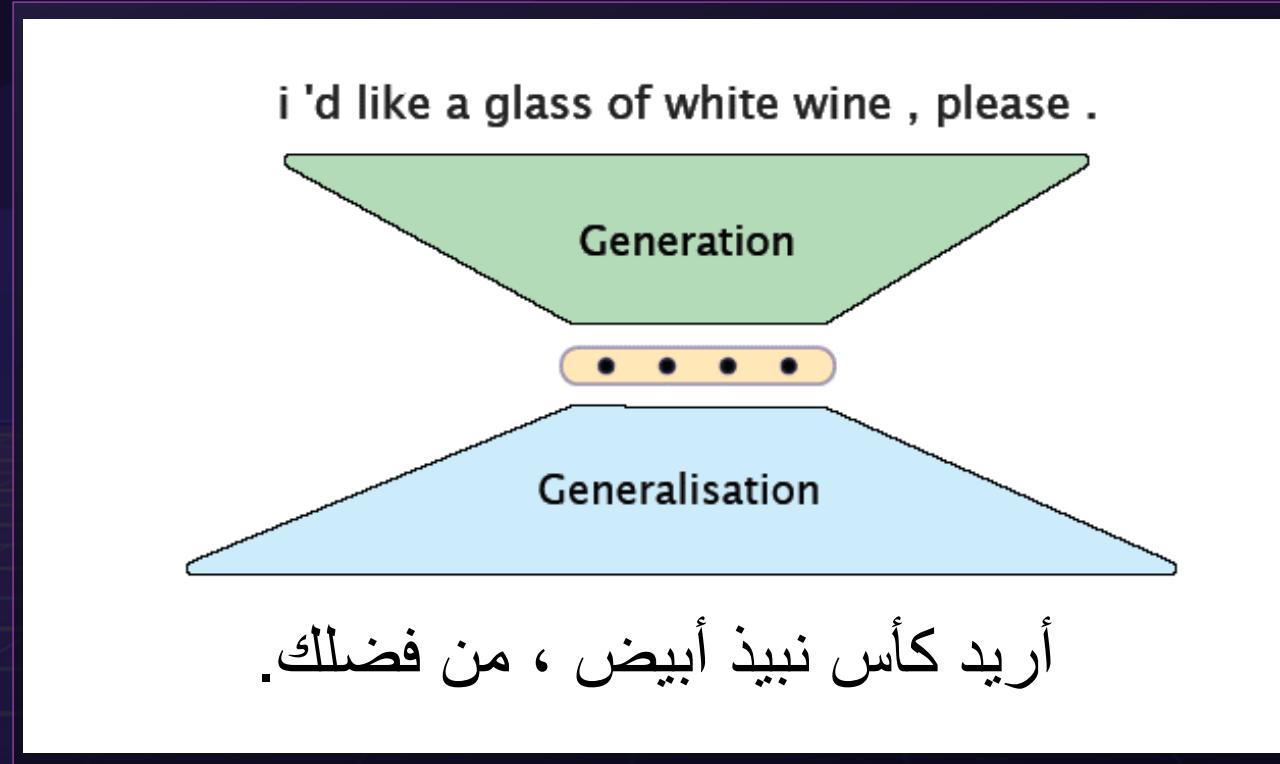


Mining Structured Knowledge from Free Text (2)



Sentence-level feature extraction: convolutional network

Machine Translation (Common Model)



Decoder
Semantic vector
Encoder

The most common model: encoder-decoder model

Sentiment Analysis

- ◆ Two core issues:
 - Sentence-level word embedding representation
 - How to encode emotional tendencies to word embeddings at all levels
 - Semi-supervised or supervised learning: The emotion tendency is encoded into the WE structure through the training process

What is Sentiment Analysis?

A linguistic analysis technique that identifies opinion early in a piece of text.

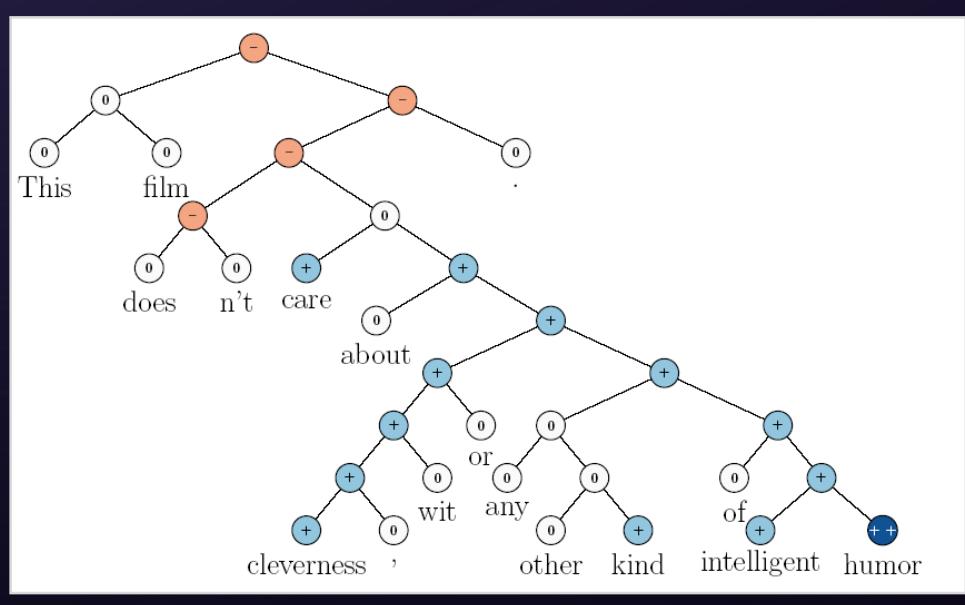
The movie is great.



The movie stars Mr. X

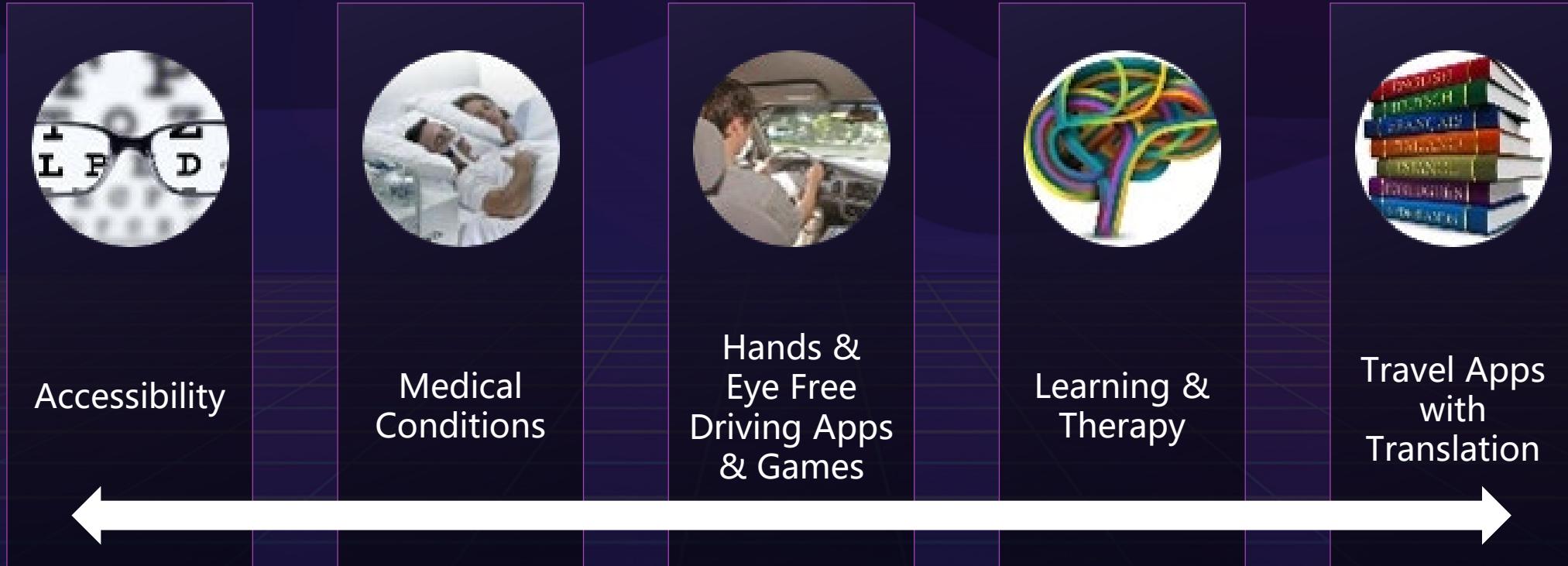


The movie is horrible.



Speech Recognition

Speech can have many roles in mobile apps





Recommendation System

Find Movies, TV shows, Celebrities and more... All

Spider-Man (2002)

PG-13 121 min - Action | Fantasy - 3 May 2002 (USA)

Your rating: ★★★★★★★★★★ 7.3

Ratings: 7.3/10 from 322,552 users Metascore: 73/100

Reviews: 1,976 user | 276 critic | 37 from Metacritic.com

When bitten by a genetically modified spider, a nerdy, shy, and awkward high school student gains spider-like abilities that he eventually must use to fight evil as a superhero after tragedy befalls his family.

Director: Sam Raimi

Writers: Stan Lee (Marvel comic book), Steve Ditko (Marvel comic book), 1 more credit »

Stars: Tobey Maguire, Willem Dafoe, Kirsten Dunst | See full cast and crew

+ Watchlist Watch Trailer Share...

Watch now At Amazon Instant Video Buy it at Amazon.com

Nominated for 2 Oscars. Another 12 wins & 44 nominations. See more awards »

Videos

Full Movie Trailer

104 photos | 10 videos | 7135 news articles »

Photos

People who liked this also liked...

Iron Man 3 (2013) PG-13 Action | Adventure | Sci-Fi

When Tony Stark's world is torn apart by a formidable terrorist called the Mandarin, he starts an odyssey of rebuilding and retribution.

Add to Watchlist

Director: Shane Black Stars: Robert Downey Jr., Gwyneth...

Cast

Cast overview, first billed only:

Tobey Maguire	... Spider-Man / Peter Parker
Willem Dafoe	... Green Goblin / Norman Osborn
Kirsten Dunst	... Mary Jane Watson

See all 7135 related articles »

User Lists Create a list »

Related lists from IMDb users

424 Sexy Pictures of Beautiful Mainstream Actresses

MY ULTIMATE BEST MOVIES: 2000-2013

10:18 PM 5/22/2013

Summary

This chapter introduces the propaedeutics of deep learning, including learning algorithms, common machine learning algorithms, hyperparameters, verification sets, parameter estimation, and estimation methods. Then it illustrates the definition, types, and development of neural networks as well as perceptron and its training rules.



1. (Single choice) Which of the following is not a deep learning development framework? ()

- A** CNTK
- B** Keras
- C** BAFA
- D** MXNet

2. (True or false) GAN is a deep learning model, which is one of the most promising methods for unsupervised learning of complex distribution in recent years. ()

- A** True
- B** False



Quiz

3. (Multiple answer question) Training errors reduce the model accuracy and generate underfitting. How can we improve the model fitting? ()

A Increasing data volume

B Feature engineering

C Reducing regularization parameters

D Adding features

4. (True or false) Compared with the recurrent neural network, the convolutional neural network is more suitable for image recognition. ()

A True

B False

Thanks

www.huawei.com





HUAWEI CLOUD EI

Overview

Contents

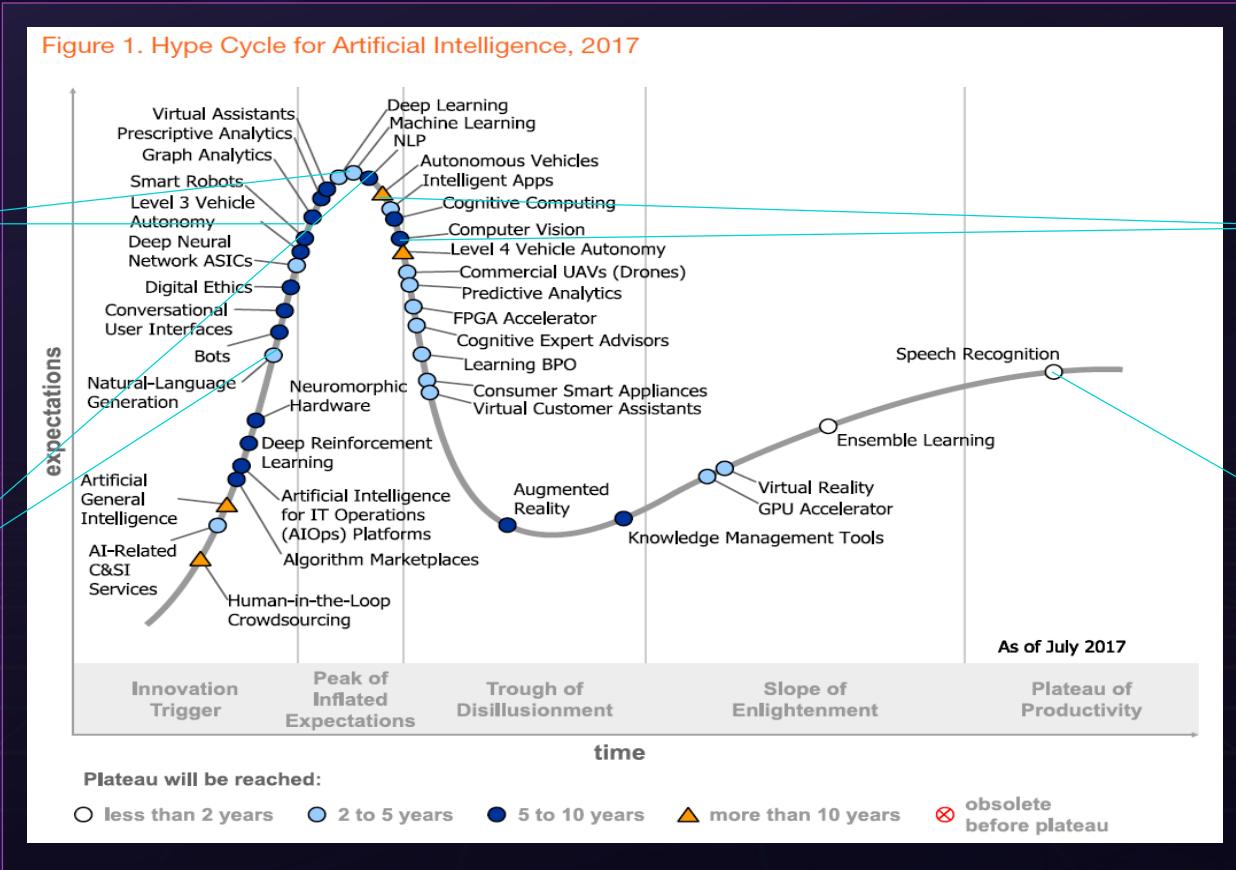
- 
- 1. Trend of AI and Origin of EI**
 - 2. Details About HUAWEI CLOUD EI**



Trend of Artificial Intelligence (AI) Technologies

The hype around machine learning will come to an end.

Natural language processing (NLP)



The computer vision technology has become mature.

Voice recognition takes the lead in commercial deployment.

Huawei AI Development Roadmap

Research on Basic Theories

- ◆ Noah's Ark Laboratory: AI
- ◆ Russell Laboratory: Big Data
- ◆ Norman Laboratory: map
- ◆ Shannon Laboratory: algorithms



Chip

Internal Practice

- ◆ GTS: intelligent site survey, intelligent review...
- ◆ Supply chain: intelligent loading, path optimization...
- ◆ Huawei Device: intelligent album, risk control...
- ◆ Process and IT: intelligent O&M and machine translation



Terminal

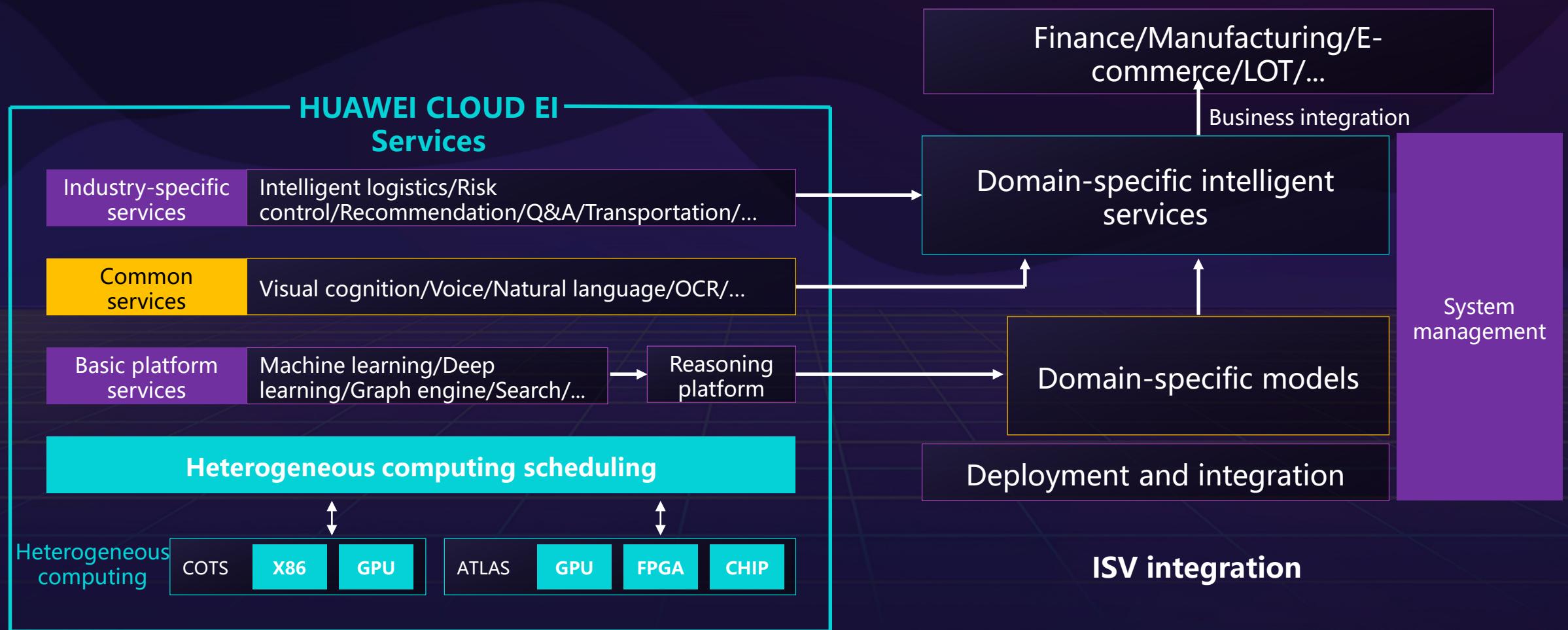
External Enablement (EI)

- ◆ Heterogeneous computing: CPU, GPU, and dedicated AI chip
- ◆ General service: vision and voice
- ◆ Basic platform: machine learning, deep learning, graph computing
- ◆ Industry enablement: intelligent logistics, intelligent Q&A, and intelligent recommendation



Cloud

HUAWEI CLOUD EI: Making Enterprises Smarter



Global Practice of HUAWEI CLOUD EI

60%

Top 10 financial enterprises in China

25%

Top 50 carriers around the world

30%

Construction of Safe Cities in China



Contents

1. Concept of AI and Origin of EI

2. Details About HUAWEI CLOUD EI

- Basic Platform Services
- Common Services
- Industry-specific Services



HUAWEI CLOUD EI Service Panorama

Solutions

Enterprise Intelligence

Partners

Support

Essential Platform

Machine Learning Service
Deep Learning Service
Deep Learning HMI
Graph Engine Service

Big Data

Data Ingestion Service
Cloud Data Migration
Cloud Stream Service
MapReduce Service
Data Lake Insight
CloudTable Service
Data Warehouse Service
Cloud Search Service
Data Lake Factory

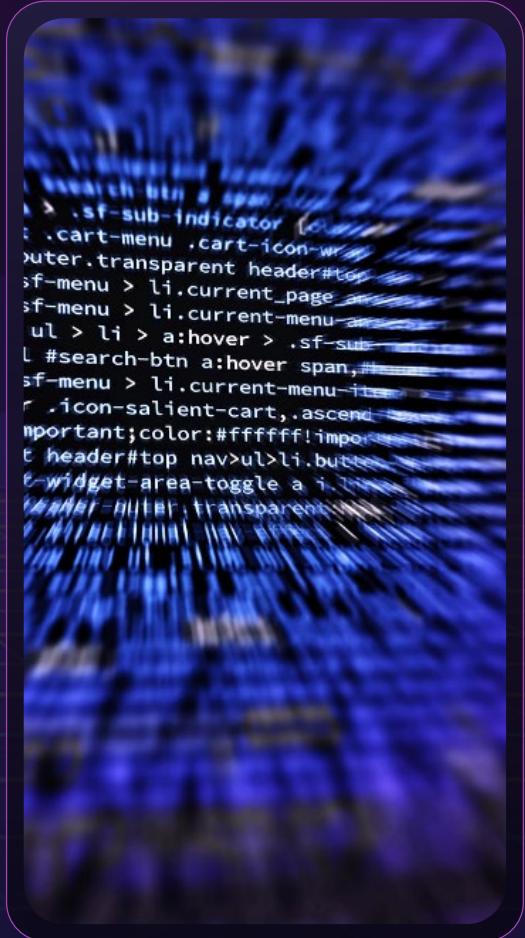
Visual Cognition

Optical Character Recognition
Face Recognition
Image Recognition
Content Moderation



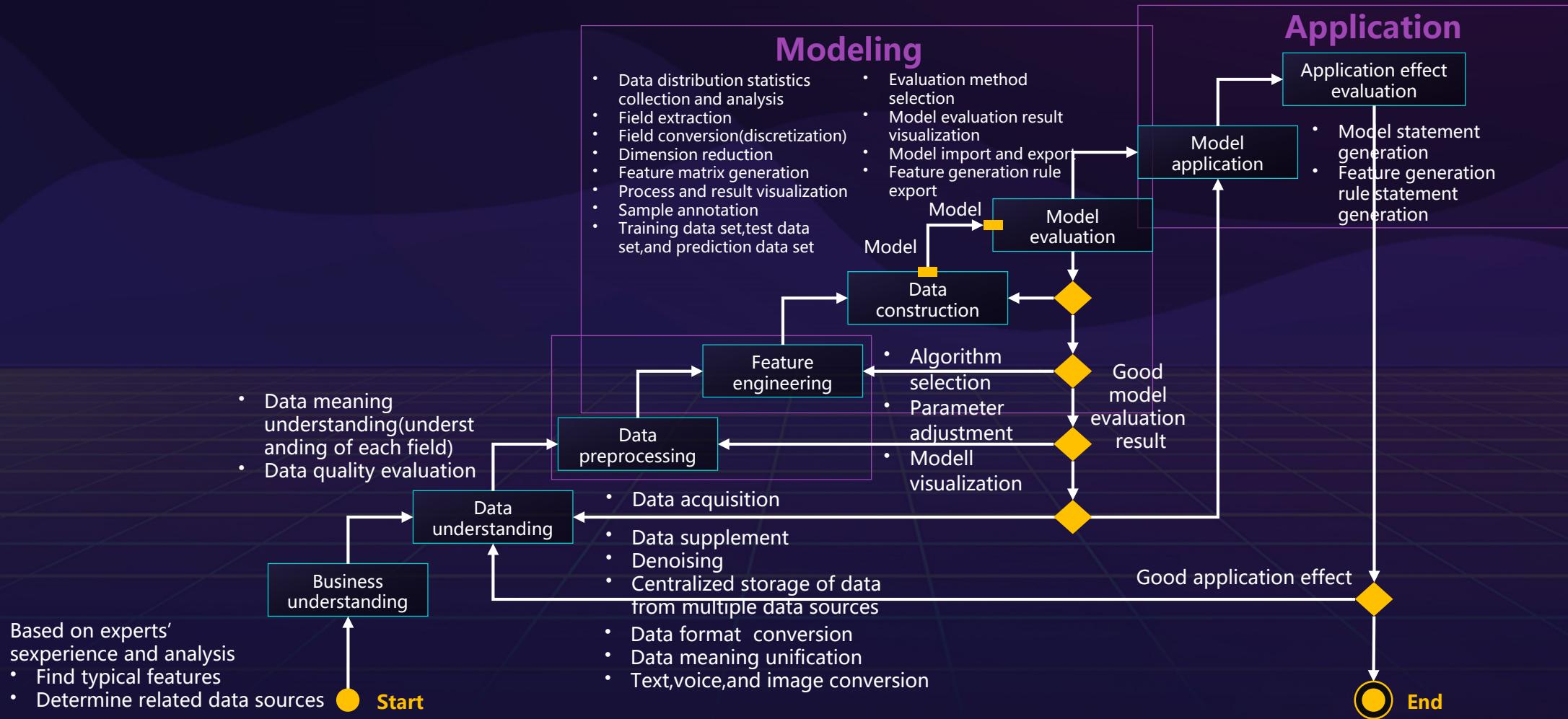
Basic Platform Services

- ◆ Machine Learning Service (MLS)
- ◆ Deep Learning Service (DLS)
- ◆ Graph Engine Service (GES)
- ◆ Deep Learning HMI
- ◆ Batch Service



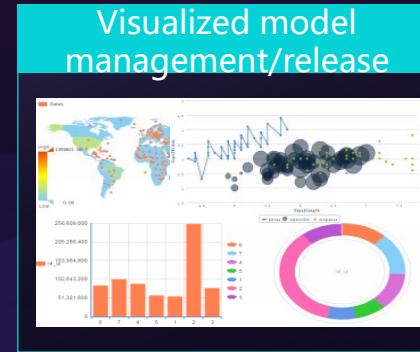
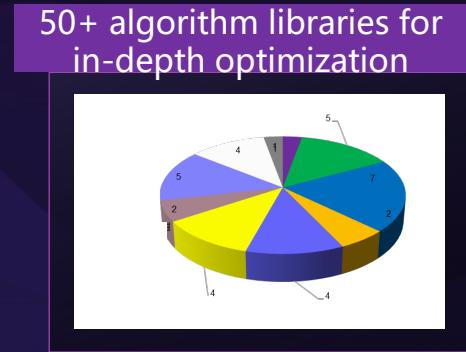
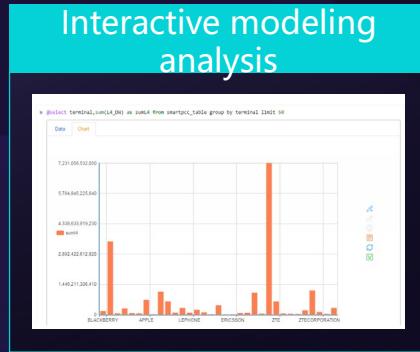
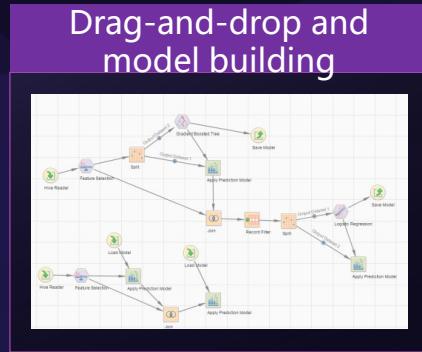


MLS: Converting from Features to Model Applications

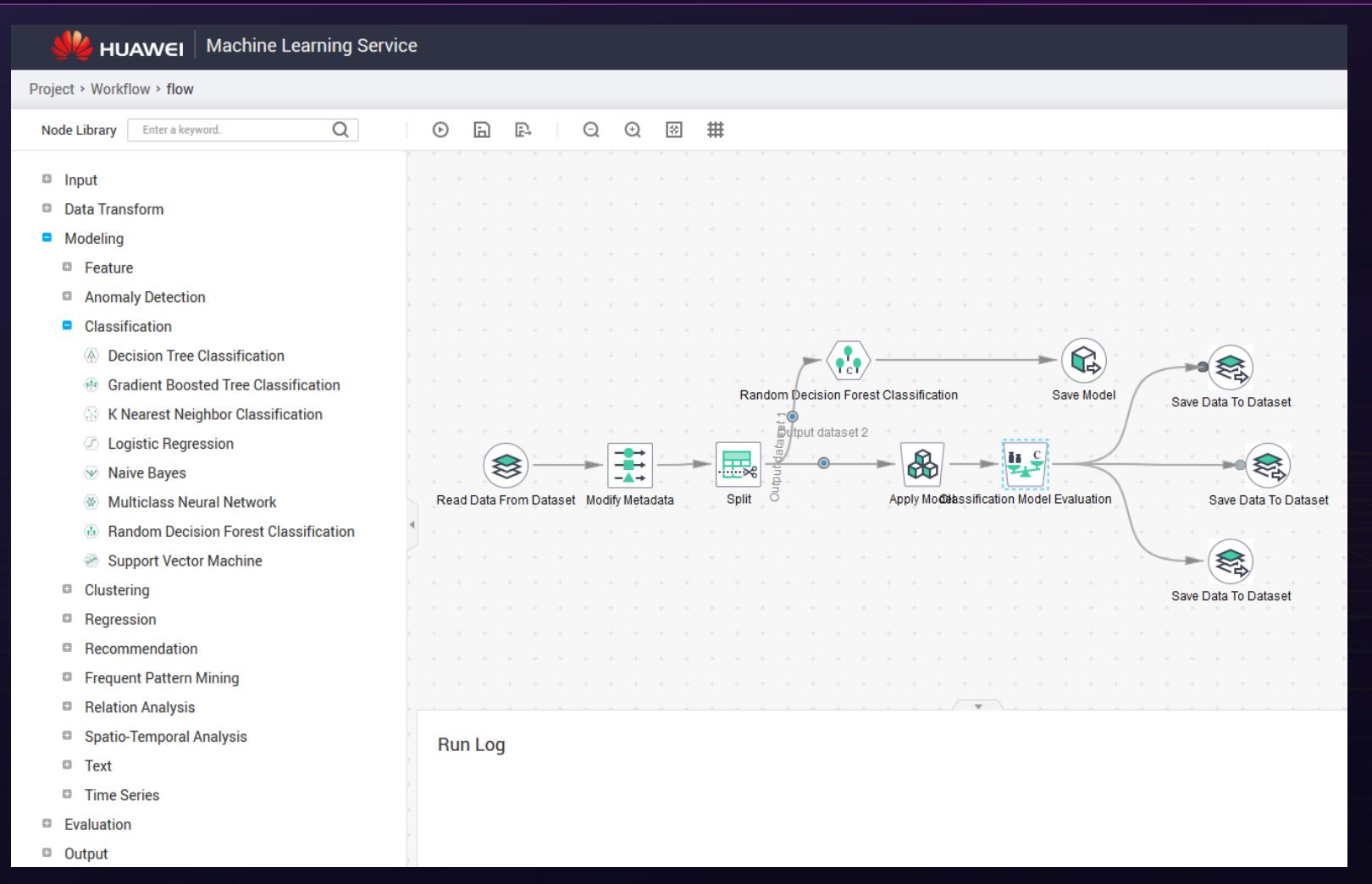




MLS: One-Stop Platform That Supports the Entire Data Analysis Process



MLS: Drag-and-Drop and Model Building



MLS Success Stories

Application recommendation

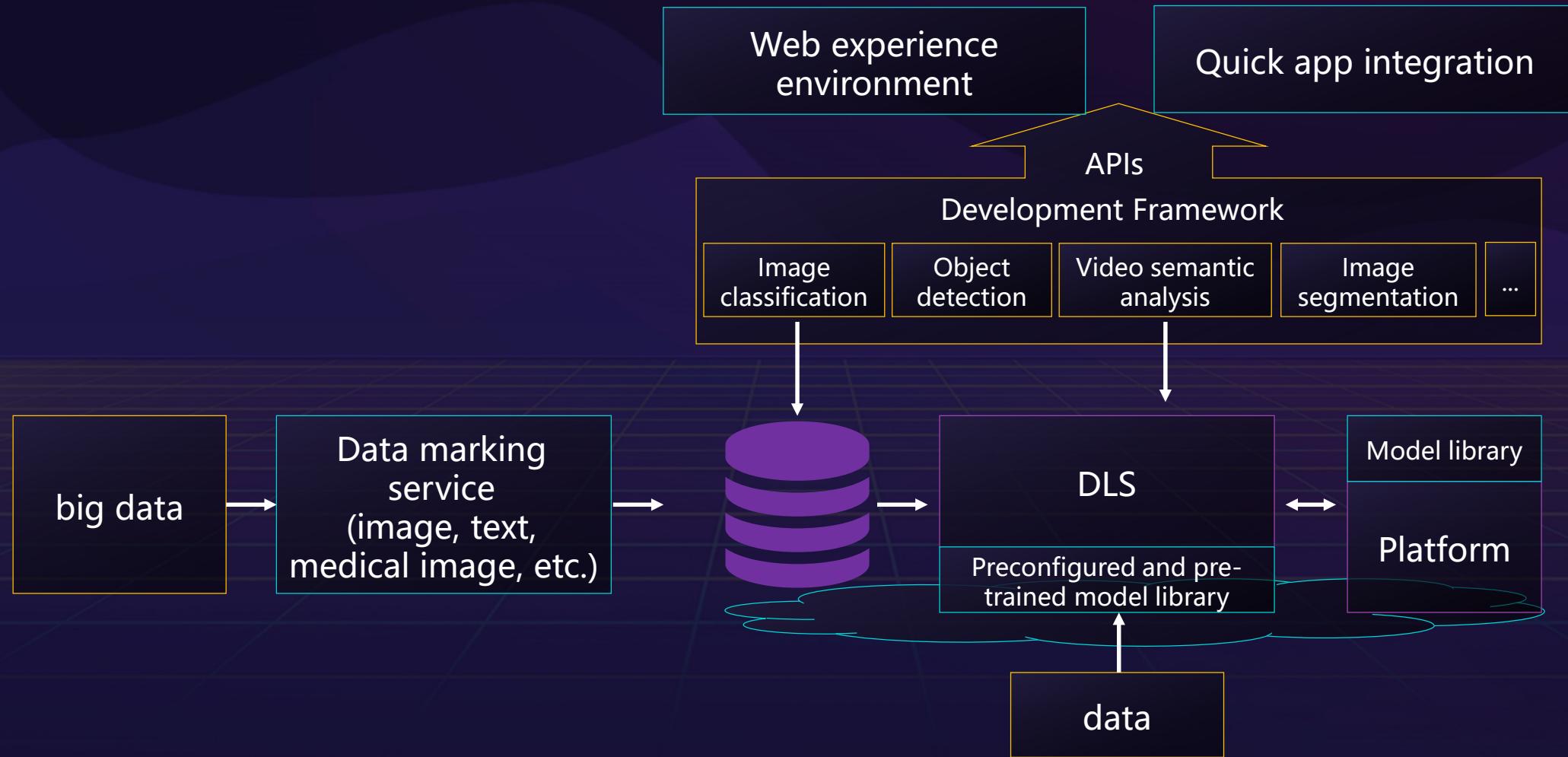


IoV analysis





DLS: Preconfigured Model Development Framework, Facilitating Service Development Through Deep Learning





DLS Success Stories (1)

Quality image identification object model

EHS: 15 intelligent object models
(gradually expanded to about 50 models)

Person	Safety helmet	Warning board
Gloves	Warning board	First-aid kit
Safety helmet	Pulley	...

TE: 25 intelligent object models (gradually expanding)

Cabinet	Battery	DCDU
BBU	RRU	Antenna
GPS	Waterproof	...

OSP, CW, and energy
(to be expanded)

Manual image annotation

EHS: 15 types of recognition objects, annotating 6000+ images (first partner: Adecco)

Scenario-specific rules

EHS intelligent review rules

No.	Object to Be Annotated	Standard definition	Business Review Rule	Accuracy
1	Person	person	There is one person.	> 80%
2	Safety helmet	helmet	There is one safety helmet.	> 80%
3	Head + safety helmet	helmet_on_head	Every person has a safety helmet.	> 80%
4	Fluorescent clothing	high_visible_vest	Every person has fluorescent clothing.	> 80%
5	Gloves	glove	There are two gloves.	> 70%
6	Rigger	person	There are two hooks.	> 80%
		hook		> 80%
7	First-aid kit	first_aid_box	There is a first-aid kit.	> 80%
8	Fire extinguisher	extinguisher	There is a fire extinguisher.	> 80%
9	Warning board	warning_board	There is a warning board.	> 80%
10	Warning sign	warning_sign	There is at least one warning sign.	> 80%
11	Warning tape	warning_line	There are some warning tapes.	> 80%
12	Toolkit	toolkit	There is a toolkit.	> 80%
13	Safety shoes	safety_shoe	Every person has two safety shoes.	> 70%
14	Pulley	pulley	There is a pulley.	> 80%
15	Insulation gloves	insulated_glove	There are two insulation gloves.	> 70%

TE, OSP, CW, and
energy rules (to be
expanded)

Model training

Recognized:
1. Person
2. Fluorescent vest
3. Safety helmet

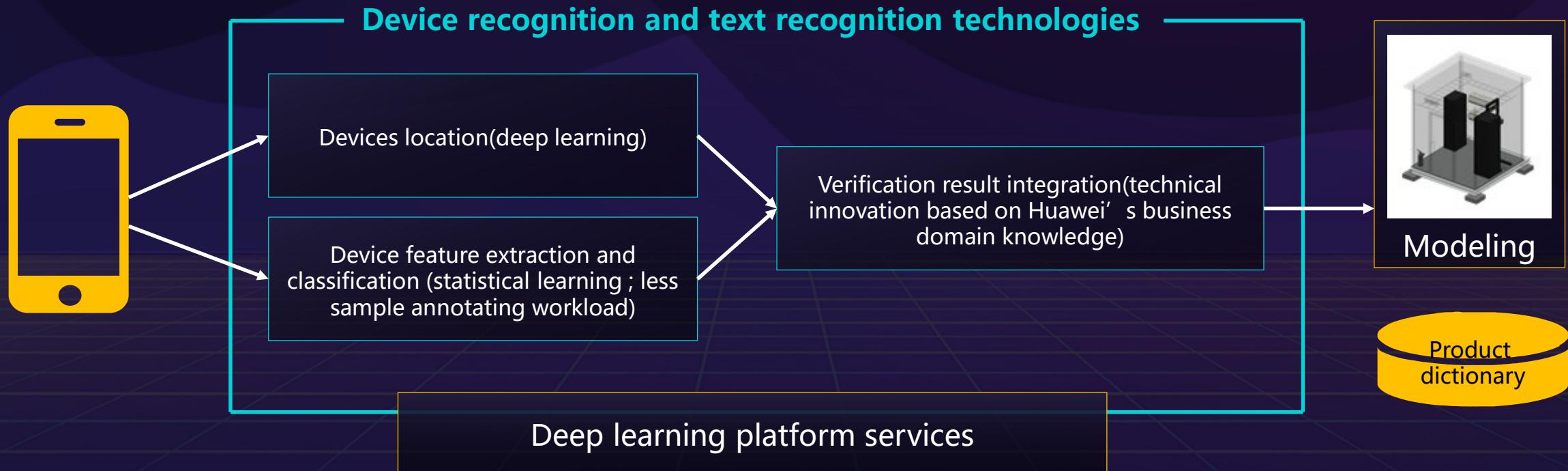
Recognized:
1. Warning board
2. 12 warning signs

No.	Recognition Item	Accuracy	Recall Rate
1	Person	100%	98.60%
2	Warning board	100%	98.36%
3	Fire extinguisher	95.24%	97.56%
4	Fluorescent vest	100%	97%
5	Helmet on head	100%	93.30%
6	Warning tape	100%	92.85%
7	First-aid kit	100%	90%
8	Warning sign	99.75%	88.89%
9	Toolkit	92.83%	76.47%
10	Helmet	97.70%	69.40%
11	Hook	85.18%	46.94%
12	Glove	96%	41.94%
All		98.71%	84.91%

Copyright © 2019 Huawei Technologies Co., Ltd. All rights reserved.

 HUAWEI

DLS Success Stories (2)



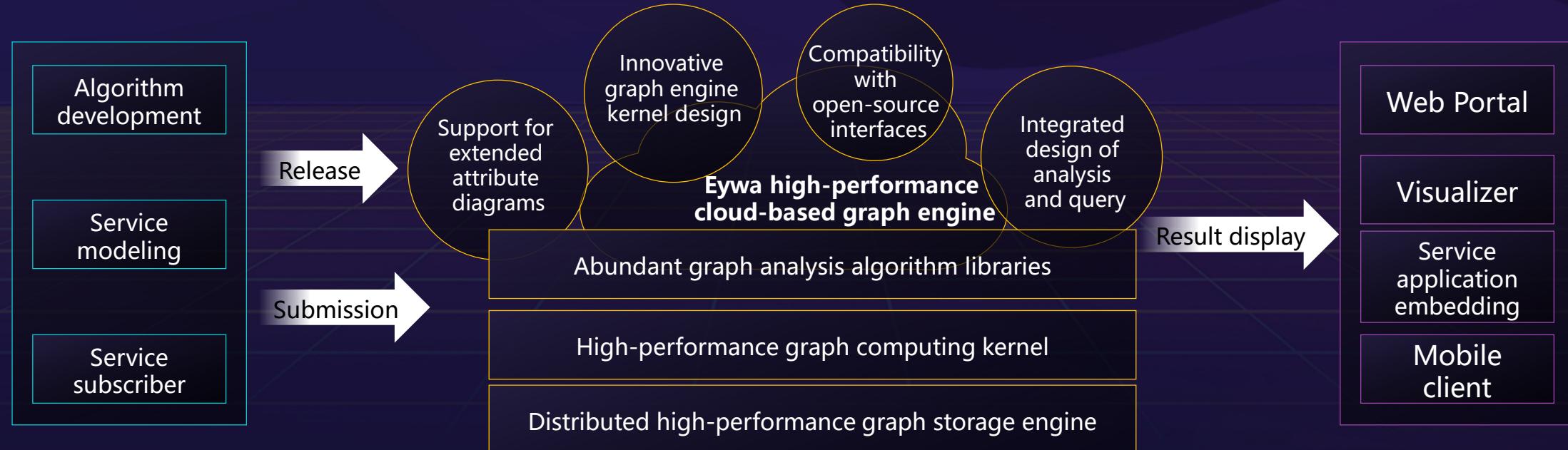


GES: Hyper-Scale and Integrated Graph Analysis and Query

GES provides integrated platform capabilities, such as associated data query and relationship analysis, and related service applications. It features **large scale, high concurrency, and low latency**.

Second-level query response for graphs with **1 billion** nodes and **over 100 billion** edges

GES supports **graphs with standard attributes** and extended graphs, provides powerful model description capabilities, and is compatible with mainstream **graph computing frameworks and graph query interfaces**.



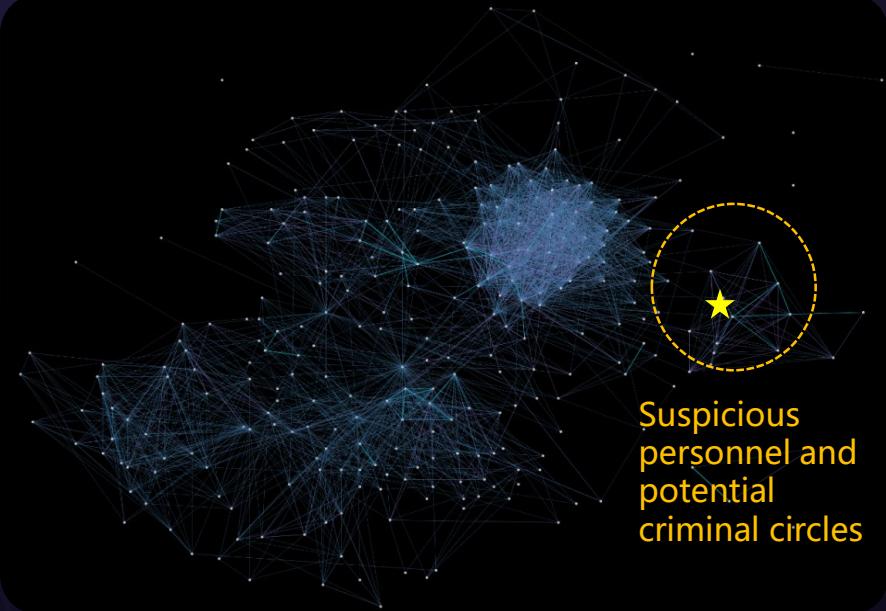
GES Success Stories

Route planning



Duration for planning hundreds of thousands of routes: 6.8 hours -> minutes

Vehicle warranty fraud prevention



Hundreds of millions of nodes and billions of edges: saving US\$N for enterprises

Contents

1. Concept of AI and Origin of EI

2. Details About HUAWEI CLOUD EI

- Basic Platform Services
- Common Services
- Industry-specific Services



EI Visual Cognition



Image Recognition Service



Optical Character Recognition Service



Content Moderation Service



Deblur Service

EI Voice and Semantics



Machine Translation Service



Text To Speech Service



Automatic Speech Recognition Service



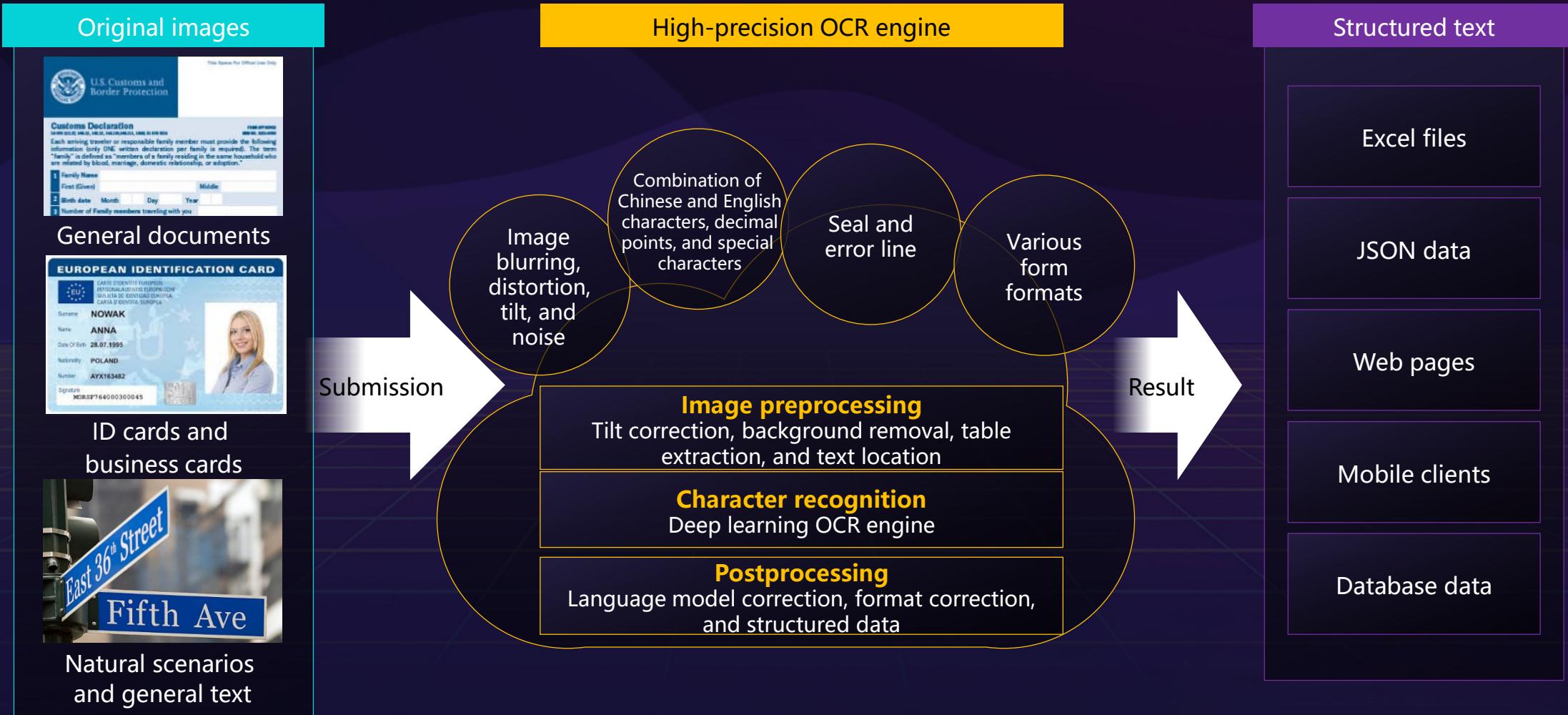
Natural Language Processing Service



Conversational Bot Service



OCR: High-Precision, Optical, and Automatic Character Recognition Capability





Moderation: Automatic Content Detection



Porn identification
for text and images



Sensitive political
information identification



Violence and terror
identification



Deblur: Dark Enhance and Defog

Low-illumination
image



Multidimensional histogram
equalization algorithm

vs

Huawei



Dark Enhance

Brightness disorder error rate: < 3%

Foggy
image



ECCV 2016

vs

Huawei



Defog

Actual luminance retention

Contents

1. Concept of AI and Origin of EI

2. Details About HUAWEI CLOUD EI

- Basic Platform Services
- Common Services
- Industry-specific Services



EI Industry-specific Services (1)



**Intelligent Logistics
Service**

**Intelligent
manufacturing Service**

**Intelligent
Finance Service**

**Intelligent
Retail Service**



EI Industry-specific Services (2)



**Intelligent
Power Service**

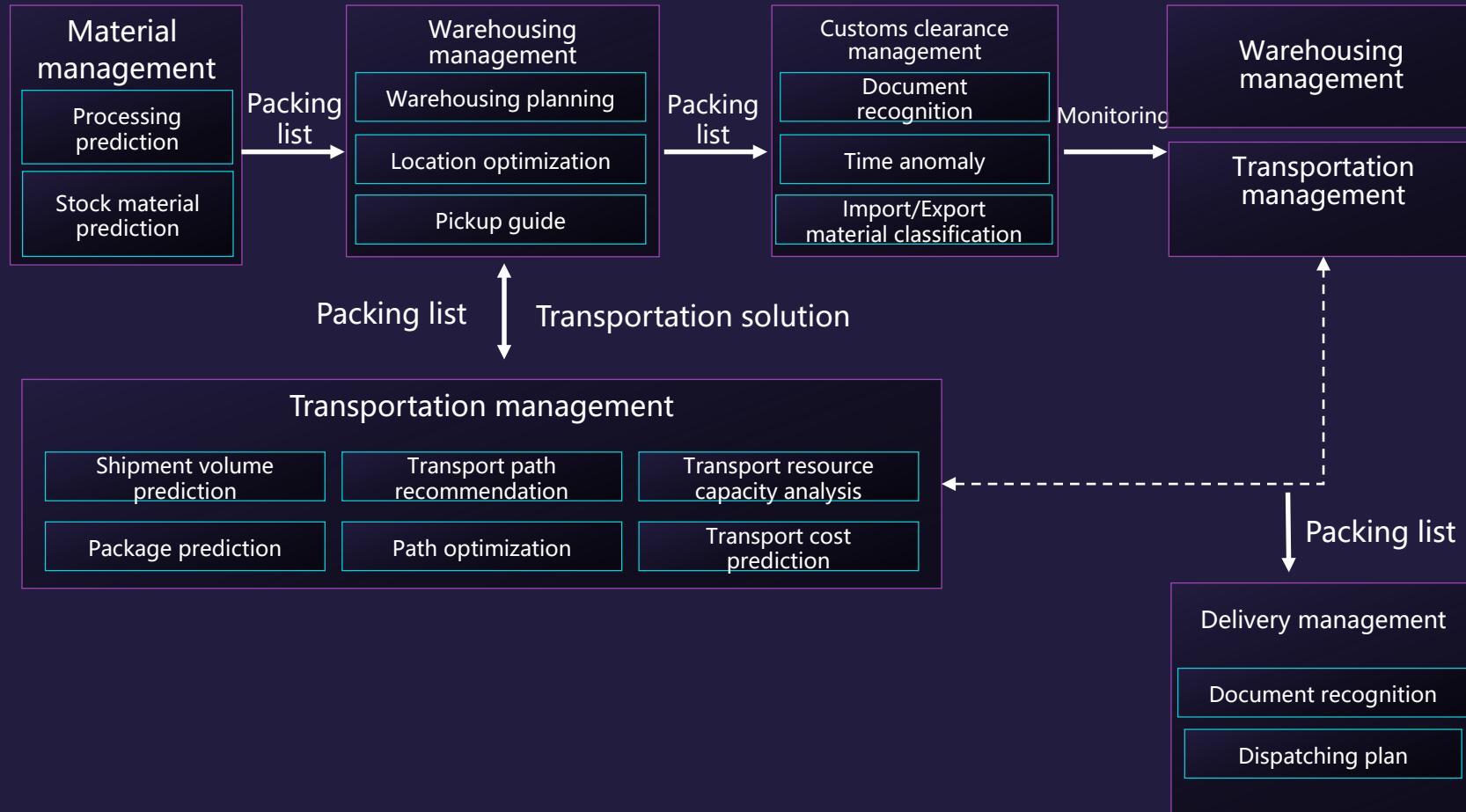


**Intelligent
Transport Service**

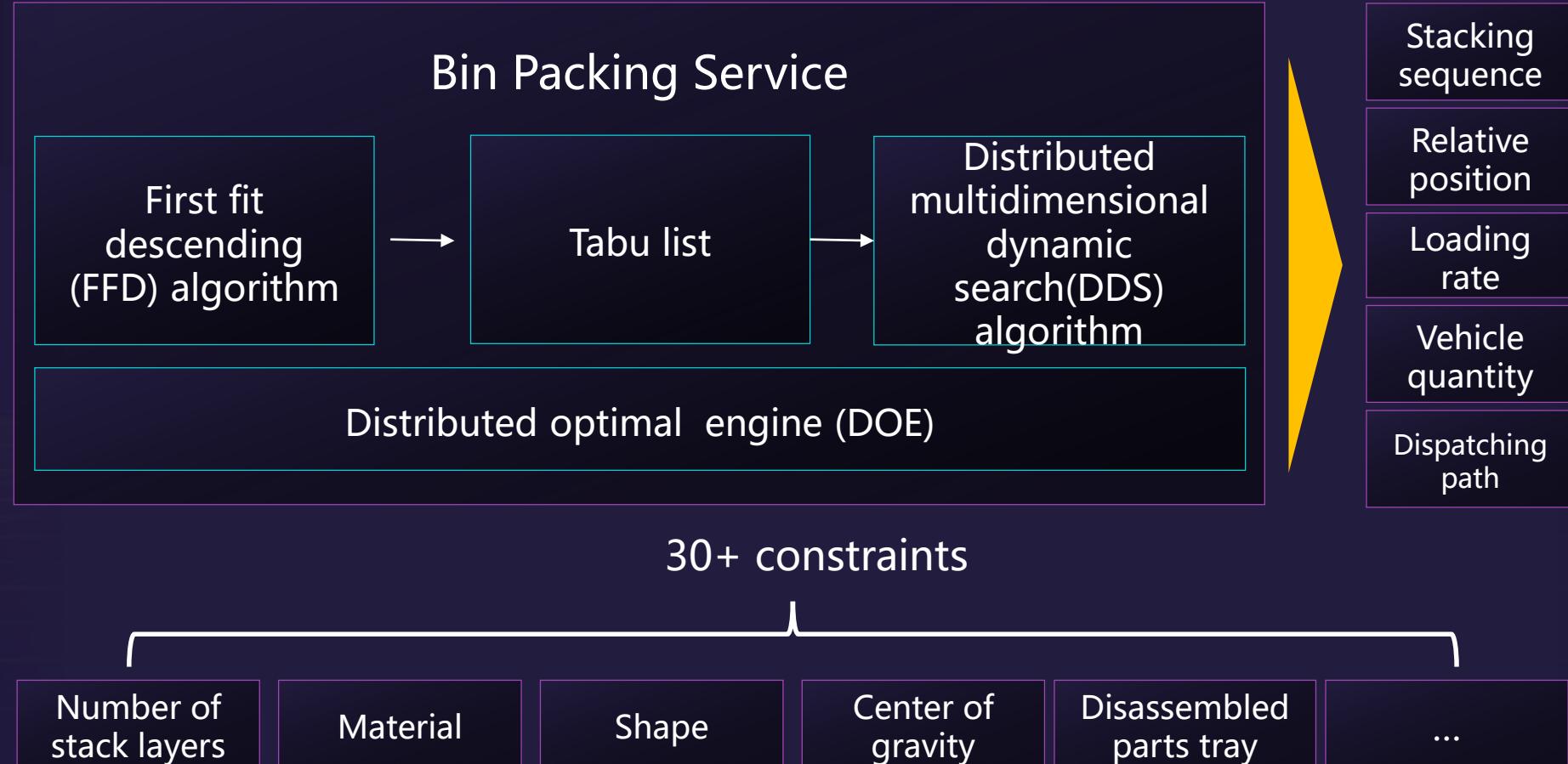


**Intelligent
Water Service**

Panorama of the Intelligent Logistics Solution

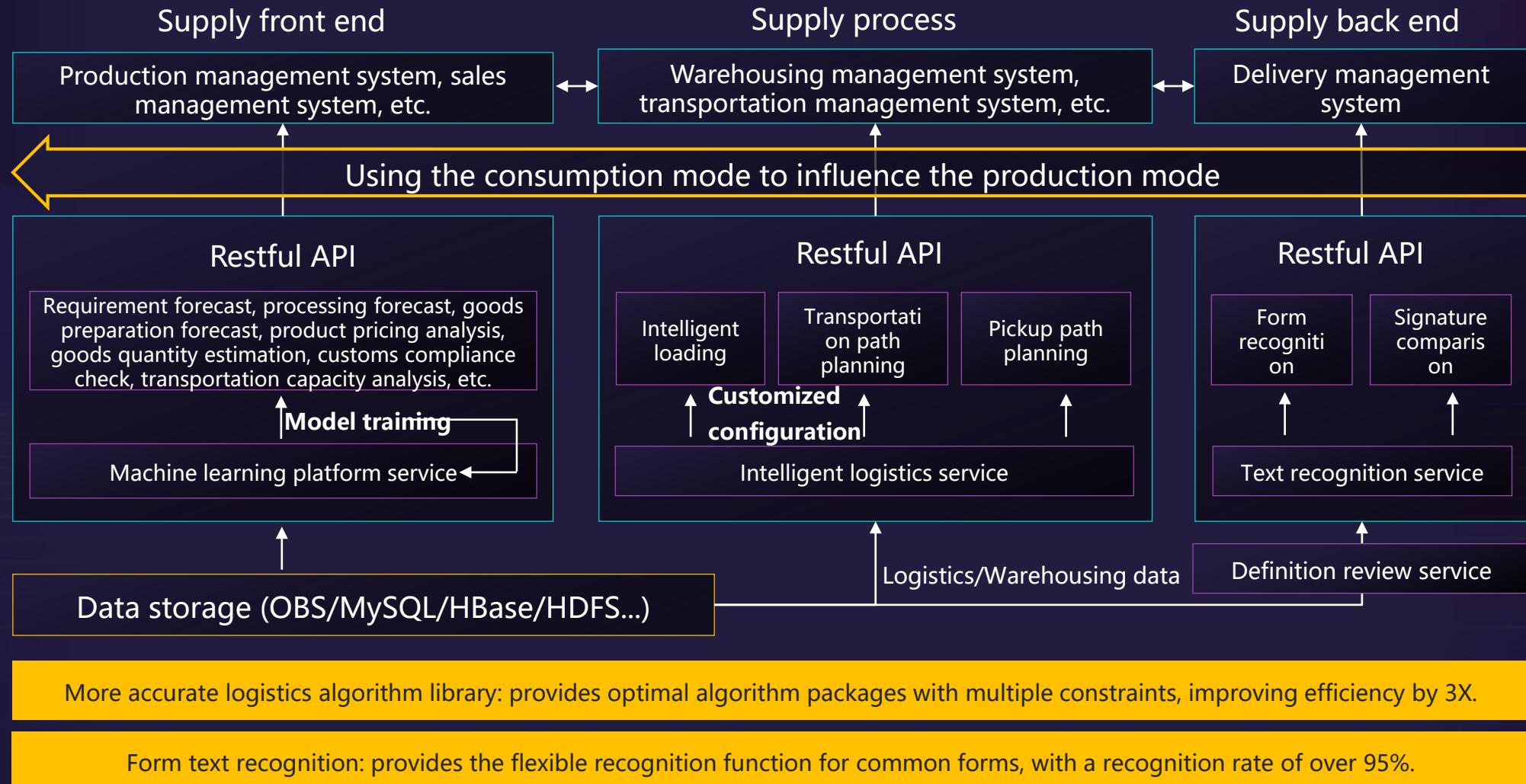


Bin Packing Service

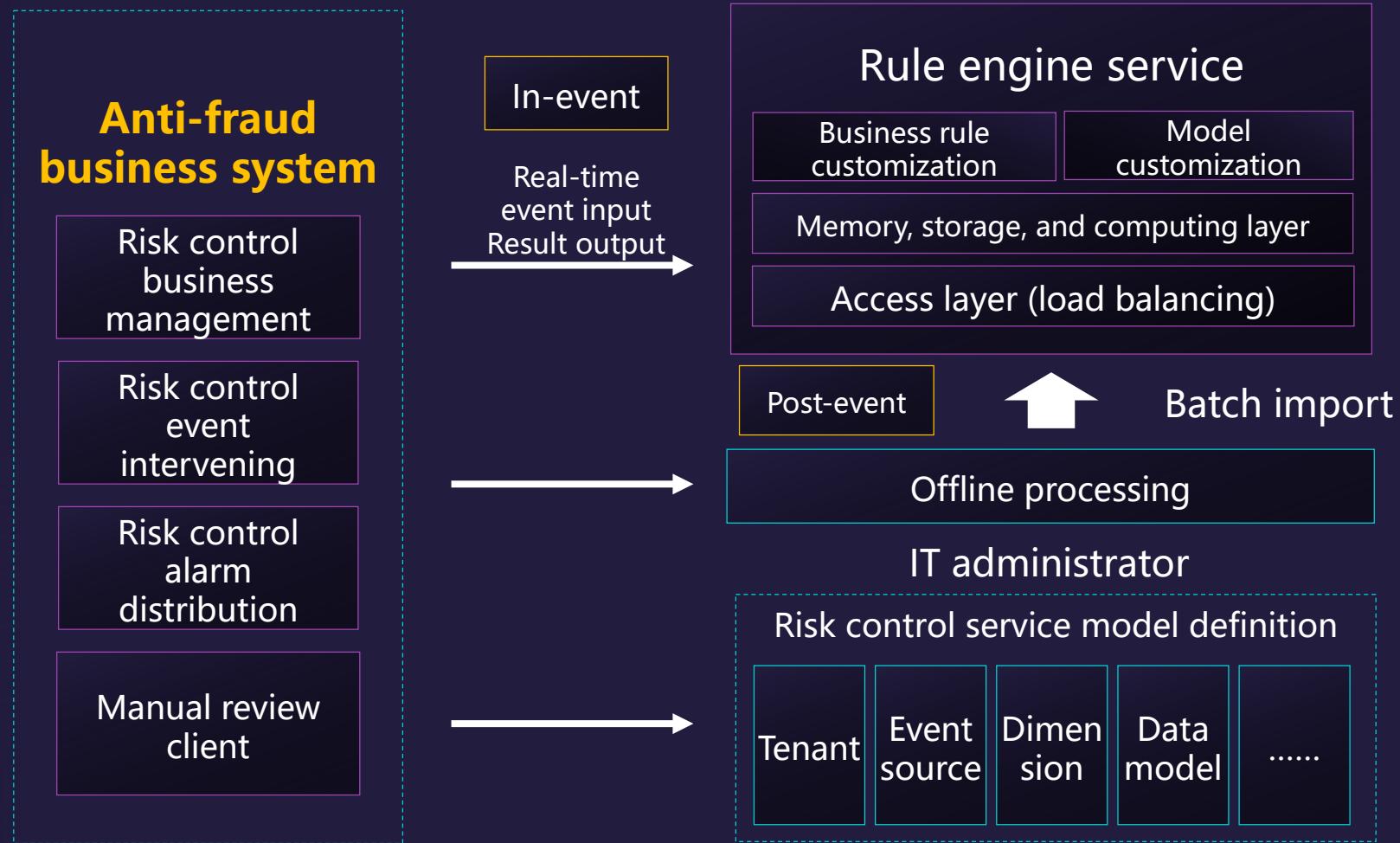




Cloud-based Intelligent Manufacturing Solution

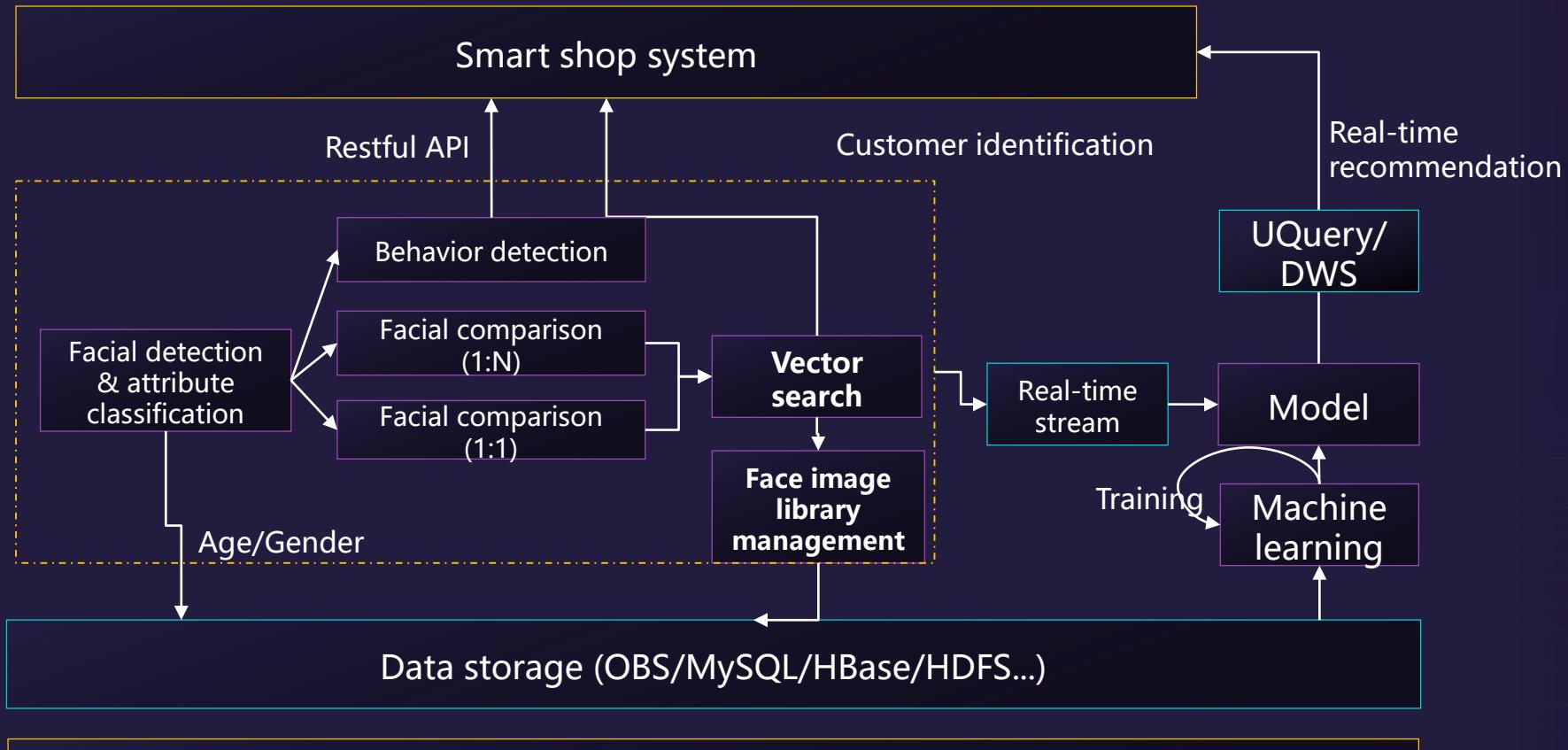


Intelligent Risk Control



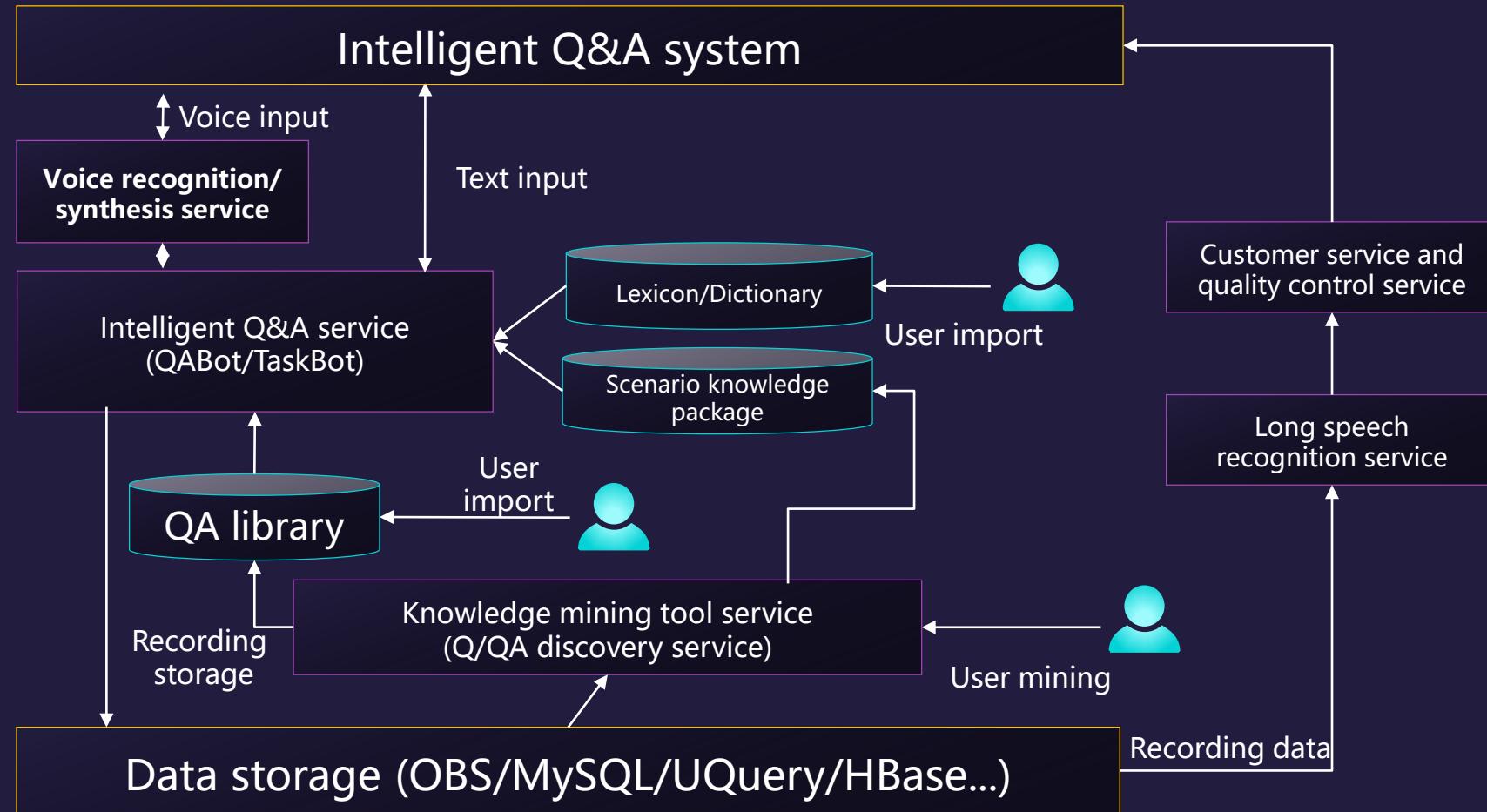


Cloud-based Intelligent Shop Solution for the Retail Industry



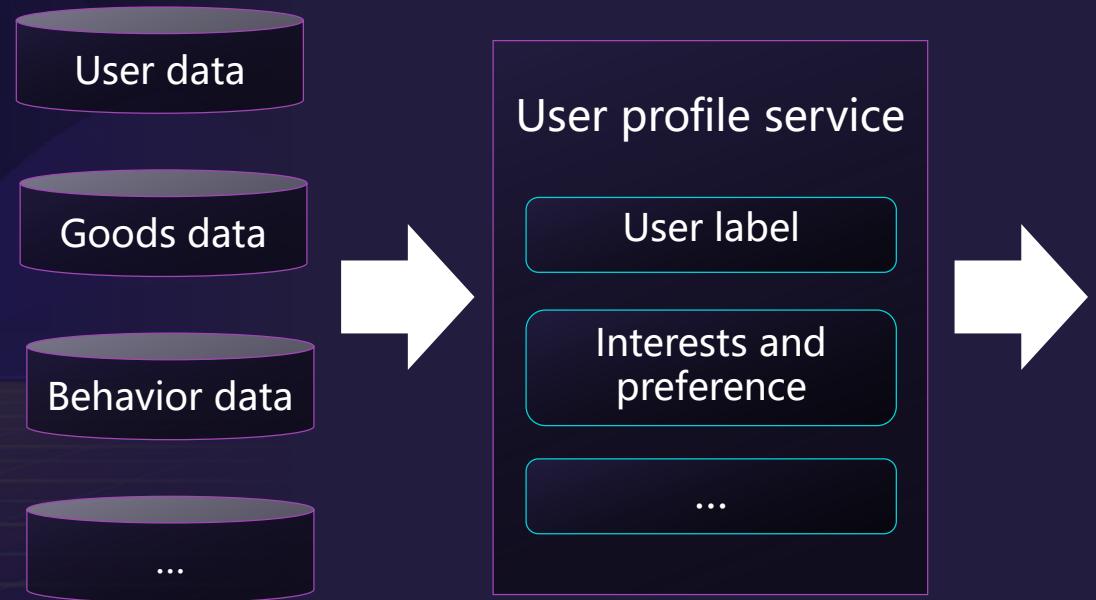


Cloud-based Intelligent Customer Service Solution





Intelligent Recommendation: Personalized Recommendation



Recommendation service

(Home page, association, advertisement, classification, novel, fun, guess you like, local popular, model, increase rank, topic, push, music, video, direct news, direct service, etc.)

Algorithm template

Algorithm template

Algorithm template

ABTest

Effect monitoring

ML/DL/Data IDE

Offline computing

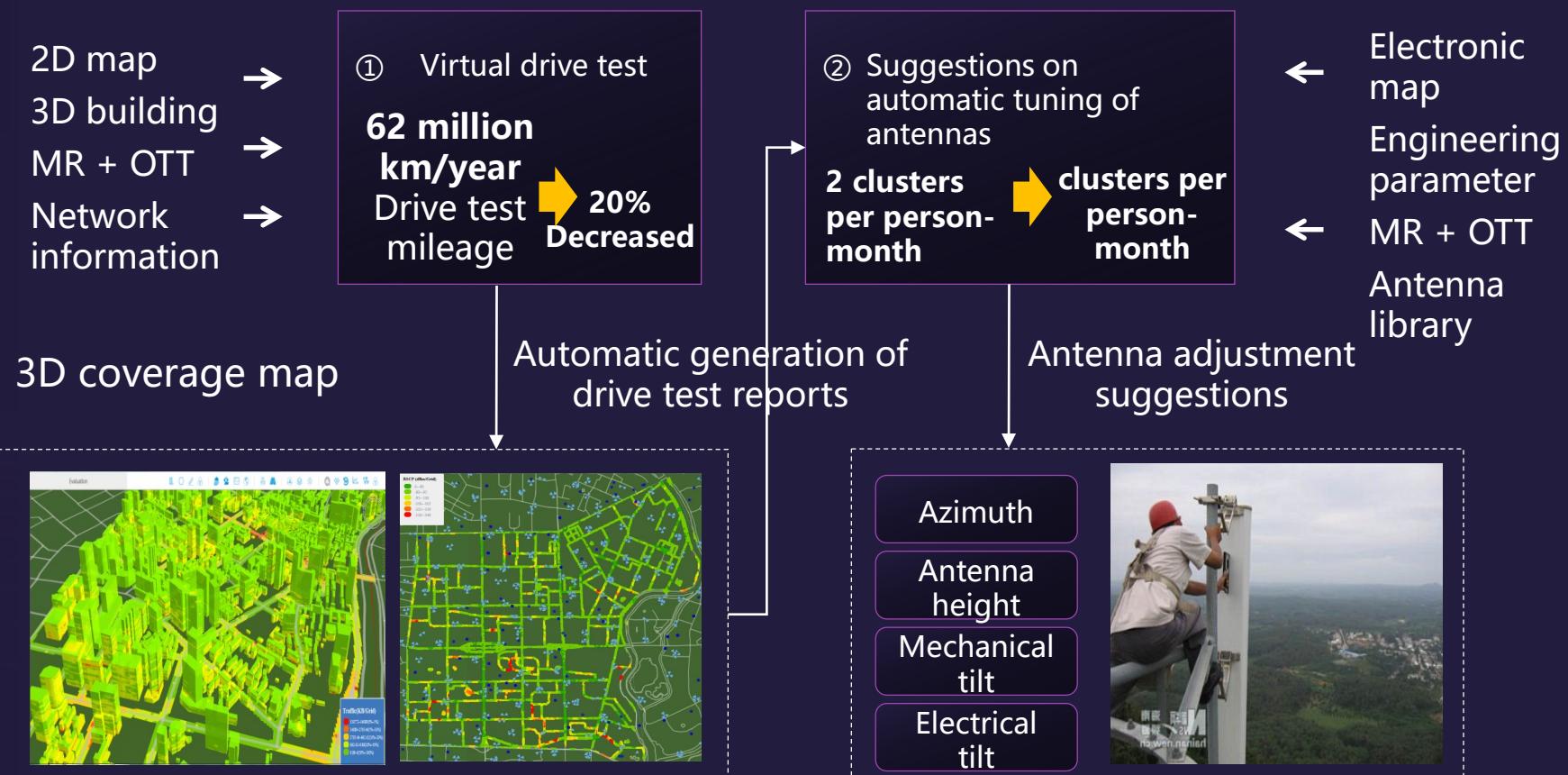
Nearline computing

Online computing

Personalized display



Parameter Adjustment for Intelligent Cell Planning





EI Big Data Services (1)

1

Data Ingestion Service

2

Cloud Data Migration Service

3

Cloud Stream Service

4

MapReduce Service

5

Recommendation Engine Service



EL Big Data Services (2)

1

Data Lake Insight Service

2

CloudTable Service

3

Data Warehouse Service

4

CloudTable Service

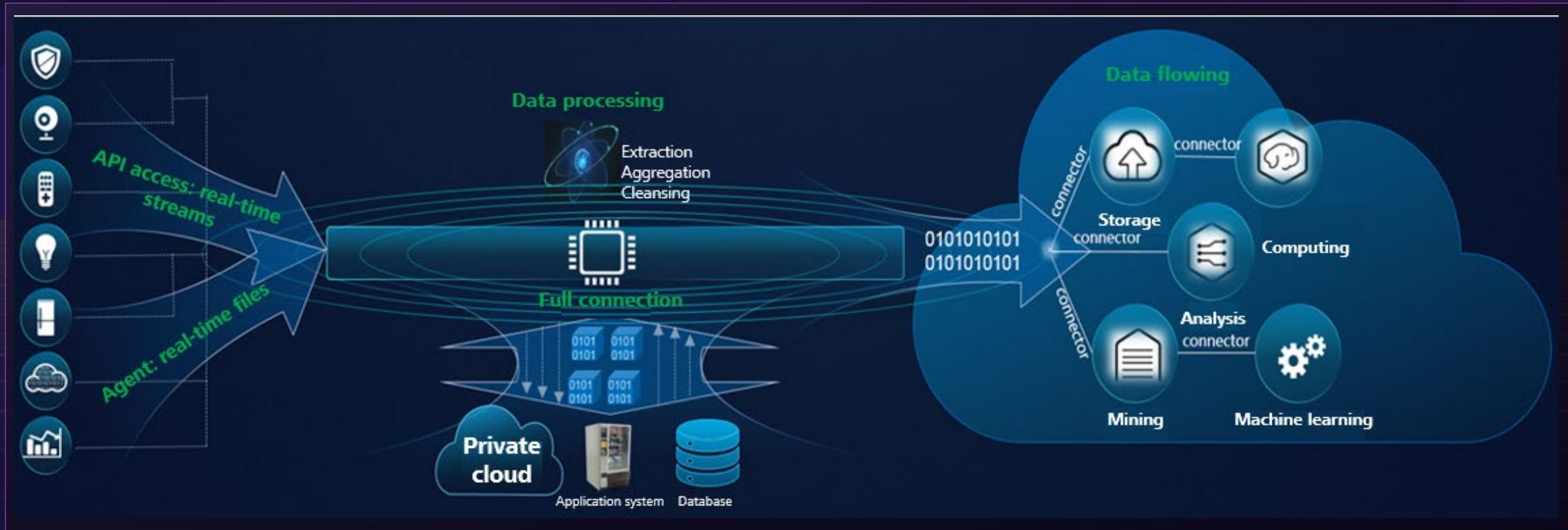
5

Data Lake Factory Service

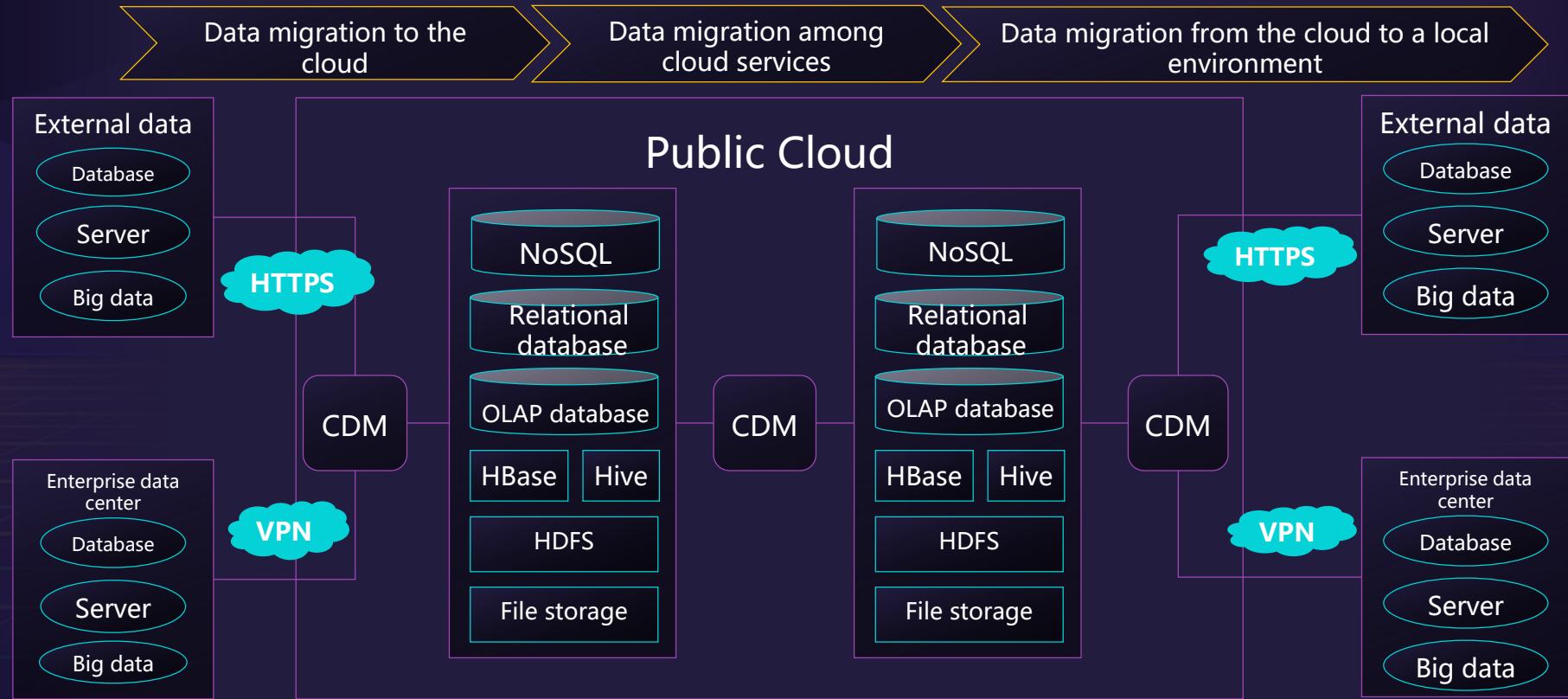


DIS: IoT Real-Time Full Data Connection Pipe

- ◆ Data collection
- ◆ Data transmission
- ◆ Data flowing
- ◆ Intelligent data processing



CDM: Efficient Batch Data Migration



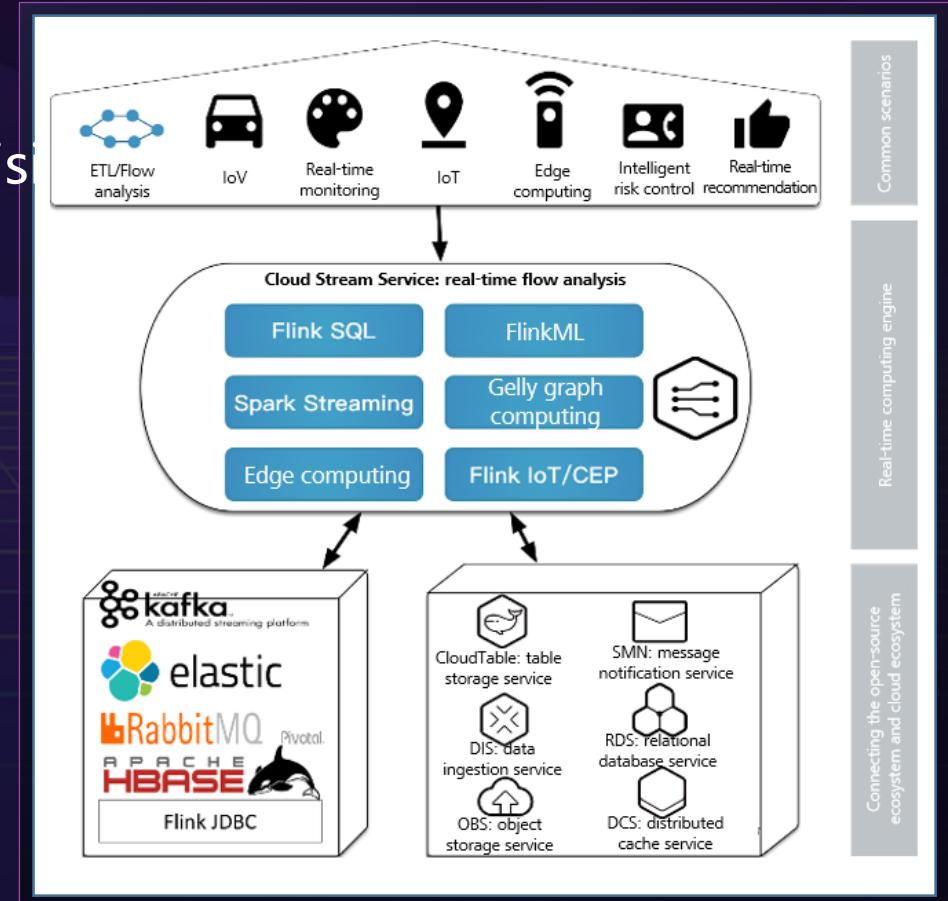
Cloud Stream Service: IoT Core Service

◆ Application scenarios:

- IoT and IoV
- Finance/Securities/E-commerce/Game/Advertisi

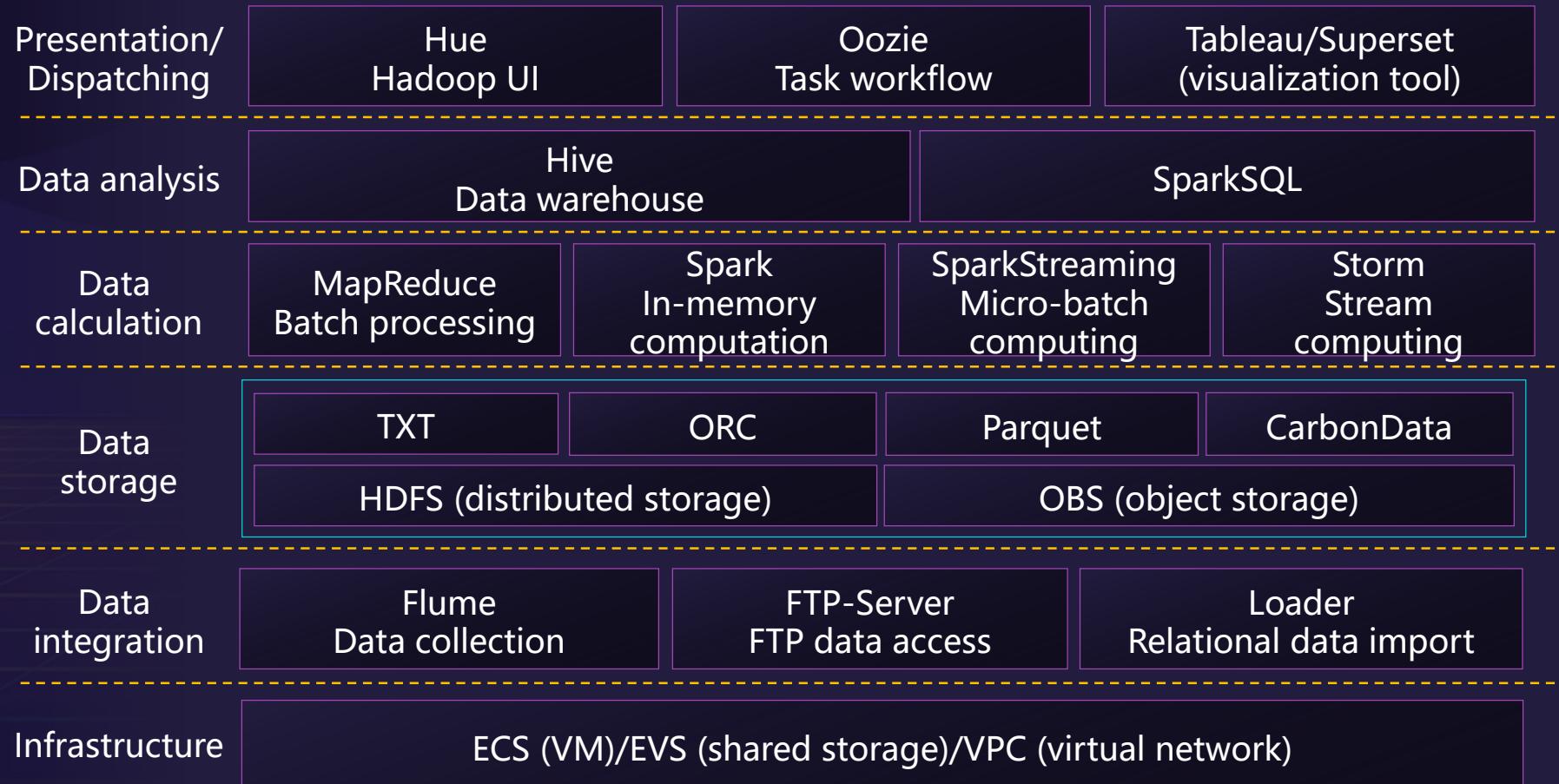
◆ Key competitiveness:

- Dual-engine + dual-service mode
- Easy to use
- Intelligent flow

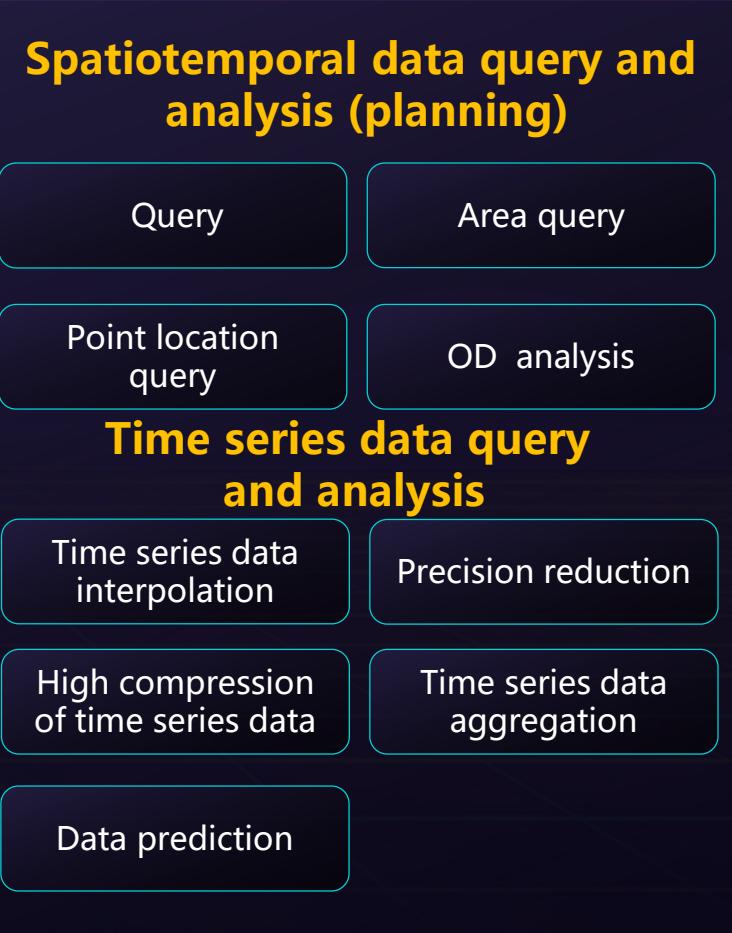
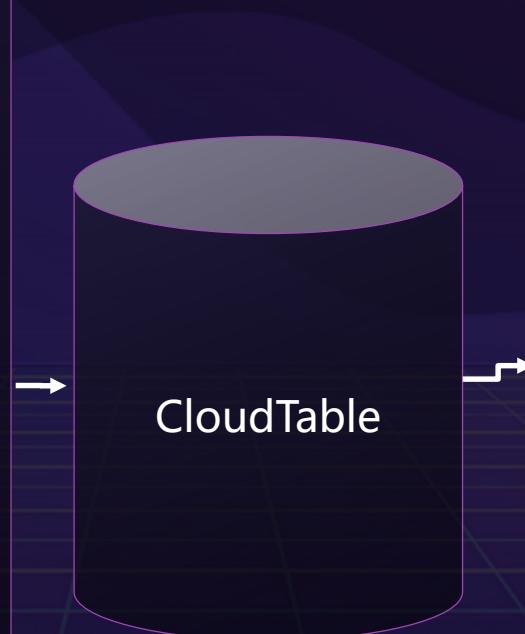




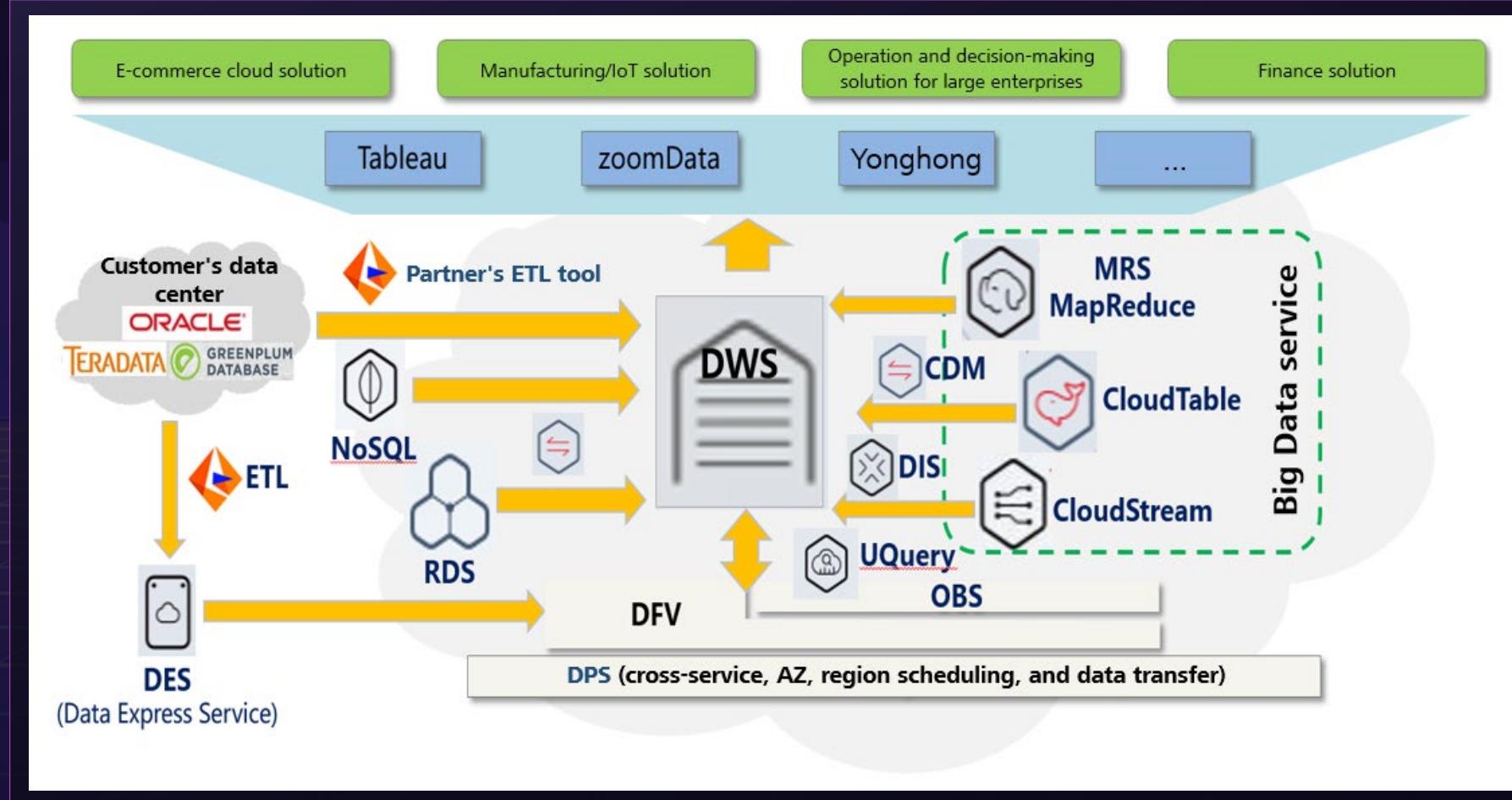
MRS: Enterprise-Level Big Data Cluster Cloud Service



CloudTable: Millisecond-Level NoSQL Database



DWS: Enterprise-Level Cloud Data Warehouse





Section Summary

This document introduces the HUAWEI CLOUD EI ecosystem, development trend of Big Data and AI technologies, capabilities of HUAWEI CLOUD EI service products, and service scenarios and solutions of HUAWEI CLOUD EI.



Quiz

1.What is AI?

2.What services are provided by HUAWEI
CLOUD EI?



1. (Single answer question) When did Huawei provide cloud services and work with more partners to provide a wider variety of AI practices? ()

- A** 2003
- B** 2012
- C** 2015
- D** 2017



2. (True or false) MLS is a data mining and analysis platform service. It enables users to quickly discover data rules, build a prediction model, and deploy the prediction analysis solution. ()

- A** True
- B** False

Thanks

www.huawei.com

