Essentials of Intelligent Technology

John Sum
Institute of Technology Management
National Chung Hsing University
Taichung 402, Taiwan

September 2, 2020

Abstract

In this chapter, the technologies for language understand and image recognition are introduced. Then, the advanced technologies supporting these two applications are presented, with deeper introduction. Owing to highlight that the advanced technologies are indeed developed on top other core technologies, the ideas behind the models and the learning algorithms of the core AI/ML technologies are sketched. Some issues regarding to the core AI/ML technologies development and applications are discussed

Contents

List of Figures

List of Tables

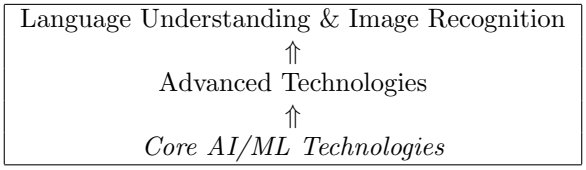
"For every great idea, there are millions of great individuals who saw beyond what was to see what is."

IEEE *The Spirit of Innovation*[1], 2006.

1 Introduction

To introduce the current advancement of intelligent technologies, one simple approach is to introduce the problems (resp. applications) to be tackled by the technologies. Without getting involved with robot, there are two broad areas of problems – language understanding and image recognition. They are the problems (resp. applications) that your smartphone can tackle. From that, the technologies (both intelligent and non-intelligent) supporting such applications could be introduced. Moreover, the limitations of such technologies could be highlighted. Finally, the core AI/ML technologies will be highlighted.

The core technologies are the **essential components** for making such applications work. Take the voice assistant and auto-driving system as examples. A voice assistant applies a number of language understanding (equivalent NLP) technologies for which to recognize the meaning of a voice command. An auto-driving system applies a number of image recognition technologies for which to recognize the objects and their motions. Thus, the controller could decide what actions the mechanical system should take.



As the backgrounds to understand the core AI/ML technology must have advanced level of calculus and statistics, I will only sketch the concepts and ideas behind these technologies. To have a comprehensive knowledge of them, you need to take other advanced courses in AI/ML.

1.1 Language Understanding

Language understand is already a difficult problem. Its level of difficulty raises along with the form of the information feeding in. To understand a plain (resp. printed) text message, the problem is easier if **the text message has no grammatical error and wrong spelling**. If the text message is in a form of historical printed text, the text will have to be converted to the modernized printed text form. If the text message is a handwritten message, the conversion process could be demanding as different people might have different writing styles. If the message is in a form of voice, voice recognition and the text message generation would be even complicated.

[1] Video available on YouTube `https://www.youtube.com/watch?v=h8Jyts3UyLQ`.

So, one can see that the form of the information feeding for language understanding has added additional complexity to the problem itself. If we only have to understand a plain text, in which the text message has no spelling mistake and no grammatical mistake, the problem is the easiest one. Even though, language understand is already a difficult problem.

Even if one can read a text message (resp. listen a speech), one might not interpret the hidden meaning of the text message (resp. the speech). One might not catch the emotion of the writer (resp. speaker) who wrote the text (resp. spoke the speech). These two problems bring the research in language understanding to the next level.

Without considering the problems about the hidden meaning of a message (resp. speech) and the emotion behind the writer (resp. speaker), let me introduce for each form of information the actual problems an AI system (resp. a human being) has to tackle.

1.1.1 Printed Text

Below messages are written in two European languages. Pretty clear, except the name "John Sum", their contents are not in English.

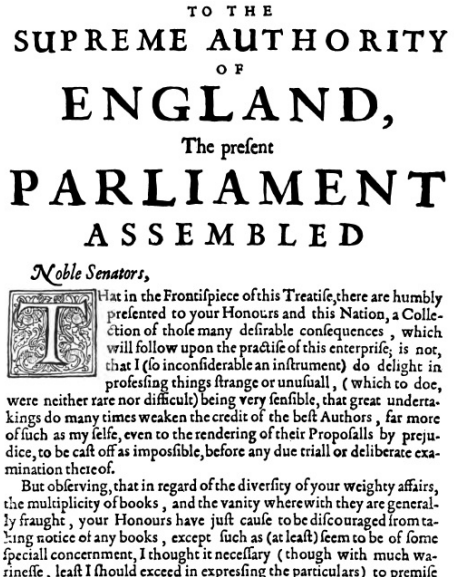
```
Message 1
=========
John Sum est un bel homme dans notre école.
Il s'intéresse à la recherche en IA.

Message 2
=========
John Sum é um homem bonito em nossa escola.
Ele está interessado em pesquisas de IA.
```

Without using Google translator, how could you find out the meanings of these two messages? One simple answer is to learn these languages. After a few months, we are likely to tell what are the meanings of the messages. Before that, we need to find out what languages they are.

In addition, if the first message is post by a student on his course feedback, what exactly he would like to say? That is to say, what is his **hidden meaning behind this message**. If the second message is post by a professor in ITM on a blog, what exactly is his hidden meaning? Besides, are they happy or anger when the time they are posting these messages, **the emotion of the person who post the message**?

1.1.2 Historical Printed Text

A similar problem is to understand a document, like the one as shown in Figure 1. This is a page in the book *Key of Wealth* which was written by William Potter and then published in 1650. The book was written in English. However, you might find something interesting in this paper. Some words seem to have



TO THE
SUPREME AUTHORITY
OF
ENGLAND,
The preſent
PARLIAMENT
ASSEMBLED

Noble Senators,

THat in the Frontiſpiece of this Treatiſe, there are humbly preſented to your Honours and this Nation, a Collection of thoſe many deſirable conſequences, which will follow upon the practiſe of this enterpriſe; is not, that I (ſo inconſiderable an inſtrument) do delight in profeſſing things ſtrange or unuſuall, (which to doe, were neither rare nor difficult) being very ſenſible, that great undertakings do many times weaken the credit of the beſt Authors, far more of ſuch as my ſelfe, even to the rendering of their Propoſalls by prejudice, to be caſt off as impoſſible, before any due triall or deliberate examination thereof.

But obſerving, that in regard of the diverſity of your weighty affairs, the multiplicity of books, and the vanity wherewith they are generally fraught, your Honours have juſt cauſe to be diſcouraged from taking notice of any books, except ſuch as (at leaſt) ſeem to be of ſome ſpeciall concernment, I thought it neceſſary (though with much wa-rineſſe, leaſt I ſhould exceed in expreſſing the particulars) to premiſe

Figure 1: A page in a book published in 1650.

strange spelling. Some lowercase 's' characters were printed like an 'f'. In fact, they are correct. In the books printed in the 17th and 18th centuries in England and other English-speaking countries, lowercase 's' could be printed as a "long s"[2].

In such case, to understand the meaning of the text, one would need to recognize all the characters in the text and convert them to modern characters. For example, the first sentence of the page in Figure 1 will be "To The Supreme Authority of England, The Present Parliament Assembled". Then, the first sentence in the body will be "That in the Frontispiece in this Treatise, there are humbly presented to your Honours and this Nation, a collection of those many desirable consequences, ...".

After the conversion of the document to the modernized English text, one can then read the book in its modernized form. However, how many people could eventually **understand the concepts and ideas disseminated in the book**. This is another challenging problem in language understanding research.

1.1.3 Handwritten Text

For handwritten text documents, the conversion problem is even harder. It is because the writing style of one person could be very difference from the other person. Figure 2 shows a photo with handwritten text. The text was written on a piece of paper one night when I was in a pub relaxing. Suddenly, a thought showed up in my mind. Then, I wrote them down on this paper.

[2] `https://en.wikipedia.org/wiki/Long_s`.

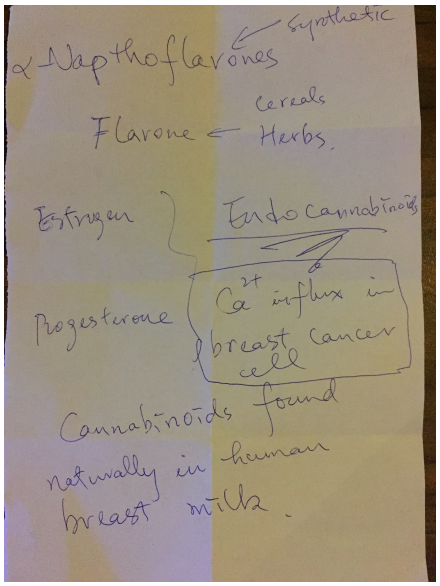


Figure 2: Handwritten text on a paper.

The technical terms are related to breast cancer research. If you are working in breast cancer research, you might understand the meaning of those technical terms. However, **the hidden picture of these terminologies** might not be easy to be interpreted.

Figure 3 shows a more challenging situation. A photo has many objects and one object has handwritten text on it. It is clear from the phot that I was in a pub, with my notebook computer and a paper notebook. The written note is actually something related to AI research. It is full of mathematical equations. Yes, it is one of my research problem in AI. To understand the meaning on the notebook, one needs to **understand the meaning of the English writing together with the meaning of the mathematical equations**. Even a human, it might not be that easy.

If I do not tell that the writing is something related to AI research, could you figure out by yourself? It is hard.

1.1.4 Voice

The last form of information for language understanding is voice, or speech. I speak Mandrin. However, I have accent. For someone from the northern Taiwan might speak different from someone form the southern Taiwan. Therefore, the same speech spoken by different people might have different soundings. Sometimes, a speech might even include a number of dialects. Interpretation of a speech would be even harder.

As mentioned in other chapter, a speech signal is basically a stream of elec-

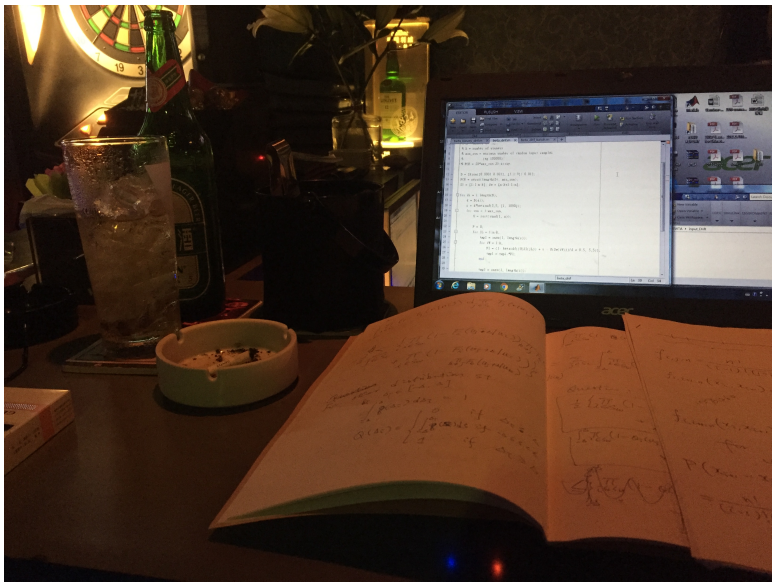


Figure 3: Handwritten text on a notebook.

trical signals. After noise filtering, a clean electrical signal for the speech could be obtained. The next step is to analyze from the electrical signal what words they are. Finally, a text message for the speech can be generated.

If everything is fine, we do not have to worry. However, **accents and dialects could make the conversion of a speech to a text message even harder**. Not to mention, if the speech itself contains more than one language, the conversion is more complicated.

1.2 Image Recognition

Image recognition is another problem every one of us has to tackle in our daily live and daily work. To be nice, we normally call the name of a friend whom we have came across in a street. It is image recognition.

1.2.1 Face Recognition

Here are two sample photos. Both of them were taken in graduation dinner a few years ago. The male student would like to take a selfie from four of us. In the end, he failed to do so even he had attempted twice. For the photo in Figure 4, could you tell me how many *happy faces* there? For the photo in Figure 5, could you tell me how many *happy faces* there?

To answer these two questions, one need to identify all the faces and their locations in the photos. Once all the faces have been found, the emotion of each person is analyzed by the *facial expression*. For the ladies, you probably guess

Figure 4: Dinner photo 1.

Figure 5: Dinner photo 2.

Figure 6: Inside a hardware shop.

that they are happy. *But, how about me? How about my male student in the second photo.* Yes, this problem is essentially the same as the problem to figure out the emotion of the writer of a message. From an image, could we find out **the emotions of the people in the image**.

1.2.2 Object Recognition

Object recognition is another difficult problem in AI. Figure 6 is a photo taken inside a local hardware shop. In this photo, there are many objects in it. Could you tell me what are they? How do you recognize them?

Luckily, object has no emotion. We do not have to figure out if they are happy or not. Still, the problem might not be easy. **This hardware shop sells a lot of different objects. They include "...".** Can you tell what are they?

2 Technologies for Language Understanding

The fundamental usage of language understanding is to generate a text command, or a program, to the operating system. Let us have a simple example.

2.1 Text Command

If I would like to rename a file "john.pdf" to "mary.pdf", the following Microsoft Windows command will be text on the command prompt. Let say, I have changed the directory to "`C:\Users\JohnSum\Desktop>`".

```
C:>cd Users
C:\Users>cd JohnSum
C:\Users\JohnSum>cd Desktop
C:\Users\JohnSum\Desktop>rename john.pdf mary.pdf
```

Here, "cd" and "rename" are commands for the Windows Operating System (OS). "cd" stands for "change directory to". "rename" is used for renaming a filename. Windows OS has more than 280 commands available for a user to command the system. To command the operating system, one needs to follow the syntax specified by the operating system. Like "cd", the whole message must follow the following syntax.

'cd directoryname', or 'cd directorypath'

In technical term, `C:\Users\JohnSum\Desktop` is called the path of the directory. If you wrongly type the directory name "JohnSum" as "John-Sum", the OS will fail to do so and an error message will be shown on screen.

2.2 Voice Command

Imagine that I have built a software *Jorman* to handle voice command. The above file rename task could be accomplished by the following dialogue.

```
John: Hey Jorman!
Jorman: What can I do for you?
John: Run the command prompt.
Jorman: Done!
John: What is the current directory?
Jorman: Root directory of the C-drive.
John: Is Users a directory here?
Jorman: Yes!
John: Change directory to Users.
Jorman: Done!
John: Check if JohnSum is a directory here.
Jorman: Yes, it is.
John: Change directory to JohnSum.
Jorman: It is done!
John: Check if Desktop a directory here.
Jorman: Yes, it is.
John: Change directory to Desktop.
Jorman: It is done!
```

```
John: Check if the file "john.pdf" is here.
Jorman: Yes, it is.
John: Rename it to "mary.pdf".
Jorman: It is done.
```

It is clear from this example, talk with a voice assistant to complete a task might not be as simple as typing in the commands on the command prompt. What a voice assistant does is simply to **convert a voice command to a text command which can be understood by the OS**. If the voice command is complicated, it will be converted to a sequence of text commands. In other words, the voice command is converted to a program instructing the OS to do a sequence of jobs.

2.3 Technologies

Let us look at what exactly *Jorman* has done for me. The specific tasks being done are put in the brackets.

```
John: Hey Jorman!
(Voice recognition)
(Voice synthesization)
Jorman: What can I do for you?
John: Run the command prompt.
(Voice recognition)
(C:\>cmd )
(Voice synthesization)
Jorman: Done!
John: What is the current directory?
(Voice recognition)
(C:\>chdir )
(Voice synthesization)
Jorman: Root directory of the C-drive.
John: Is Users a directory here?
(Voice recognition)
(C:\>dir Users)
(Voice synthesization)
Jorman: Yes!
John: Change directory to Users.
(Voice recognition)
(C:\>cd Users )
(Voice synthesization)
Jorman: Done!
John: Check if JohnSum is a directory here.
(Voice recognition)
```

```
(C:\Users>dir JohnSum )
(Voice synthesization)
Jorman: Yes, it is.
John: Change directory to JohnSum.
(Voice recognition)
(C:\Users>cd JohnSum )
(Voice synthesization)
Jorman: It is done!
John: Check if Desktop a directory here.
(Voice recognition)
(C:\Users\JohnSum>dir Desktop )
(Voice synthesization)
Jorman: Yes, it is.
John: Change directory to Desktop.
(Voice recognition)
(C:\Users\JohnSum>cd Desktop )
(Voice synthesization)
Jorman: It is done!
John: Check if the file "john.pdf" is here.
(Voice recognition)
(C:\Users\JohnSum\Desktop>dir john.pdf )
(Voice synthesization)
Jorman: Yes, it is.
John: Rename it to "mary.pdf".
(Voice recognition)
(C:\Users\JohnSum\Desktop>rename john.pdf mary.pdf )
(Voice synthesization)
Jorman: It is done.
```

Therefore, a basic usage of Siri or Alexa is to **convert the voice speech into a sequence of commands which is understood by the corresponding operating system**. Another usage is speech synthesization. Now, one should realize that **the set of recognizable speeches (i.e. voice commands) depends on the set of text commands available in an operating system**.

2.4 Limitations

Here, I said, "Check if the file john.pdf is here.". It is very specific and very clear. A voice assistant could interpret its meaning easily. What if I change the sentence to one of the following sentences?

```
John: Look up the directory seeing if the file "john" is here.
John: Look up the directory seeing if john is here.
```

These sentences are not clear enough. The voice assistant might not be able to interpret.

Another condition is about the voice quality. If the voice commander (i.e. *John*) cannot speak clearly or the background is too noisy, the voice assistant is not able to interpret. Recall that voice assistant needs to do a few tasks in order to response to a voice command. *First, the voice has to be converted to electrical signal. Second, noise is filtered from the electrical signal. Third, words are segmented from the electrical signal. Forth, a sentence is generated from the words and its meaning is analyzed.* "Meaning" refers to the task to be done by the assistant for the voice commander. The voice assistant can interpret a voice command only if there is no any problem in each of these four steps. If the voice assistant get stuck in any one step, the voice command cannot be interpreted.

In the following, I list a few exceptional cases which could cause problems to the voice assistant.

- Voice command is not clear enough.
- The background is too noisy.
- The voice command is too long.
- The speaking is too fast or too slow.
- The voice commander has too strong accent.
- The voice commander is not an authorized user.

It could be many more on the list.

2.5 Advanced Technologies

In Section 2.3, I have only mentioned the names of three technologies for language understanding – voice recognition (equivalently voice-to-text) system, sentence understanding (equivalently semantic analysis) system and voice synthesization (equivalently text-to-speech) system. Moreover, I have recalled in Section 2.4 the four-step language understanding process. In fact, many advanced technologies have been applied in each step to make the process of language understanding work.

2.5.1 Signal Processing

A voice recognition system has applied many advanced technologies from the area of *signal processing*. Once a voice signal has been sensed by a microphone, the corresponding electrical signal generated is a continuous analog electrical signal. Figure 7 shows an exemplar speech signal.

A digital computer is not able to process analog electrical signal. Thus, a special hardware called ADC[3] will have to sample from this analog electrical

[3]ADC stands for analog-to-digital converter.

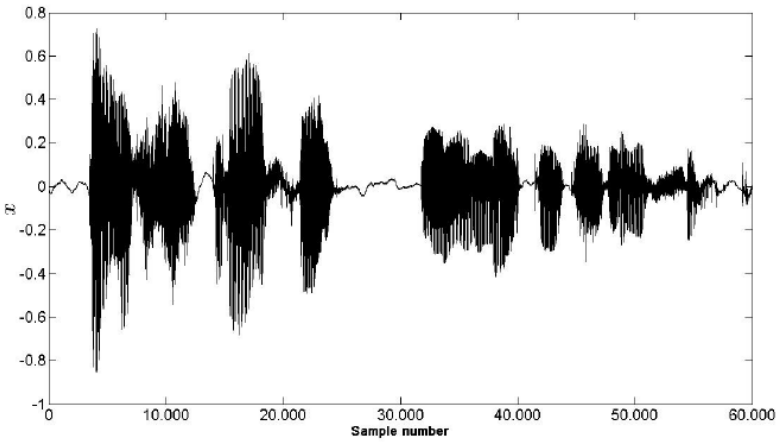


Figure 7: Sample speech signal.

signal to a sequence of numerical data. Each data is the voltage level of the signal sampled at one specific moment. As the frequency of a speech ranges from 300 Hertz to 3400 Hertz, number of samples to be catched in a second for a speech is 8000. In other word, the sampling frequency is 8000Hz or 8000 samples per second. So, the ADC will generate a sequence of 16000 data if your voice command lasts for two seconds. So, the speech shown in Figure 7 should likely be lasted for 7.5 seconds.

As the voice command must be corrupted with background noise, this sequence of numerical data must be processed in order to generate another sequence of numerical data corresponding to the noise-free voice command. This step is called noise cancellation or noise filtering. The signal shown in Figure 7 is actually the speech signal already gone through noise cancellation.

2.5.2 Signal to Words

Once a clean signal has been obtained, the next step is to segment the signal into a number of short signals. Each short signal (i.e. phoneme) could be identified as a word or part of a word. This process is complicated. This step consists of two tasks – **segmentation** and **word labelling**. Both tasks require different technologies.

It could be seen from Figure 7 that the speech consists of roughly nine sounds. Each sound could be mapped to a word. Segmentation of the nine sounds is relatively easy, identifying all possible stop-moments and then isolating the word-signal from every two consecutive stop-moments. Once an isolated word-signal (i.e. a phoneme) has been segmented, its corresponding word will have to be recognized. This part is difficult. Many (intelligent) technologies have been developed for this task.

The segment word-signal is passed to a frequency analyzer to get the frequency spectrum of the word-segment. Note that frequency spectrum is a signature (equivalently a set of features) for a phoneme. This step is also called **feature extraction** – extracting from the electrical signal the features of a phoneme. Based upon the set of features, machine learning model like **multilayer perceptron (MLP)** could be applied to map the signal features to label for the word.

Long in the history, researchers have known that the efficiency of a voice recognition system is bounded by the computation power of the processors, such as the CPU and the GPU. It is because **the feature extraction algorithm and the word-label mapping algorithm involve intensive mathematical calculations**. Each algorithm might involve thousands and even tens of thousands floating point number multiplications. Imagine that a processor can only handle one thousand floating point number multiplication in a second[4], the processing time for feature extraction and word-label mapping would be in term of seconds or even more. This time lag is not acceptable for real-time applications.

2.5.3 Meaning Extraction

The last step in language understanding is to get the meaning of the voice command. Note that we have now a sequence of words. There are many methods for extracting the meaning of a sentence. Some of them apply concepts from natural language processing (NLP)[5]. NLP is a research area in linguistic and computer science. I do not have background on it.

So, I can only explain to you the following method which could be applied to extract the meaning of a voice command. This approach is called template matching. It involves a step of creating a set of command templates. It is not an intelligent method. For certain extent, it is a dump method. But, it always works for simple tasks.

Command Template

S1 To collect from the volunteers a set of voice commands. Let say, there are 200 volunteers. Each of them is asked to give 20 voice commands for 20 different applications of an iPhone.

S2 Each voice command is then manually converted to a text command.

S3 All these text commands are then labelled based on their applications.

This set of command templates is thus used for extracting the meaning of a voice command.

Meaning Extraction (Semantic Analysis)

[4]In technical terminology, the measure of a computational power of a processor is called floating point operations per second (FLOPs).

[5]`https://en.wikipedia.org/wiki/Natural_language_processing`.

S1 The text command is compared with the commands in the set of command templates. The similarity values of the text command to the templates are evaluated.

S2 The command template with the highest similarity value is selected and its label is extracted.

S3 If the similarity value is higher than an acceptable level, the label is assigned to the text command.

S4 As the label is the code of a specific command to the OS, the actual command to the OS is generated.

S5 Call the OS to run the command.

To create the set of command templates is accomplished by the research lab. Once the templates have been created, it can simply installed in the device. The voice assistant only has to perform the meaning extraction wit the aid of the set of command templates. Pretty clear, this method could be improved if the size of the set of command templates increases.

This method could also be applied to sentence reconstruction. For instance, in Step S3 in the meaning extraction, the similarity value could be smaller than the acceptance level. Then, the voice assistant is not possible to call the OS to run a command. In such case, the voice assistant could reconstruct the command and tell the commander the reconstructed command.

One should note that the technologies introduced in this section are not the core AI/ML technologies, while the section heading is called advanced technologies. The most advanced and the core technologies are something far more than your imagination, see Section 4. They are the true intelligent technologies.

2.6 Document Understanding

For the text form of language as presented in Section 1.1, such as printed text and handwritten text, those images have to be first pre-processed to extract the text. Once the text has been obtained, understanding the meaning of the text/document is basically gone through the same processes as elucidated in the Section 2.5. But, the technologies are much more sophisticated. Some of them will be introduced in Section 4.

Something amazing have been appeared – **multiple-document understanding** and **multiple-document summarization**. Even more, by understanding tons of research articles in biomedical research, researchers have developed systems to read and generate hypotheses automatically from these huge number of information sources [1, 2, 3, 4]. As mentioned in [1, p.1879], reading over 240 thousands papers with human kinases in the Medline bibliographic database [6] would need to spend 70 years to go through. system.

[6] `https://www.nlm.nih.gov/bsd/medline.html`.

2.7 Voice Commander and Document Writer

To make the language understanding system work, the voice commander and the document writer have also played an important role. **The presentation skill of a voice commander or a document writer could cause the severe failure in language understanding**. A voice commander needs to give a clear message. Normally, a simple sentence is better. It could be easily interpreted by the voice assistant. A document writer should write the document in logical manner. The ideas presented in the document have to be clearly stated.

One should note that this presentation problem not just exists in between a human voice commander and a voice assistant. It is a problem in between two humans. A human assistant can hardly interpret a voice command which is not logical and poorly spoken. For the best of my knowledge, no technology has been developed to correct these presentation problems. Thus, a voice commander or a document writer has the responsibility to present properly.

2.8 Machine-Generated Monograph

Another far beyond your imagination, the first machine-generated textbook on lithium-ion batteries has been published in 2019 [5]. This book is primarily generated by a system called **Beta Writer**. **Beta Writer** is system developed in a joint-effort and in collaboration between Springer Nature and researchers in Goethe University Frankfurt, Germany. Editors and researchers in lithium-ion batteries are involved.

As a first attempt, 1086 publications were solicited. **Beta Writer** is able to analyze the similarities among the articles and group them into clusters. Then, **Beta Writer** generates summarization for each cluster of articles. This step is tedious. The book generation process is not fully automated. **Beta Writer** serves a lot more like an writing assistant for human experts. Human experts are the key writers. They help **Beta Writer** developers refining the parameters and rules for **articles clustering**, **text-summarization** and **sections ordering**.

For implementation detail, one can refer to the *Preface* in [5]. In the end, a book surveying 1086 publications in the topic of lithium-ion batteries is eventually generated and gone through a normal peer-review process. Finally, Springer published the book in 2019.

3 Technologies for Image Recognition

Various technologies for image recognition have been developed for half a century. Image recognition technologies are important technologies for realization of information storage and auto-driving. In this section, some early technologies for image recognition and the current technologies for object recognition will be introduced.

3.1 Early Technologies for Object Recognition

In the early day, dated back to the 1970s to 1990s, the technologies for recognizing objects in an image lied in three areas – **object edge extraction**, **object segmentation** and **shape recognition**. The first two areas mainly apply the technologies in computer vision. The last part **shape recognition** requires technologies from AI/ML.

3.1.1 Human Visual System Inspired

Worthwhile to mention, some technologies in computer vision are inspired from the theories from human vision system [6, 7, 8, 9]. For example, our retinas and the visual cortexes are full of neurons. These neurons are able to extract a point and an edge from a visual image. Those neurons perform like various spatial filters[7]. The spatial filters are 2D filters, like the following 3×3 matrices[8].

$$\underset{\text{Point}}{\begin{bmatrix} -1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & -1 \end{bmatrix}} \quad \underset{\text{V. Edge}}{\begin{bmatrix} -1 & 1 & -1 \\ -1 & 1 & -1 \\ -1 & 1 & -1 \end{bmatrix}} \quad \underset{\text{-45\% Edge}}{\begin{bmatrix} 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{bmatrix}}$$

The first one can filter out an isolated point. The second one can filter out a point on a vertical edge. The third one can filter out a point on an edge with -45%.

3.1.2 Object Boundary Extraction

Let us have a simple showing how these filters are applied to construct the boundary of an object. An image of size 5-pixel by 5-pixel is originally in gray scale. After initial preprocessing, a rough edge information is extracted If a pixel is likely to be a point on the edge of the object, it is represented by '1'. Otherwise, it is '0'.

$$\mathcal{I} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{1}$$

Figure 8 shows the idea of image filtering. Let I be the image on the left and the neurons responsible for image filtering are on the right. The output of

[7] In technical term, they are Gabor filters.

[8] The filters can also be defined as a 5×5 matrix or 7×7 matrix. Moreover, the filter could be particularly defined for handling special purpose. Here are a few examples.

$$\begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix}, \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

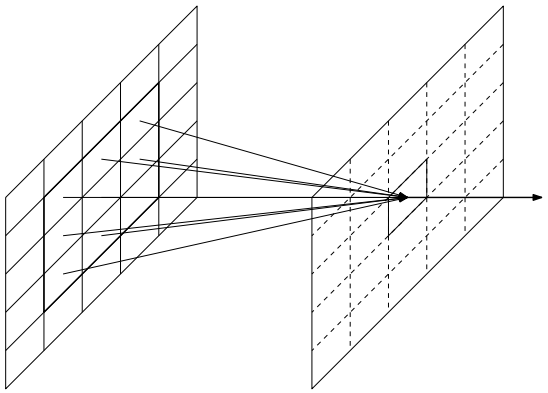


Figure 8: Image filtering. It is analogized to a two-layer neuronal network. Each layer has 25 neurons. The neuronal network on the second layer is also called a feature map.

the neurons on the right is denoted as F.

$$I = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} \\ x_{41} & x_{42} & x_{43} & x_{44} & x_{45} \\ x_{51} & x_{52} & x_{53} & x_{54} & x_{55} \end{bmatrix}, \quad F = \begin{bmatrix} f_{11} & f_{12} & f_{13} & f_{14} & f_{15} \\ f_{21} & f_{22} & f_{23} & f_{24} & f_{25} \\ f_{31} & f_{32} & f_{33} & f_{34} & f_{35} \\ f_{41} & f_{42} & f_{43} & f_{44} & f_{45} \\ f_{51} & f_{52} & f_{53} & f_{54} & f_{55} \end{bmatrix}.$$

We further let W be a filter.

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}.$$

The value of f_{ij} is given by

$$\begin{aligned} f_{ij} &= h(u_{ij}) \\ h(u_{ij}) &= \begin{cases} 1 & \text{if } u_{ij} > 0, \\ 0 & \text{if } u_{ij} \leq 0, \end{cases} \end{aligned}$$

where u_{ij} is the input to the ijth neuron.

$$\begin{aligned} u_{ij} &= w_{11}x_{i-1,j-1} + w_{12}x_{i-1,j} + w_{13}x_{i-1,j+1} \\ &+ w_{21}x_{i,j-1} + w_{22}x_{ij} + w_{23}x_{i,j+1} \\ &+ w_{31}x_{i+1,j-1} + w_{32}x_{i+1,j} + w_{33}x_{i+1,j+1}. \end{aligned}$$

For the boundary neuron, say f_{11}, it is assumed that there are zero elements outside the image. So,

$$u_{11} = w_{22}x_{11} + w_{23}x_{12} + w_{32}x_{21} + w_{33}x_{22}.$$

The same technique applies to other boundary neurons.

It should be noted that the number of neuronal feature maps on the second layer is equal to the number of filters. Figure 8 only shows one feature map on the second layer. If there are 16 filter types, there are 16 neuronal maps on the second layer. Each map has 25 neurons.

Once this image $\mathcal{I}$, as given in (1), is passed to the above filters, we will get the following information.

$$\underset{\text{Point}}{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}} \quad \underset{\text{V. Edge}}{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}} \quad \underset{\text{-45\% Edge}}{\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}}.$$

The first matrix indicates which pixel is an isolated point. The second matrix indicates which pixel is a point on a vertical edge. The third matrix indicates which pixel is a point on a -45% edge. This information is thus applied to reconstruct the boundary of the object in three steps.

- The first step is to remove isolated points. It could be done by check the first matrix. So, the pixel at the left bottom is removed.
- The second step is to merge the results from the second and the third matrix. It results the following matrix representing the pixels which are the boundary points of the object. Clearly, it is not a connected boundary.

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

- The third step is to connect these two edge segments. Depending on the methods applied, two possible outcomes could be obtained.

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

  It is usually assumed that a boundary should have certain **smoothness**[9]. So, the boundary of an object should be represented as the following

[9]Note that smoothness is a common assumption in many image processing and object recognition systems.

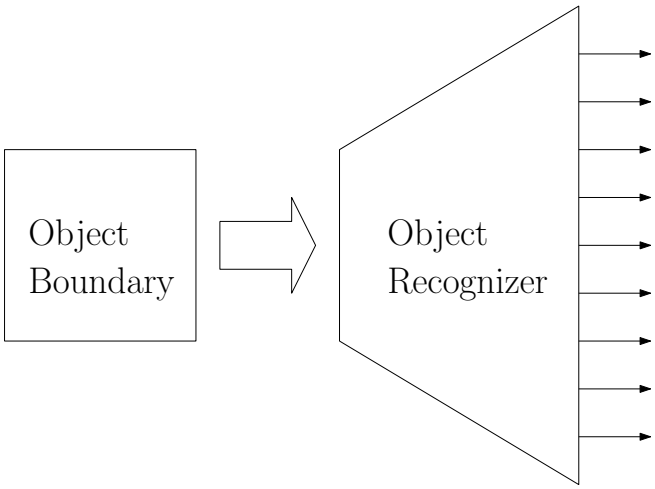


Figure 9: A well-trained object recognizer which can recognize nine objects. Usually, the input to the recognizer is the raw object boundary. It is the normalized object boundary.

matrix.

$$\mathcal{O} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}. \tag{2}$$

The boundary of an object has been obtained. The boundary matrix (2) can then be used for object segmentation[10] and object recognition.

3.1.3 Object Recognition

Finally, the object boundary matrix (2) is thus used for object recognition. To do so, an object recognizer is applied. The boundary matrix $\mathcal{O}$ is fed to the recognizer. The recognizer then outputs a label for $\mathcal{O}$. In this part, the object recognizer is usually an AI/ML model like MLP which is a **well-trained model**. Well-trained model is the model which gives the highest accuracy, say 99.5%. To obtain such a well-trained model, we need to apply a **learning algorithm** to train the model, see Section 4.

[10]To segment an object, the idea is simply to fill up the elements inside the boundary to one, i.e.

$$M = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

and use it as a mask to segment the object out of the image.

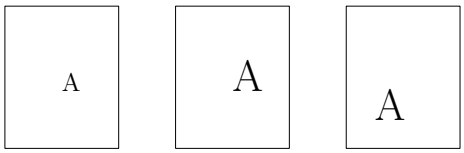


Figure 10: Three sample images of the capital letter A.

Figure 9 shows an object recognizer which has been trained to recognize nine objects[11]. Let $z_1, z_2, \cdots, z_9$ be the outputs of the recognizer. The output z_i $(i = 1, \cdots, 9)$ is confined to the range $[0, 1]$. Each output corresponds to an object. For ideal case, the recognizer should have only a single output with '1' and others '0'. In reality, it always happens something like the following results.

Object	Dog	Man	Boy	Cat	Axe	Egg	Car	Lady	Girl
Output	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8	z_9
Value	0.2	0.9	0.7	0.2	0.1	0.1	0.1	0.3	0.5

Problem 1 *If the output values are neither zero or one, how do we make decision from the object recognizer?*

Getting this well-trained recognizer requires the knowledge from the core AI/ML. It will be presented later in Section 4. In regard to the input to the recognizer, it is usually not the raw object boundary. As the size of an object boundary could vary form image to image, the boundary has to be scaled to a fixed size before feeding to the recognizer. This step is called **normalization**. The actual input to the recognizer is called the **normalized object boundary**.

Here, I have just sketched a few key ideas in the conventional technologies for object recognition. For further reading, one can refer to a monograph authored by Richard O. Duda, Peter E. Hart and David G. Stork [10]. It is a classic in the areas of computer vision, image processing and pattern recognition. For sure, you need to have a good command of mathematics.

3.2 Neocognitron/DNN for Object Recognition

As introduced, early object recognition technologies have already applied some ideas from neuroscience on human vision system. Still, their performance are not the same as our vision systems. A human does not have to rely on segmentation and normalization to recognize an object. A human can recognize an object irrespective to its size, location and orientation on the image. These three properties are called scale invariant, translational invariant and orientational invariant.

[11] For your information, the latest object recognizers can recognize thousands of objects. This number will definitely raise in the future.

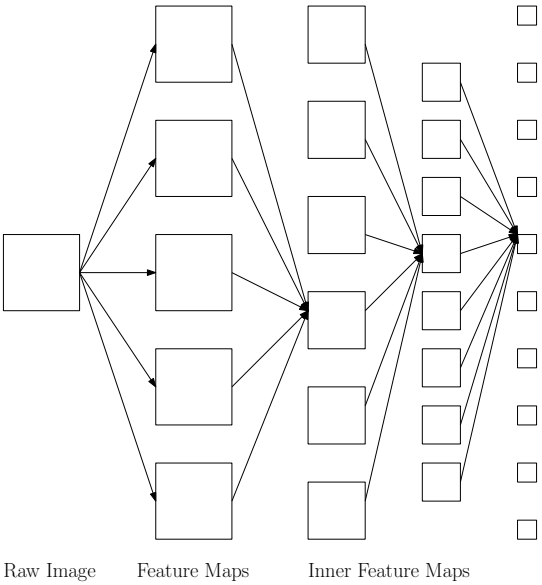


Figure 11: Multi-layer structure of a human visual system-inspired neural network. Raw image at left is the input to the neural network. The layer of nodes on the right is the output layer.

Figure 10 should three images of the capital letter 'A'. The left one is a small size 'A' located around the middle of the image. The middle one is a bigger size 'A'. The size of the 'A' in the right hand side is the same as the on in the middle image. But, its location is different. Clearly, we are able to see that the letters are all 'A'.

The scale, translational and orientation invariant properties have drawn the attention of scholars. In the end, two notable models were developed and presented in 1975 and 1980 respectively. Inspired by findings and postulations from physiologists [11, 12, 13, 6, 7], Kunihiko Fukushima developed Cognitron [14] and later Neocognitron [15] for object recognition. These models have later influenced Yann LeCun *et al.* to develop a multilayer network in 1989 [16] and the LeNet-5 convolution neural network (CNN) for character recognition in 1998 [17], Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton to develop the AlexNet for object recognition in 2012 [18] and subsequent deep convolutional neural networks in the late 2010s [19, 20, 21, 22].

3.2.1 Working Principle

The working principle behind these models is based upon the following ideas which are discovered by David H. Hubel and Torsten N. Wiesel [11, 12, 13].

- The visual system consists of multiple-layer of feature maps.

- In each layer, there are multiple feature maps.
- Each feature map in the first layer is of the same size as the raw image.
- Each feature map from the $(k-1)$-layer will map to all the feature maps at the k-layer.
- The size of a feature map in a inner layer, say the k-layer, is always smaller than the size of a feature map in the precedence layer, i.e. the $(k-1)$-layer. The size of each feature map in the final-layer is the smallest amongst all the feature maps in the network. Thus, the features captured in a k-layer map is always more general than the features captured in a $(k-1)$-layer feature map.
- All the final-layer feature maps are fed to the last layer of neurons for object recognition.

The general structure of this kind of neural networks is shown in Figure 11. The mapping between the raw image to the first-layer feature maps, we could interpret the mapping is a generic feature mapping, such as edge extraction and orientation feature extraction. For feature maps in deeper layers, the mapping would not be that easy to interpret. One could treat them as advanced feature maps. They capture the so-called complex features of an image [16], as compared with the localized simple features captured in the early feature maps.

3.2.2 Practical Considerations

While the principle behind the neural network models for object recognition is quite straight forward, the actual implementations of such models face a number of practical concerns.

- Size of an image.
- Number of object classes.
- Number of layers.
- Number of feature maps in each layer.
- Size of a feature map in each layer.
- Size of a filter of a feature map.
- Computation power of a computer.

One should know that the models introduced in here and in Section 3.1.3 are essentially a multiple-input-multiple-output mapping function. Each function has many parameters. They also called the model parameters.

Consider the model as shown in Figure 11. There are five feature maps in the first layer. Each map corresponds to the feature extracted by a single filter. If we assume that all the filter matrices are 3×3 matrix, number of parameters

Table 1: Number of parameters of the model in Figure 11.

Connection Layer	Filter Size	No.	Filter Size	No.
Image $\rightarrow L_1$	3×3	45	3×3	45
$L_1 \rightarrow L_2$	2×2	24	3×3	54
$L_2 \rightarrow L_3$	2×2	32	2×2	32
$L_3 \rightarrow$ Output	–	320	–	320
Total	–	421	–	451

Table 2: Number of parameters of deep neural networks.

Model	No. of Parameters
LeNet (1998)	60,000
AlexNet (2012)	60,000,000
GoogLeNet (2014)	7,000,000
VGG16 (2014)	138,000,000
VGG19 (2014)	144,000,000
ResNet18 (2015)	11.700,000
ResNet50 (2015)	25,600,000
ResNet101 (2015)	44,600,000

is $5 \times (3 \times 3)$, i.e. 45. From the first layer to the second layer, each feature map in the first layer is mapped to six different feature maps in the second layer. Assume that the filter is now of size 2×2, number of parameters is $6 \times (2 \times 2)$, i.e. 24. Then, each feature map in the second layer is mapped to 8 feature maps in the third layer. Again, assume that the size of a filter is 2×2, number of parameters is $8 \times (2 \times 2)$, i.e. 32. From the third layer to the output layer, each feature element connects to all output nodes. Number of parameters is 320. As a result, the total number of parameters of the model as shown in Figure 11 is 421. If the size of the filters in $L_1 \rightarrow L_2$ is changed to 3×3, the total number of parameters is 451, as depicted in Table 1.

The total number of parameters reveals the complexity of a model. If you have a well-trained model, you can simply use it right the way for object recognition. A model with 400 plus parameters is a simple model. Number of mathematical calculation in the process of object recognition is relatively small. Normal CPU can do it in less than a second. If you do not have a well-trained model, you cannot randomly set the parameters. **Learning algorithm** will have to be applied to train the model. The time spent on training is again related to the number of parameters. Luckily, for this small size model, it might take less than an hour to get a well-trained model. Then, we can use it for object recognition.

3.2.3 Increasing Complexity – Use of GPU

From the above example, one should note that the time spent on training a model for use is much longer than the time a model is being used. Now, let us take a look on some popular deep neural network models depicted in Table 2. These models have very large number of parameters. The largest number is 144 millions. Using a normal CPU, it might take a few minutes for a well-trained model to do an object recognition. To conduct a training, it could take a few weeks. For sure, it is not efficient. So, specialized design processor (GPU) has been employed to perform these tasks. In research lab, researchers applied GPUs to conduct the training. Once the well-trained model has been obtained, the model parameters are saved in the application software for use, like the auto-driving system and the optical character recognition system.

3.3 Neural Network for Object Recognition

Neural network is a model inspired by the property of a biological neuron. In Section 3.1.3, I have mentioned a model called MLP. Actually, it is a neural network model. Here, I would like to emphasize that **neural network** refers to a collection of models[12]. Some models have already been applied in object recognition.

Like the multi-layer model in Figure 11, MLP is a multi-layer structured network. It has one input layer, multiple hidden layers and one output layer. The number of nodes in the input layer (resp. output layer) depend on the application. For instance, in a handwritten digit recognition application, the database consists of 70,000 images[13]. Each image is of size 28, i.e. 784 pixels. Each image belongs to one and only one of these 10 classes – '0', '1', '2', '3', '4', '5', '6', '7', '8' and '9'. In real application, the MLP need to recognize if a new 28 belongs to one of these classes. So, we can simply consider the MLP as a 784-input-10-output mapping model.

3.3.1 Model Specification

Figure 12 shows an MLP for handwritten digit recognition application. For illustration, the image is of size 4×4. So, this MLP has 16 input, instead of 784 input. Note that each image is indeed a 4×4 matrix. The value of the ijth element is the brightness of the ijth pixel of the image. As the input to the input layer must be in one dimension, the image is first partitioned into four rows. The elements in the first row is fed to the first 4 input nodes of the MLP. The leftmost element is fed to the input node on top. Then, the elements in the second row is fed to the second 4 input nodes and the leftmost element in this row is fed to the fifth input node. The elements in the third and forth rows are fed to the input nodes in the same manner.

[12] In Section 4 of the lecture note on *Evolution of Technology*, I have introduced some of them. Formally speaking, **neural network** refers a collection of **families of models**.

[13] It is the MNIST dataset. It is available for download from `http://yann.lecun.com/exdb/mnist/`.

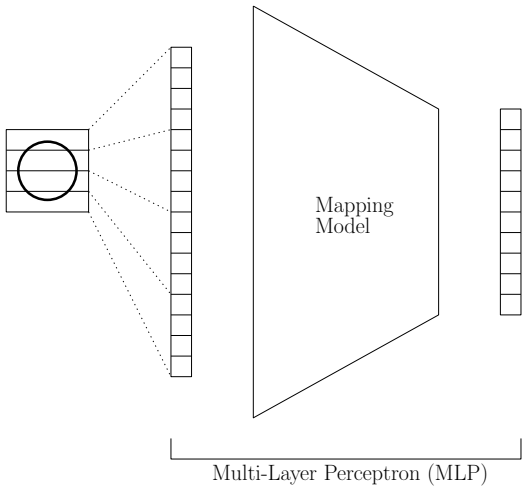


Figure 12: Consider an MLP as a mapping model.

For the output nodes, their mapping to the digits are defined in accordance to the following table. Here z_1 corresponds to the topmost output node. The output z_i, for $i = 1, \cdots, 10$, is in the range from 0 to 1, i.e. $z_i \in [0, 1]$.

Label	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8	z_9	z_{10}
0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	1	0	0
8	0	0	0	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	0	1

This definition is also applied to the label of an image in the database. Usually, the kth image is denoted by $\mathbf{x}_k$ and its label is denoted by $\mathbf{y}_k$. For the handwritten digit database MNIST introduced above, $k = 1, \cdots, 70000$.

Once the specification of the model input and output has been defined, the model of the MLP can simply be treated as a multiple-input-multiple-output (MIMO) mapping model. Pretty clear, the multi-layer model as shown in Figure 11 could also be plugged in as this mapping model.

3.3.2 Model for Object Recognition

Symbolically, the model can be denoted as $\mathbf{f}(\mathbf{x}, \mathbf{w})$, where $\mathbf{x}$ is the input vector and $\mathbf{w}$ is set of model parameters. For $i = 1, \cdots, 10$, $f_i(\mathbf{x}, \mathbf{w}) = z_i(\mathbf{x}, \mathbf{w})$.

Similar to the argument presented in Section 3.2.2, different sets of model parameters will give different performance. Some models perform better and some perform worsen.

So, what should be the model to be used for application? **Learning algorithm** will have to be applied to train the model, i.e. to get the parameters $\mathbf{w}$ of the model that will give the best performance.

So, how to measure the performance of a model with parametric vector $\mathbf{w}$? A simple measure is the mean squared error (MSE). For the MNIST database, there are $70,000$ images of handwritten digits, the measure of the performance of a model, denoted as $E(\mathbf{w})$, is given by

$$E(\mathbf{w}) = \frac{1}{70000} \sum_{k=1}^{70000} \left(\frac{1}{10} \sum_{i=1}^{10} \left(y_{ki} - f_i(\mathbf{x}_k, \mathbf{w}) \right)^2 \right).$$

In compact form, it can also be written as follows :

$$E(\mathbf{w}) = \frac{1}{70000} \sum_{k=1}^{70000} \left\| \mathbf{y}_k - \mathbf{f}(\mathbf{x}_k, \mathbf{w}) \right\|_2^2,$$

where $\| \cdot \|_2$ is the l_2-norm.

Thus, the model to be used for object recognition must be the one with the best performance, i.e. the one with the minimum $E(\mathbf{w})$. It is the one which gives the least error.

3.3.3 Learning Algorithm

Now, it comes to another problem – how to find (or search for) the parametric vector $\mathbf{w}$ which gives the minimum $E(\mathbf{w})$? Clearly, it is about the so-called learning or training. Suppose the best model is of a parametric vector $\mathbf{w}^*$, i.e.

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left\{ E(\mathbf{w}) \right\}.$$

In essence, learning or training is referred to a numerical algorithm which could iteratively update the model parametric vector $\mathbf{w}$ and eventually $\mathbf{w} \to \mathbf{w}^*$. If a model has millions of parameters, like VGG101, the time spent on running the learning algorithm to get the parametric vector $\mathbf{w}$ could be in terms of months if GPU is not used.

3.3.4 Practical Considerations

To apply neural network in image/object recognition, the same practical concerns as in the use of DNN have to be considered.

- Size of an image.
- Number of object classes.
- Number of layers.

- Number of neurons in each layer.
- Transfer function of a neuron.
- Computation power of a computer.

These concerns would determine the applicability of a neural network model.

3.3.5 My Experience

In 2019, I was requested to conduct a research on the MNIST database. The model I applied is a MLP. The total number of parameters of the model being studied is 90294. Using my desktop computer in my office, it took almost a week to run the program to get the best model. At the same time, I asked one of my friends in Nanyang Technological University to run the same program with the use of GPUs. It took only two hours.

3.4 Application in Auto-Driving Systems

In Section 3.2, I have introduced a few deep neural networks (DNNs) that have succeeded in accurate object recognition. Those DNNs are able to recognize the objects in a natural image with size 224×224 pixels. The number of objects to be recognized is in term of thousands. Thus, these well-trained DNNs have been applied in auto-driving systems. The cameras installed in a vehicle capture real-time 3D images, say 38 frames per second. The object recognizer in the vehicle computer recognizes from the image frames what objects appear in front of it. Those information are input to the dynamic control system to control the mechanical driving system to react to those objects.

3.4.1 Practical Considerations

It should be noted that the AI/ML model is applied to recognize the objects from a video in real-time. The problem complexity is very different from the object recognition task conducted in a lab. In a lab, there is no time limit for the model to recognize the objects in an image. In auto-driving system, the model has to recognize the objects from 38 images in one second. Computational work is far more demanding.

Here, I list some factors that could affect the performance and the safety of an AI/ML model in auto-driving system.

- Number of images taken in a second.
- Size of an image.
- Number of object classes that can be recognized.
- Complexity of the AI/ML model.
- Accuracy of the AI/ML model in object recognition.

- Computation power of the computer installed in a vehicle.
- Maximum braking force and the weight of a vehicle.

The last factor has no direct relation to the AI/ML model. Instead, it is a constraint to be considered in the overall design.

Let us have a simple example. A vehicle is moving at speed 90km/h, i.e. 25 m/s. Its AI/ML model has recognized a big obstacle located 50 meters in front of the vehicle. If the vehicle does not slow down, it will crash the obstacle in 2 seconds. So, the AI/ML system sends a command 'brake and stop within 25 meters' to the dynamic control system. The vehicle has to be decelerated at 6.25 m/s/s. If the vehicle is a sports car or family car, the braking force should be strong enough to make it stop. If the vehicle is a truck of 10 tons, the braking force might not be strong enough.

3.4.2 Car Accident

The above example reveals that car accident could happen even if the AI/ML has made correct decision. Not enough time for the dynamic system to execute the decision is one factor.

Apart from this, other factors could also cause car accident. One of them is from the AI/ML system. It should be bare in mind that no any AI/ML model can give 100% accuracy. Therefore, an auto-driving system can have mistakes because the AI/ML system makes mistaken recognition.

Imagine that a deep neural network (DNN) can achieve 99.9% accuracy in object recognition. The car has been running for 30 minutes. In each second, the camera generates 38 frames of image. Assume that the average number of objects in an image is 10. The total number of objects being recognized in a period of 30 minutes is $10 \times 38 \times 60 \times 30$, i.e. $684,000$. This number clearly indicates that the AI/ML system could have already made 684 mistakes after a 30 minutes trip. Even if the accuracy raises to 99.99%, still, there are 68 mistakes. These mistakes could lead to wrong decision to the dynamic control system.

The accuracy of an AI/ML model in object recognition is based upon its recognition on a predefined set of objects and the images in an image database. If an image contains an unknown object, the unknown object could possibly be ignored. Then, the AI/ML system could lead to wrong decision.

Another factor is about the label of a still object. If a still object is labelled as 'nothing', it is likely to be removed from the image for (moving) object recognition. Thus, the probability of the auto-driving vehicle crashing on an accidental vehicle or a road block will be high[14].

3.5 Sentiment Analysis from a Photo

Similar to the area of document understanding, sentiment analysis is also a research topic in image understanding. However, the technologies developed for

[14] `https://www.youtube.com/watch?v=FVgkWii5JdM`.

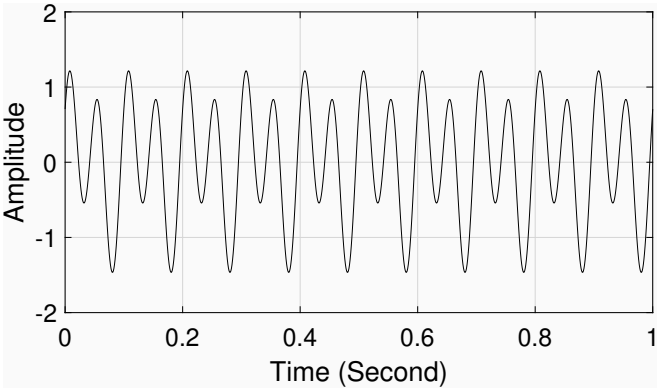


Figure 13: A simple signal.

this area have yet to be matured. Just like the photos in Figure 4 and Figure 5, how could tell which one of us is happy or not? The challenge in sentiment analysis of a person in a photo lies on the truth revelation of person during the moment of photo taking. If a person intentionally acted like happy but he/she was not happy in that moment, it is not easy for a system to identify if his/her feeling is positive or not. So far, there has not yet had promising for this. Might be, sentiment analysis on a person in a photo can never be realized. A video with sound might make it possible.

4 Core AI/ML Technologies

In the previous sections, a number of technologies have been mentioned. Only one AI/ML model has been mentioned, the MLP. The working principle of MLP has not been introduced. In this section, I will explain more detail why we can apply MLP in language understanding. To have a comprehensive coverage on AI/ML, you would need to have *good knowledge* in mathematics, statistics and programming. If you would like to challenge yourself, you can read the monographs [23, 24, 25, 26].

4.1 Core AI/ML : A Case of Sound Recognition

Imagine that there are two sets of samples – labelled set and unlabelled set. Each sample in the labelled set has been labelled with its corresponding sound label. The samples in the unlabelled set have not been labelled. That is to say, their corresponding sound labels are unknown.

Problem 2 *Given a set of labelled samples, find the labels for the samples in the unlabelled set.*

Table 3: Five labelled signals and a unlabelled signal.

Signal	10	20	30	40	50	Label				
d	0.5	1	0	0	0	1	0	0	0	0
k	0	0	0.2	0.4	0	0	1	0	0	0
p	0	0.2	0	0.1	1	0	0	1	0	0
la	0.3	0	0.5	0	0.1	0	0	0	1	0
aa	0	0.7	0	1.2	0.1	0	0	0	0	1
?	0	0.6	0.1	0.9	0	?	?	?	?	?

4.1.1 Problem Formulation

Recall that a signal feature is physically the amplitude of a sinusoidal wave of a particular frequency. Figure 13 shows a simple signal consisting of two sinusoidal waves. Let $s(t)$ is the amplitude of the signal at time t. The signal is given by

$$s(t) = \alpha_1 \sin(2\pi f_1 t + \phi_1) + \alpha_2 \sin(2\pi f_2 t + \phi_2),$$

where $\alpha_1 = 0.5$, $f_1 = 10$, $\phi_1 = 0$, $\alpha_2 = 1$, $f_2 = 20$ and $\phi_2 = \pi/4$. Now, I predefine a set of frequencies for generating the feature for a signal. The set is $\{10, 20, 30, 40, 50\}$.

Let F_s be the feature vector (a column vector) of the signal $s(t)$. We get that

$$F_s = \begin{bmatrix} 0.5 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

If there is another signal $s'(t)$. Its signal is given by

$$s'(t) = \alpha'_1 \sin(2\pi f'_1 t + \phi'_1) + \alpha'_2 \sin(2\pi f'_2 t + \phi'_2),$$

where $\alpha'_1 = 0.4$, $f'_1 = 40$, $\phi'_1 = \pi/2$, $\alpha'_2 = 0.2$, $f'_2 = 30$ and $\phi'_2 = \pi/8$. Its feature vector is that

$$F_{s'} = \begin{bmatrix} 0 \\ 0 \\ 0.2 \\ 0.4 \\ 0 \end{bmatrix}.$$

Suppose we have five signals. Their features and labels (in binary) are depicted in Table 3. The first two signals are $s(t)$ and $s'(t)$. In the table, there is one unlabelled signal.

Solution: *Using the labelled samples, find the function mapping from signal features to the labels.*

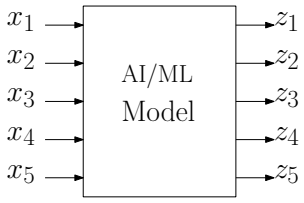


Figure 14: Model for mapping.

Table 4: Position assignment.

	(Input **x**)					(Output **z**)				
k	x_1	x_2	x_3	x_4	x_5	z_1	z_2	z_3	z_4	z_5
1	0.5	1	0	0	0	1	0	0	0	0
2	0	0	0.2	0.4	0	0	1	0	0	0
3	0	0.2	0	0.1	1	0	0	1	0	0
4	0.3	0	0.5	0	0.1	0	0	0	1	0
5	0	0.7	0	1.2	0.1	0	0	0	0	1

The idea is to define a **model** with five inputs and five outputs, as shown in Figure 14. At the same time, the features and the labels of the labelled signals depicted in the Table 3 are assigned with their corresponding positions at the model, as depicted in Table 4.

4.1.2 Model Definition & Graphical Structure

Now, it comes to another problem. What model should be defined? Formally speaking, what should be the mathematical definition of the model for Problem 2.

Problem 3 *What should be the mathematical definition of the model for the signal labelling problem as stated in Problem 2?*

Technically speaking, Problem 3 is an open problem. There is no unique answer. Many possible models could be applied. It could be defined as the following model.

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} & w_{15} \\ w_{21} & w_{22} & w_{23} & w_{24} & w_{25} \\ w_{31} & w_{32} & w_{33} & w_{34} & w_{35} \\ w_{41} & w_{42} & w_{43} & w_{44} & w_{45} \\ w_{51} & w_{52} & w_{53} & w_{54} & w_{55} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}, \tag{3}$$

where (w_{ij}) and b_i are the parameters to be determined. If we assume that z_1,

z_2 and z_3 do not depend on x_4 and x_5, we could define the model as follows :

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & w_{13} & 0 & 0 \\ w_{21} & w_{22} & w_{23} & 0 & 0 \\ w_{31} & w_{32} & w_{33} & 0 & 0 \\ 0 & 0 & 0 & w_{44} & w_{45} \\ 0 & 0 & 0 & w_{54} & w_{55} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}, \tag{4}$$

where (w_{ij}) and b_i are the parameters to be determined. If we plug in a nonlinear function to bound the value of z_i in the range of $[0, 1]$, the model is defined as follows :

$$\begin{aligned} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix} &= \begin{bmatrix} \phi\left(w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}x_4 + w_{15}x_5 + b_1\right) \\ \phi\left(w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + w_{24}x_4 + w_{25}x_5 + b_2\right) \\ \phi\left(w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + w_{34}x_4 + w_{35}x_5 + b_3\right) \\ \phi\left(w_{41}x_1 + w_{42}x_2 + w_{43}x_3 + w_{44}x_4 + w_{45}x_5 + b_4\right) \\ \phi\left(w_{51}x_1 + w_{52}x_2 + w_{53}x_3 + w_{54}x_4 + w_{55}x_5 + b_5\right) \end{bmatrix}, \\ \phi(u) &= \frac{1}{1 + \exp(-u)}, \end{aligned} \tag{5}$$

where (w_{ij}) and b_i are the parameters to be determined. $\phi(\cdot)$ in (5) is called the standard logistic sigmoid function, or simply the sigmoid function, see Figure 15(a). If the sigmoid function is replaced by a step function, a similar definition is given as follows :

$$\begin{aligned} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix} &= \begin{bmatrix} \phi\left(w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}x_4 + w_{15}x_5 + b_1\right) \\ \phi\left(w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + w_{24}x_4 + w_{25}x_5 + b_2\right) \\ \phi\left(w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + w_{34}x_4 + w_{35}x_5 + b_3\right) \\ \phi\left(w_{41}x_1 + w_{42}x_2 + w_{43}x_3 + w_{44}x_4 + w_{45}x_5 + b_4\right) \\ \phi\left(w_{51}x_1 + w_{52}x_2 + w_{53}x_3 + w_{54}x_4 + w_{55}x_5 + b_5\right) \end{bmatrix}, \\ \phi(u) &= \begin{cases} 1 & \text{if } u \geq 0 \\ 0 & \text{if } u < 0, \end{cases} \end{aligned} \tag{6}$$

where (w_{ij}) and b_i are the parameters to be determined. The function $\phi(\cdot)$ is called the step function or threshold function, as shown in Figure 15(b). Clearly, we can define a more complicated model as follows :

$$\begin{aligned} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} &= \begin{bmatrix} \phi\left(w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}x_4 + w_{15}x_5 + b_1\right) \\ \phi\left(w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + w_{24}x_4 + w_{25}x_5 + b_2\right) \\ \phi\left(w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + w_{34}x_4 + w_{35}x_5 + b3\right) \end{bmatrix}, \\ \phi(u) &= \begin{cases} 1 & \text{if } u \geq 0 \\ 0 & \text{if } u < 0 \end{cases} \end{aligned}$$

and

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix} = \begin{bmatrix} w'_{11} & w'_{12} & w'_{13} \\ w'_{21} & w'_{22} & w'_{23} \\ w'_{31} & w'_{32} & w'_{33} \\ w'_{41} & w'_{42} & w'_{43} \\ w'_{51} & w'_{52} & w'_{53} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} + \begin{bmatrix} b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \end{bmatrix}, \tag{7}$$

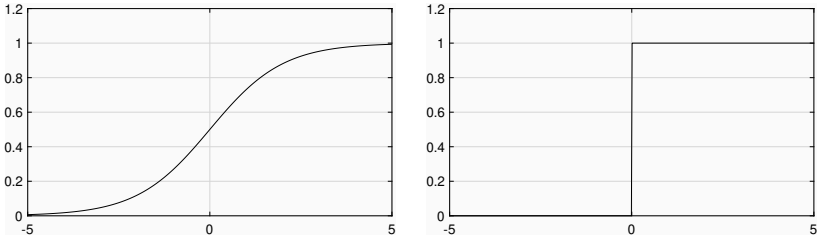


Figure 15: Nonlinear functions: (a) sigmoid and (b) step.

where (w_{ij}), (w'_{ij}), b_i and b'_i are the parameters to be determined.

The above models, from (3)-(7), are five different models. Formally speaking, they have different model structures. Their graphical structures are shown in Figure 16. The structures of (3), (5) and (6) are fully connected, Figure 16(a). The structure of (4) is shown in Figure 16(b). It is a partial connected structure. The structure of (7) is a layered structure, as shown in Figure 16(c). Note that, graphical structure is just a visual model. The actual definition of a model has to refer to the mathematical definition. Like (3), (5) and (6), their graphical structures are the same but their mathematical definitions are different.

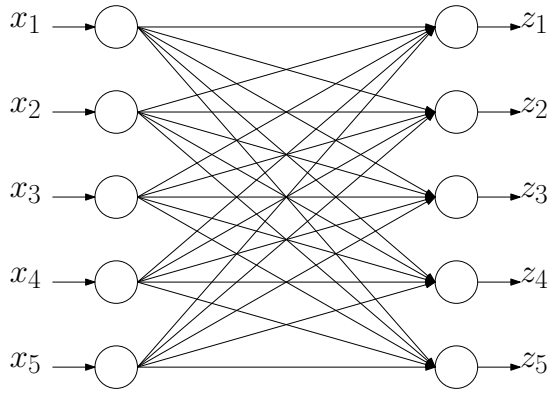
The models, from (3)-(7), are the only FIVE OUT OF MANY possible models. **Could you tell which model should be defined for the mapping in Figure 14?**

4.1.3 Learning : What is it about?

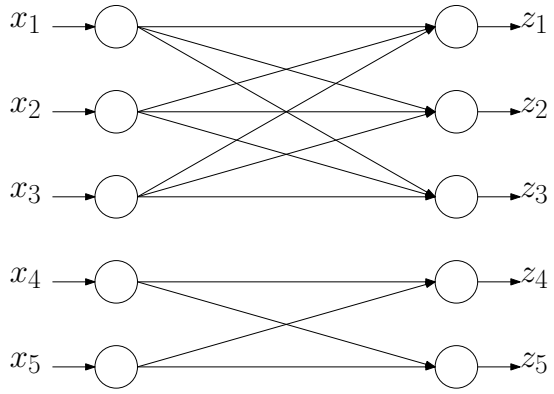
Suppose a model structure has been selected, the next step is to determine its model parameters so that it can give a good label for unlabelled signal in Table 3. One might think of a term 'learning'. Let the model learn from the labelled samples to get the parameters. But, what is the exact meaning of 'learning'? It is hard to tell. So, let us answer this question from another perspective. What is the purpose of learning with reference to the above signal labelling problem? For this, we can give an answer.

Definition 1 *The purpose of 'learning' is to let the model learn to give accurate labels for the labelled samples and at the same time to give 'good' labels for the unlabelled signals.*

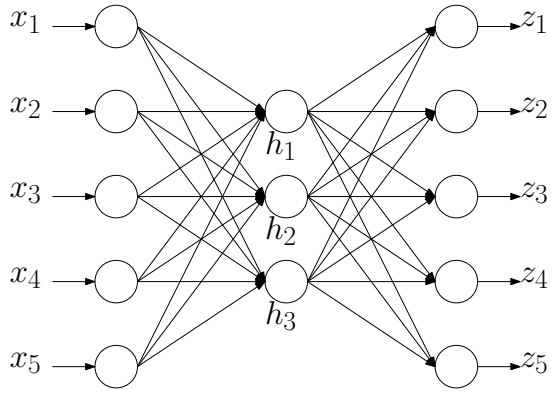
In accordance with the above definition, the purpose of learning has no different from the purpose of a *parametric estimation* method in statistics, a *function approximation* method or *interpolation* method in mathematics and a *model identification* method in control theory. So, a learning algorithm is essentially a parametric estimation method.



(a) Fully Connected



(b) Partial Connected



(c) Layered Structure

Figure 16: Graphical structures of a model – (a) fully connected, (b) partial connected and (c) layered structure.

Table 5: Labelling of the labelled signals by the model.

	(Input $\mathbf{x}$)					(Model Prediction)				
k	x_1	x_2	x_3	x_4	x_5	z_1	z_2	z_3	z_4	z_5
1	0.5	1	0	0	0	0.9	0.1	0	0	0
2	0	0	0.2	0.4	0	0	0.95	0	0	0.1
3	0	0.2	0	0.1	1	0	0.2	0.99	0.1	0
4	0.3	0	0.5	0	0.1	0	0	0	0.85	0
5	0	0.7	0	1.2	0.1	0.05	0	0.12	0	0.93

4.1.4 Goodness Measure $E(\mathbf{w})$

Again, here comes in another problem. In Definition 1, there are adjectives 'accurate' and 'good'. What if the model labels for the labelled samples as depicted in Table 5, should we say that the model is useless as the output values are deviated from the actual values of the labelled signals? So, what should be the measure for 'goodness'?

Problem 4 *What should be the mathematical definition of goodness measure for a model which is applied in the signal labelling problem as stated in Problem 2?*

There are many possible definitions for goodness measure. One typical definition is the mean squared error (MSE). Believe that I have mentioned in some where. With reference to the signal labelling problem, we let $\mathbf{x}_k$ be the input vector of the kth sample. The i^{th} element of $\mathbf{x}_k$ is denoted as x_{ki}. Its label is denoted by $\mathbf{z}_k$. The i^{th} element of $\mathbf{z}_k$ is denoted as z_{ki}.

Furthermore, we let $\mathbf{f}(\mathbf{x}, \mathbf{w})$ be the vector function corresponding to a given model which is applied in the signal labelling problem. The i^{th} element of $\mathbf{f}(\mathbf{x}, \mathbf{w})$ is denoted as $f_i(\mathbf{x}, \mathbf{w})$. Here $\mathbf{w}$ is called the parametric vector. Its elements are the parameters of the model. Finally, we let $E(\mathbf{w})$ be the mean squared error, i.e. the goodness measure. So, a mathematical definition of the goodness measure is given by

$$E(\mathbf{w}) = \frac{1}{5}\sum_{k=1}^{5}\left(\frac{1}{5}\sum_{i=1}^{5}\left(f_i(\mathbf{x}_k, \mathbf{w}) - z_{ki}\right)^2\right). \tag{8}$$

For a database with N labelled samples and the dimension of the outputs is n, the general form of the goodness measure is given by

$$E(\mathbf{w}) = \frac{1}{N}\sum_{k=1}^{N}\left(\frac{1}{n}\sum_{i=1}^{n}\left(f_i(\mathbf{x}_k, \mathbf{w}) - z_{ki}\right)^2\right). \tag{9}$$

With the above definition of goodness measure, the purpose of learning as stated in Definition 1 can be rewritten.

Definition 2 *The purpose of 'learning' is to let the model learn to give good labels for the labelled samples such that its goodness measure $E(\mathbf{w})$ is minimum and at the same time to give 'good' labels for the unlabelled signals.*

If we let $\mathbf{w}^*$ be the model with the minimum $E(\cdot)$, i.e. $E(\mathbf{w}) \geq E(\mathbf{w}^*)$ for all $\mathbf{w} \neq \mathbf{w}^*$, we can now define what is a learning algorithm.

Definition 3 *A learning algorithm is a method which is applied to search for the best model $\mathbf{w}^*$. Learning is simply the process of search.*

There are many definitions for goodness measure. Mean absolute error as stated in (10) is one alternative.

$$E(\mathbf{w}) = \frac{1}{N}\sum_{k=1}^{N}\left(\frac{1}{n}\sum_{i=1}^{n}|f_i(\mathbf{x}_k, \mathbf{w}) - z_{ki}|\right). \tag{10}$$

Various loss functions, like (11) and (12), are other alternatives.

$$E(\mathbf{w}) = \frac{1}{N}\sum_{k=1}^{N}\exp\left(\frac{1}{n}\sum_{i=1}^{n}|f_i(\mathbf{x}_k, \mathbf{w}) - z_{ki}|\right). \tag{11}$$

$$E(\mathbf{w}) = \exp\left(\frac{1}{N}\sum_{k=1}^{N}\exp\left(\frac{1}{n}\sum_{i=1}^{n}(f_i(\mathbf{x}_k, \mathbf{w}) - z_{ki})^2\right)\right). \tag{12}$$

4.1.5 Learning Algorithm Design

Now, we have the labelled samples. By convention, the set of labelled samples is denoted as $\mathcal{D}$. If a model structure $\mathbf{f}(\mathbf{x}, \mathbf{w})$ has been selected and a goodness measure $E(\mathbf{w})$ has been defined, the only problem left is the learning algorithm design.

Problem 5 *How to design a learning algorithm for the model $\mathbf{f}(\mathbf{x}, \mathbf{w})$ to learn from the labelled samples such that its goodness measure $E(\mathbf{w})$ is the minimum?*

To solve this problem, we can apply some ideas from optimization theory [27, 28]. For instance, gradient descent and Newton method have been two popular approaches for the learning algorithm design, equivalently the learning algorithm development.

Applying the idea of gradient descent, the learning algorithm is defined as follows :

$$\mathbf{w} \leftarrow \mathbf{w} - \mu\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}}, \tag{13}$$

where $\partial E(\mathbf{w})/\partial \mathbf{w}$ is the gradient vector and μ is a small positive number, say 0.1. μ is commonly called the step size. Usually, this step size decreases as the number of iteration increases.

Apply the idea of Newton method, the learning algorithm is defined as follows :

$$\mathbf{w} \leftarrow \mathbf{w} - \mu \left(\frac{\partial^2 E(\mathbf{w})}{\partial \mathbf{w}^2} \right)^{-1} \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}}, \tag{14}$$

where $\partial^2 E(\mathbf{w})/\partial \mathbf{w}^2$ is the second order derivative matrix called the Hessian matrix and μ is a small positive number.

One can see that the learning algorithms (13) and (14) are just two iterative algorithms. By the theory in gradient descent and Newton method, the learning algorithm either (13) or (14) can eventually get $\mathbf{w}^*$ (i.e. the best model) after enough number of iterations.

Problem 6 *One condition for the applicability of the learning algorithms (13) and (14) is that $E(\mathbf{w})$ is differentiable. How about if $E(\mathbf{w})$ is non-differentiable, like (10) and (11)? Nevertheless, it could happen when the model is non-differentiable, like (6) and (7).*

A Solution to Problem 6 : *Brute-Force Search – Try all possible combinations of $\mathbf{w}$!* (Are you kidding me?)

A Solution to Problem 6 : *Gradient-Free [29, 30] – Replace the gradient vector by numerical gradient vector.*

$$\mathbf{w} \leftarrow \mathbf{w} - \mu \frac{E(\mathbf{w} + \Delta\mathbf{w}) - E(\mathbf{w} - \Delta\mathbf{w}))}{2\Delta\mathbf{w}}, \tag{15}$$

where

$$\begin{aligned} & \left(\frac{E(\mathbf{w} + \Delta\mathbf{w}) - E(\mathbf{w} - \Delta\mathbf{w})}{2\Delta\mathbf{w}} \right)_i \\ = \quad & \frac{E(w_1, \cdots, w_i + \Delta w_i, \cdots, w_{N_w}) - E(w_1, \cdots, w_i - \Delta w_i, \cdots, w_{N_w})}{2\Delta w_i}. \end{aligned}$$

Here, N_w is the number of model parameters and Δw_i, for $i = 1, \cdots, N_w$, is a small positive number.

Clearly, there are other solutions for the Problem 6. Most of them belongs to an area called random search or stochastic search methods. Simulated annealing [31, 32] and Markov Chain Monte Carlo (MCMC) method [33] are two examples.

4.1.6 Validation from Testing Set

Even if we have designed a learning algorithm and even if we have got the best model $\mathbf{w}^*$, it does not guarantee that this well-trained model will give good label for the unlabelled samples. This is a fact.

Fact 1 *Even if we have got the best model $\mathbf{w}^*$, where $\mathbf{w}^* = \arg\min_{\mathbf{w}} \{E(\mathbf{w})\}$, no one knows how good the model will be in labelling the unlabelled samples.*

So, it comes to another problem.

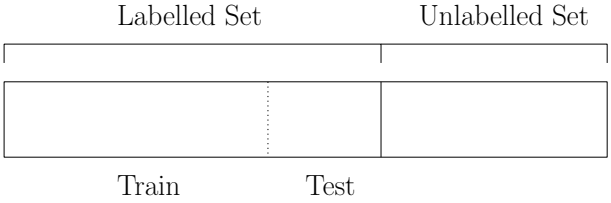


Figure 17: Training set and testing set.

Problem 7 *Would there be any method that can be applied to anticipate the performance of a well-trained model in labelling the unlabelled samples?*

One method is to partition the labelled samples into two sets – training set and testing set, see Figure 17. The samples in the training set are used for getting a model and the samples in the testing set are used for validation. Their goodness measures are defined as follows :

$$E_{Train}(\mathbf{w}) = \frac{1}{N_{Train}} \sum_{k=1}^{N_{Train}} \left(\frac{1}{n} \sum_{i=1}^{n} (f_i(\mathbf{x}_k, \mathbf{w}) - z_{ki})^2 \right). \quad (16)$$

$$E_{Test}(\mathbf{w}) = \frac{1}{N_{Test}} \sum_{k=1}^{N_{Test}} \left(\frac{1}{n} \sum_{i=1}^{n} (f_i(\mathbf{x}'_k, \mathbf{w}) - z'_{ki})^2 \right). \quad (17)$$

Next, the learning algorithm is applied to the training set only, i.e.

$$\mathbf{w} \leftarrow \mathbf{w} - \mu \frac{\partial E_{Train}(\mathbf{w})}{\partial \mathbf{w}}. \quad (18)$$

The testing set pretends to be the unlabelled set.

After training, we have two goodness measures $E_{Train}(\mathbf{w}^*)$ and $E_{Test}(\mathbf{w}^*)$. $E_{Train}(\mathbf{w}^*)$ indicates how fit the model does for the samples in the training set. $E_{Test}(\mathbf{w}^*)$ indicates how fit the model does for the samples in the testing set. As the samples in the testing set do not involve in the learning process, we could then use $E_{Test}(\mathbf{w}^*)$ to anticipate the performance of the model in labelling the unlabelled samples.

Fact 2 *(1)* $E_{Train}(\mathbf{w}^*) < E_{Test}(\mathbf{w}^*)$. *(2)* $E_{Train}(\mathbf{w}^*)$ *could be close to zero. (3)* $E_{Test}(\mathbf{w}^*)$ *could be large even if* $E_{Train}(\mathbf{w}^*) = 0$.

The last one is a common phenomena called **overfitting**. The model fits too well to the samples in the training set. If a model overfits, its performance on the labelling the unlabelled samples will be doubtful.

4.1.7 Overfitting

Overfitting is not acceptable and should be avoided. To make the point clear, here gives you a simple example by using the labelled samples depicted in Table 4. Let the first four samples be the training samples and the fifth sample be

Table 6: Training set and testing set of labelled samples.

Training Set

	(Input $\mathbf{x}$)					(Output $\mathbf{z}$)				
k	x_1	x_2	x_3	x_4	x_5	z_1	z_2	z_3	z_4	z_5
1	0.5	1	0	0	0	1	0	0	0	0
2	0	0	0.2	0.4	0	0	1	0	0	0
3	0	0.2	0	0.1	1	0	0	1	0	0
4	0.3	0	0.5	0	0.1	0	0	0	1	0

Testing Set

	(Input $\mathbf{x}$)					(Output $\mathbf{z}$)				
k	x_1	x_2	x_3	x_4	x_5	z_1	z_2	z_3	z_4	z_5
5	0	0.7	0	1.2	0.1	0	0	0	0	1

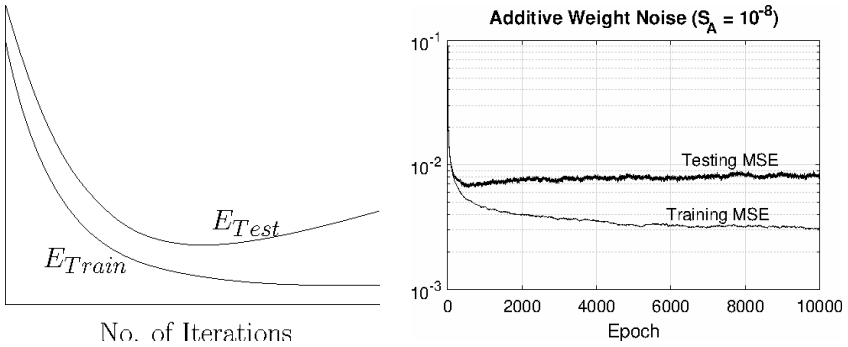
the testing sample. The model as defined in (6) is applied for the labelling task. It can be verified that the model as stated below in (19) will give zero training MSE for the training samples. However, its output for the testing sample is $[0, 0, 0, 0, 0]$. The testing MSE is $1/5$.

$$\begin{aligned}
\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix} &= \begin{bmatrix} \phi\left(x_1 + x_2 - 5x_3 - 5x_4 - 5x_5\right) \\ \phi\left(-5x_1 - 5x_2 + x_3 + x_4 - 5x_5\right) \\ \phi\left(-5x_1 + x_2 - 5x_3 + x_4 + x_5\right) \\ \phi\left(x_1 - 5x_2 + x_3 - 5x_4 + x_5\right) \\ \phi\left(-x_1 - x_2 - x_3 - x_4 - x_5\right) \end{bmatrix}, \\
\phi(u) &= \begin{cases} 1 & \text{if } u \geq 0 \\ 0 & \text{if } u < 0, \end{cases}
\end{aligned} \tag{19}$$

Even if we pick other four samples as the training samples, similar result will be obtained. The well-trained model fits too well to the training samples. But, it fails to recognize the testing sample. It should be noted that the models obtained by different sets of training samples are different.

Train				Test	E_{Train}	E_{Test}
1	2	3	4	5	0	1/5
1	2	3	5	4	0	1/5
1	2	4	5	3	0	1/5
1	3	4	5	2	0	1/5
2	3	4	5	1	0	1/5

One might think that the problem is due to the number of training samples. Yes, it is one reason. As long as the number of training samples is finite, this problem could appear. Normally, the number of training samples has to be larger than the number of model parameters. For the model as defined in (6), there are 30 model parameters. So, the number of training samples should be at least 30. In practice, the number should be 5×30 although there is no guarantee



(a) Conceptual (b) Actual

Figure 18: The learning curves. (a) Conceptual diagram. (b) Actual learning curves obtained from a research.

that overfitting can be avoided. In principle, overfitting could possibly disappear if the number of training samples is infinite.

4.1.8 Actual Learning Objective Function $V(\mathbf{w})$

As long as the number of training samples is finite, overfitting could appear. Figure 18(a) shows an example of learning curves. The curve on top (resp. below) shows the change of $E_{Test}(\mathbf{w})$ (resp. $E_{Train}(\mathbf{w})$) after each iteration. Figure 18(b) shows an actual learning curves obtained from a research[15]. For the training MSE, the value decreases as the learning goes on. For the testing MSE, it first decreases as the number of iteration increases. After some point, its value raises. In this moment, the model starts to overfit. The effect of overfitting increases further as the learning continues.

As using mean squared error for learning will lead to unavoidable overfitting, it comes to another problem that we need to deal with.

Problem 8 *How to obtain a model that is able to fit well to the training samples as well as the testing samples?*

Equivalently, the problem is how to obtain a model $\mathbf{w}^*$ with small $E_{Train}(\mathbf{w}^*)$ and small $E_{Test}(\mathbf{w}^*)$.

Many techniques have been developed for alleviating the problem of overfitting. One simple technique is called *early stopping*[23, pp.203-205] – stops training once the value of E_{Test} starts to raise. Another technique is to define a new cost function for training. This new cost function is usually called the learning objective function. My usual practice, this objective function is denoted as $V(\mathbf{w})$. The design of an objective function is usually simple – adding

[15] A MLP is trained to recognize handwritten digits.

penalty term. That is to say,

$$V(\mathbf{w}) = E(\mathbf{w}) + \alpha\mathcal{R}(\mathbf{w}), \tag{20}$$

where $\mathcal{R}(\mathbf{w})$ is a penalty term and α is a small positive number. Two common choices for $\mathcal{R}(\mathbf{w})$ are $\|\mathbf{w}\|_2^2$ and $\|\mathbf{w}\|_1$. Recall that $\mathbf{w}$ is the parametric vector. Its elements are the model parameters. There are N_w number of parameters.

$$\begin{aligned} \|\mathbf{w}\|_2^2 &= w_1^2 + w_2^2 + \cdots + w_{N_w}^2. \\ \|\mathbf{w}\|_1 &= |w_1| + |w_2| + \cdots + |w_{N_w}|. \end{aligned}$$

Thus, the learning algorithm as stated in (13) is modified as follows :

$$\mathbf{w} \leftarrow \mathbf{w} - \mu\frac{\partial V(\mathbf{w})}{\partial \mathbf{w}}. \tag{21}$$

Again, the idea of brute-force-search or gradient-free might be applied in the development of the learning algorithm if $V(\mathbf{w})$ is not differentiable.

4.2 Issues on Computational Complexity

Computational complexity is an important factor to reveal the computational burden of an algorithm applied in solving a problem, like the gradient descent algorithm applied in learning. Roughly speaking, computational complexity could be perceived as the total number of arithmetic operations needed to solve a problem. The more complex the problem, the time required for a general purpose computer to solve it will be longer. The time required for a complex task could be in term of day and even week. Computational complexity of an algorithm has direct effect on its suitability for real-time[16] application, like auto driving. Thus, some issues on the computational complexity of an AI/ML model applying in prediction and the computational complexity of training an AI/ML model have to be introduced.

4.2.1 Complexity of Prediction in an Iteration

Consider an AI/ML model is trained for handwritten digit recognition. The training set has N training samples. During training, the model first predicts the labels of the training samples $\mathbf{f}(\mathbf{x}_k, \mathbf{w}(t))$ for $k = 1, \cdots, N$. Based upon their errors $(\mathbf{f}(\mathbf{x}_k, \mathbf{w}(t)) - \mathbf{z}_k)$ for $k = 1, \cdots, N$, the parametric vector $\mathbf{w}(t)$ is updated to $\mathbf{w}(t+1)$ by the learning algorithm, i.e.

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \Delta\mathbf{w}(t), \text{ where } \Delta\mathbf{w}(t) = -\mu\frac{\partial V(\mathbf{w}(t))}{\partial \mathbf{w}}.$$

[16] If a problem has to be solved in real-time, it has to be solved immediately without any time lag. For example, driving a car on a highway is a real-time problem. Any delay response could cause accident.

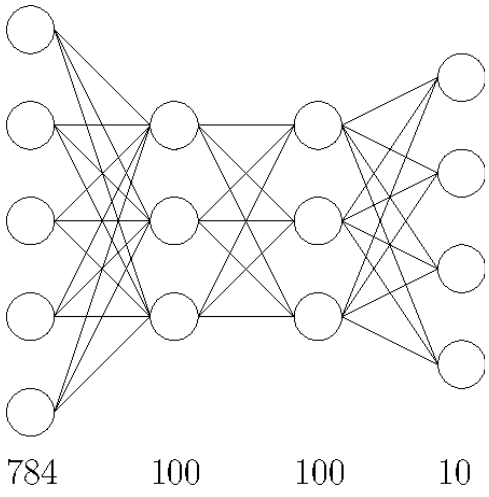


Figure 19: The MLP with two hidden layers.

Then, the model with $\mathbf{w}(t+1)$ predicts the labels of training samples again. The parametric vector is updated again.

$$\rightarrow \underbrace{\{\mathbf{f}(\mathbf{x}_k, \mathbf{w}(t)), \mathbf{z}_k\} \rightarrow \Delta\mathbf{w}(t)}_{t \text{ iteration}} \rightarrow \underbrace{\{\mathbf{f}(\mathbf{x}_k, \mathbf{w}(t+1)), \mathbf{z}_k\} \rightarrow \Delta\mathbf{w}(t+1)}_{(t+1) \text{ iteration}} \rightarrow$$

To calculate the per-iteration computational complexity, let me assume that the model is a MLP with the following specification.

- Number of input nodes is 784 (m).
- Number of output nodes is 10 (n).
- Two hidden layers, each has 100 nodes. $N_{L_1} = 100$ and $N_{L_2} = 100$
- The transfer function of the output nodes and hidden nodes is sigmoid.
- Signals from the input nodes are fed to all the nodes in the first hidden layer.
- The outputs of the nodes in the first hidden layer are fed to all the nodes in the second hidden layer.
- The outputs of the nodes in the second hidden layer are fed to all the output nodes.

This model is usually denoted in shorthand as '784-100-100-10'. Its graphical structure is shown in Figure 19.

- For a node in the first hidden layer, it requires 784 floating point multiplications and 784 floating point additions to generate its output.

- For a node in the second hidden layer, it requires 100 floating point multiplications and 100 floating point additions to generate its output.
- For a node in the output layer, it requires 100 floating point multiplications and 100 floating point additions to generate its output.

Normally, the number of CPU clock cycles for a floating point multiplication is far more than the number of CPU clock cycles for a floating point addition. If we use the notations, the total number of floating point operations (FLOP) for giving a prediction for a sample is given by

$$\text{No. of FLOP Per Sample} = m \times N_{L_1} + N_{L_1} \times N_{L_2} + N_{L_2} \times n.$$

It is about the same number as the number of model parameters N_w.

$$N_w = (m+1) \times N_{L_1} + (N_{L_1}+1) \times N_{L_2} + (N_{L_2}+1) \times n.$$

For a MLP with structure 784-100-100-10, the total number of FLOP and the total number of model parameters are given by

$$\text{No. of FLOP Per Sample} = 89400 \text{ and } N_w = 89610.$$

Therefore, the total number of FLOP for predictions of all the samples in a training set is approximately $N \times N_w$. N is the number of samples in the training set.

4.2.2 Complexity of Learning in an Iteration

Next, let us analyze the number of FLOP for updating $\mathbf{w}$. To update $\mathbf{w}$ from $\mathbf{w}(t)$ to $\mathbf{w}(t+1)$, the computer needs to calculate the gradient vector $-\mu\partial V(\mathbf{w}(t))/\partial \mathbf{w}$. In essence, the values of $-\mu\partial V(\mathbf{w}(t))/\partial w_i$ for $i = 1, \cdots, N_w$ are calculated. To calculate the value of $\partial V(\mathbf{w}(t))/\partial w_i$, it requires βN FLOP. Here β is an integer constant. Normally, this number is no more than 10. Therefore, the total number of FLOP for updating $\mathbf{w}$ in an iteration is proportional to $N \times N_w$.

4.2.3 Computational Complexity of Learning

After all, the total number of FLOP to be performed by a CPU in one iteration is proportional to $N \times N_w$. By using the Big-O notation, the per iteration complexity is $\mathcal{O}(N \times N_w)$. If we let the total number of iteration is N_{it}. The computational complexity of learning is $\mathcal{O}(N \times N_w \times N_{it})$. Let me plug in the actual numbers for the research result as shown in Figure 18(b).

$$N = 60000, \quad N_w = 89610, \quad N_{it} = 10000.$$

The total number of FLOP is approximately 5.4×10^{13}. Take Core i7 4770K (Haswell) as an example, it could achieve 27.6 GFLOPS (Giga FLOP per second)[17]. The time spent in training the MLP is 54000/27.6 seconds, which is less than an hour.

[17] From SiSoftware Official Live Ranker.

In reality, the time spent in training is much more than an hour. There are a couple reasons for this.

- One reason is that the CPU is not just installed to support the AI/ML application. It needs to support other software, such as the operating system and the Internet access software.
- Another reason is the runtime environment of the AI/ML application software. Many AI/ML applications are implemented by interpreted languages, like Python and Matlab. The interpreter acts as the interface between the program and the operating. As a result, running the program for learning would need to have intensive interactions between the interpreter and the operating system.
- For such a large scale database, the samples have to be stored in the main memory. The data transferred back and forth amongst the CPU, the RAM and the main memory takes extra time.

All these reasons will cause overhead on the computational time for the learning process. The actual time taken for a learning process is definitely more than expected.

One more point should be noted. The number of iterations is always depended on the problem. For some complicated problems, the number of iterations could be much more, say 10^9. In this regards, the computational complexity of a learning process will be high.

In summary, the complexity of the learning process is determined by six factors – (1) the size of the training samples, (2) the number of model parameters in $\mathbf{f}(\mathbf{x}, \mathbf{w})$,(3) the definition of the goodness measure $E(\mathbf{w})$ (equivalently, the definition of the learning objective function $V(\mathbf{w})$), (4) the learning algorithm for training the model, (5) the problem nature and (6) the computer system (resp. computing device) architecture.

4.2.4 Computational Complexity of the Model in Use

Clearly, the computational complexity of the learning process is very high. However, the computational complexity of the model in use is pretty low. Its complexity is proportional to the number of model parameters N_w. Compared with the computational complexity of learning, i.e. $\mathcal{O}(N \times N_w \times N_{it})$, it is nothing. Therefore, many intelligent services available in smartphones are not able to have on-line learning[18] capabilities. The processors in a smartphone are unable to handle such high computational complexity learning task. The model in use is a pre-trained model which is either developed in-house or downloaded from the Internet. These pre-trained models have a number of pitfalls.

- The model is not 100% accurate.
- The model could inherit cultural aspect of its developer.

[18]The learning is conducted in a regular basis, such as hourly basis and daily basis.

- The model could be vulnerable to adversarial attack.
- The model cannot be personalized.

These factors could cause safety concerns on the use of these models.

To make learning possible, one solution is to connect the smartphone to a cloud platform. Learning process is taken place in the cloud. The smartphone acts as a data collector. While the model is in use, the smartphone collects relevant data and then sends them to the cloud. The learning software in the cloud will then update the model parameters based on the data received. Once the model has been updated, the parameters of the new model are then sent to the smartphone. Next time around, the application software in the smartphone will use the new model for the user.

4.3 Issues on Data Collection

Now, developers face another difficult decision – what specific data should be collected.

Problem 9 *How many number of features should be defined? Which words should be included in the bag-of-words?*

4.3.1 Data Collection

The first decision could be determined by the current technologies in **signal processing**. The second decision would have to be determined by the application domain of the system. Take *Jorman* as an example. He is designed to assist me to command the Windows OS. So, the set of words must be related to the OS commands. Application systems like Amazon Echo and Google Home would have predefined another sets of words. In the future, the number of features (resp. words) to be included in the set (resp. bad-of-words) must be increased. One reason is due to the advancement of the processing power of a CPU or GPU. The other reason would tentatively due to the release of new machine learning models for feature extraction and word labelling.

4.3.2 Use of Public Database

Today, many research labs from both industrial and academic have released their databases to the public. Researchers could download the databases for the development of core AI/ML technologies.

- Google has released many databases for the research community. Google has released a voice database called AudioSet[19], an image database[20], a

[19] `https://research.google.com/audioset/`.
[20] `https://opensource.google/projects/open-images-dataset`.

news database[21], a video database called YouTube8M[22][34, 35] and many others[23] for the research community.

- Stanford Computer Vision Lab has released an image database called ImageNet[24][36, 37].
- Princeton has released a database called WorddNet[25] which consists of natural text messages.
- An early audio database called TIMIT was created by a project with collaboration of Massachusetts Institute of Technology, SRI International and Texas Instruments. This database can now be downloaded from the Linguistic Data Consortium homepage[26]. This database has been used for early speech recognition technology development.
- Facebook has released a very interesting database – Deepfake Detection Challenge Dataset[27]. It consists of 124,000 videos (as in August 2020) which are not real. They are fake videos which are generated by some AI/ML algorithms. The purpose of its release is to foster the research community to develop AI/ML algorithms to counter-attack those fake news and fake video.
- The Center for Machine Learning and Intelligent Systems at the Bren School of Information and Computer Science, University of California, Irvine has long been maintaining a repository[28] for AI/ML researches. The repository has contained numerous databases for download.

All these databases play important roles in the AI/ML advancement.

4.4 On-Line Learning – Personalization

In Section 4.2, I have stressed that computational complexity of a learning process is usually high. For the applications, like translation and object recognition, the models are very complex. So, on-line learning is infeasible to be conducted in a smartphone or a desktop computer. However, there are certain applications having on-line learning. The models being used in these applications are generally simple. Thus, the processor in a smartphone is powerful enough to handle the learning task. Here are a few examples.

- Many voice assistance, like Siri, is able to learn on-line from the user speeches his/her speaking style – personalized voice signature. For instance, the voice assistant is able to re-adjust the time span for 'sound'

[21] `https://opensource.google/projects/kaggle-news-lab-dataset`.
[22] `https://research.google.com/youtube8m/`.
[23] `https://opensource.google/projects/list/databases`.
[24] `http://www.image-net.org/`.
[25] `https://wordnet.princeton.edu/`.
[26] `https://catalog.ldc.upenn.edu/LDC93S1`.
[27] `https://ai.facebook.com/datasets/dfdc/`.
[28] `http://archive.ics.uci.edu/ml/index.php`.

segmentation. For someone who talks slow, the time span would be elongated. For someone who talks fast, the time span would be shortened. By that, the voice assistant could improve its performance in voice recognition. The voice assistant could use this personalized voice signature as a biometric user identification.

- iOS QuickType (predictive text engine) is an application that we use almost everyday. Every time we would like to type a message to a friend, suggested words will appear on top of the keyboard. These words are generated by the QuickType. QuickType is able to learn on-line from the user typing habit to revise some model parameters. So that, the recommended words to be generated would be much fitting to the user need.

- Some auto driving assistants are able to learn from the drivers their driving habits and then fine-tune the control systems to assist the drivers. Note that the system simply assists the driver. It does not take over the control from the driver. After some period of time, the assistant and the control system together would behave like the driver himself/herself in driving the car. If voice assistant has also been installed in the car, the voice assistant will learn the personalized voice signature of the driver. So that, the voice assistant could better recognize the voice commands.

One important advantage of on-line learning is to make the application much more personalized.

4.5 Core AI/ML Technologies Development – Research

Recall from a complicated application with translation of a Chinese voice command to an English voice command. Besides, the command is to ask the Map server to find the direction for the user to go to a restaurant for a date before a specific time, see Figure 20. The core technologies supporting this application are (i) the voice-to-text technology, (ii) the text-to-voice technology, (iii) the path finding algorithm in the Map server and (iv) the NLU technology in the NLU server. One should be aware that developing a core AI/ML technology is not an easy task. It is a research.

From 1960 to 2019, there had been an increasing number of researches on AI/ML, as depicted in Table 7 and as shown in Figure 21 had been increased could be reflected from the number of books and papers published in each year. The number is obtained by searching Google Scholar with the following command and the year selected.

```
allintitle: "deep learning" OR "neural network"
          OR "artificial intelligence" OR "machine learning"
```

In this sixty years, AI/ML research seemed to have gone through four active periods. The first period is before the early 1980s. The second period is from the early 1980s to 2001. The third period is from 2002 to 2014. The forth period is from 2015 to today. The number of publications in each year started from 2

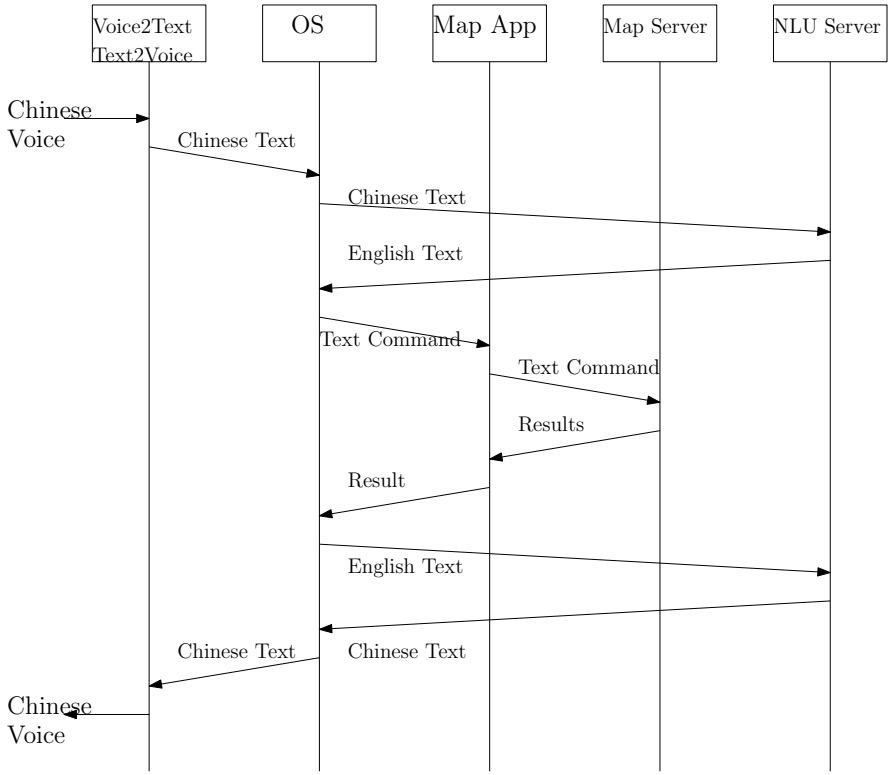


Figure 20: A working principle behind a complicated service with translation. Note that the Map Server and the NLU server are located in a remote cloud.

Table 7: Number of books and papers in AI/ML published from 1960-2019.

Year	No.	Year	No.	Year	No.	Year	No.
2019	17000	2004	4960	1989	962	1974	51
2018	16700	2003	4120	1988	808	1973	39
2017	16300	2002	3570	1987	556	1972	23
2016	15500	2001	2920	1986	422	1971	31
2015	12600	2000	3060	1985	278	1970	85
2014	10200	1999	3050	1984	250	1969	20
2013	10300	1998	2770	1983	160	1968	10
2012	10100	1997	2950	1982	80	1967	6
2011	8630	1996	2870	1981	74	1966	9
2010	9360	1995	2870	1980	71	1965	4
2009	8590	1994	2380	1979	46	1964	5
2008	7010	1993	2450	1978	56	1963	6
2007	6560	1992	2260	1977	42	1962	11
2006	6480	1991	1530	1976	51	1961	3
2005	4640	1990	1340	1975	45	1960	2

in year 1960 and increased to 17000 in year 2019. The period from 1960 to the early 1970s is also called the AI Winter.

4.5.1 Number of Core-Technology Researches

If each publication presents a new technology in AI/ML, the total number of technologies developed in the last sixty years is already enormous. It is just the number of successful cases. If the number of patents and the number of failure cases are included, the number would have to be multiplied by 10.

These numbers, from 1960 to 2019, came from the efforts of many independent researchers and research groups around the world who proposed and experimented their ideas in AI/ML. After a number of attempts and redesigns, some models have eventually been demonstrated with success in certain applications. They are thus applied for real-life applications, i.e. technical transfer.

$$\textbf{Core Technolgy Development} \xrightarrow{\text{Transfer}} \textbf{Commercial Product}$$

Clearly, successful technology transfer is the ultimate goal of a core technology development.

4.5.2 Motivations for a Development Project

The motivation for the development of a core AI/ML technology could come from different perspectives. One common motivation is to improve the standard of living and the standard of working environment [38].

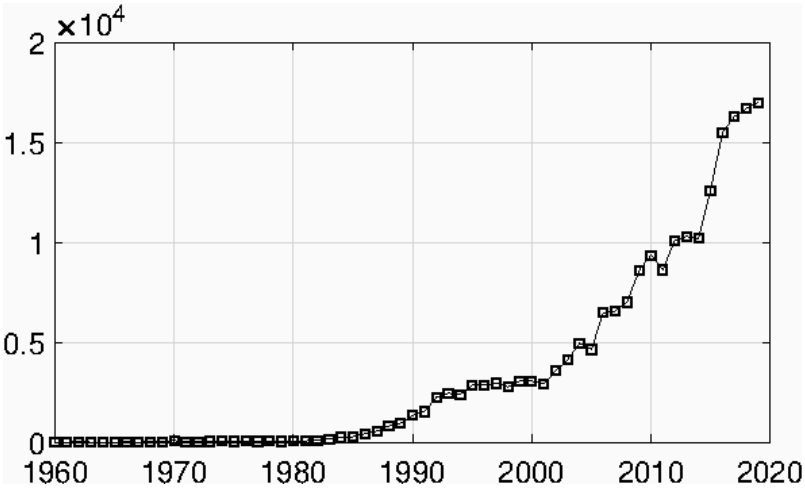


Figure 21: Number of books and papers in AI/ML published from 1960-2019.

The core technologies developed for Apple iPhone are clearly applied to improve our quality of lives. The core technologies developed for Google Translate could improve the quality of a secretary working on the tasks related to international business. While the core technologies were not intelligent, Ford Motor had applied technologies from electrical engineering to implement the electric motor-driven assembly line which improved the efficiency of car production in the early 20 century [38].

Another common motivation is just because of academic interest – to postulate mathematical models for the mechanisms of information processing and biological learning in our brains. The Perceptron [39, 40], Cognitron [14] and Neocognitron [15] are three notable examples.

4.5.3 Types of Development Projects

Usually, a core AI/ML technology development project could be classified into one of the following types.

- Developing a new model for solving a new problem.
- Developing a new model for solving an existing problem.
- Modification of an existing model to an advanced version.
- Application of an existing model for solving a new problem.
- Hardware advancement.
- New hardware development.

Developing a new model or a new hardware is the most difficult project. Modification of an existing AI/ML model to an advanced version is a relatively easier project.

4.5.4 New Model Development

Developing new model for solving a new problem or an existing problem requires two important factors: (i) the talent of the developer or the researcher, and (ii) good mathematical background for a number of theoretical analysis. Here, another two factors have not been mentioned – the level of English proficiency and the programming skill.

- Data collection for the development project.
- Extensive computer simulation studies (Empirical analysis) – To demonstrate if the new model with the corresponding learning algorithm is able to solve the problem and identify under what conditions the model work.
- Theoretical analysis on properties of the new model if it is applied to solve the problem.
- Theoretical analysis on the conditions in which the learning algorithm is applicable to train the model to solve the problem.
- Theoretical analysis on the conditions in which the learning algorithm is definitely not applicable to train the model to solve the problem.
- Theoretical analysis on the robustness of the learning algorithm if there is any perturbation on the model.

To conduct computer simulation, good programming skill and good logic in program design are necessary. To accomplish the theoretical analysis, knowledge of mathematics to the competence level is definitely required.

References

[1] S. Spangler, A. D. Wilkins, B. J. Bachman, M. Nagarajan, T. Dayaram, P. Haas, S. Regenbogen, C. R. Pickering, A. Comer, J. N. Myers *et al.*, "Automated hypothesis generation based on mining scientific literature," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 1877–1886.

[2] Y. Gil, D. Garijo, V. Ratnakar, R. Mayani, R. Adusumilli, H. Boyce, and P. Mallick, "Automated hypothesis testing with large scientific data repositories," in *Proceedings of the Fourth Annual Conference on Advances in Cognitive Systems (ACS)*, vol. 2, 2016, p. 4.

[3] J. Sybrandt, M. Shtutman, and I. Safro, "Moliere: Automatic biomedical hypothesis generation system," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1633–1642.

[4] S. J. Wilson, A. D. Wilkins, M. V. Holt, B. K. Choi, D. Konecki, C.-H. Lin, A. Koire, Y. Chen, S.-Y. Kim, Y. Wang *et al.*, "Automated literature mining and hypothesis generation through a network of medical subject headings," *BioRxiv*, p. 403667, 2018.

[5] B. Writer, *Lithium-Ion Batteries : A Machine-Generated Summary of Current Research*. Springer, 2019.

[6] D. Marr, "A theory of cerebellar cortex," *Journal of Physiolgy*, vol. 202, no. 2, pp. 437–470, 1969.

[7] ——, "A theory for cerebral neocortex," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 176, no. 1043, pp. 161–234, 1970.

[8] D. Marr and T. Poggio, "A computational theory of human stereo vision," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 204, no. 1156, pp. 301–328, 1979.

[9] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 207, no. 1167, pp. 187–217, 1980.

[10] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. John Wiley & Sons, 2012.

[11] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of Physiology*, vol. 148, no. 3, p. 574, 1959.

[12] ——, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of Physiology*, vol. 160, no. 1, p. 106, 1962.

[13] ——, "Receptive fields and functional architecture of monkey striate cortex," *The Journal of Physiology*, vol. 195, no. 1, pp. 215–243, 1968.

[14] K. Fukushima, "Cognitron: A self-organizing multilayered neural network," *Biological Cybernetics*, vol. 20, no. 3-4, pp. 121–136, 1975.

[15] ——, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, pp. 193–202, 1980.

[16] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[22] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," *arXiv preprint arXiv:1602.07261*, 2016.

[23] S. Haykin, *Neural Networks and Learning Machines*. Pearson Education, 2010.

[24] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[25] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 2nd ed. MIT press, 2018.

[26] M. P. Deisenroth, A. A. Faisal, and C. S. Ong, *Mathematics for Machine Learning*. Cambridge University Press, 2020.

[27] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, 4th ed. Springer, 2016.

[28] S. S. Rao, *Engineering Optimization: Theory and Practice*. John Wiley & Sons, 2019.

[29] J. Kiefer, J. Wolfowitz *et al.*, "Stochastic estimation of the maximum of a regression function," *The Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 462–466, 1952.

[30] J. C. Spall *et al.*, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Transactions on Automatic Control*, vol. 37, no. 3, pp. 332–341, 1992.

[31] H. Szu and R. Hartley, "Fast simulated annealing," *Physics Letters A*, vol. 122, no. 3-4, pp. 157–162, 1987.

[32] P. J. Van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated Annealing: Theory and Applications*. Springer, 1987, pp. 7–15.

[33] C. Andrieu, N. De Freitas, A. Doucet, and M. I. Jordan, "An introduction to mcmc for machine learning," *Machine Learning*, vol. 50, no. 1-2, pp. 5–43, 2003.

[34] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," *arXiv preprint arXiv:1609.08675*, 2016.

[35] J. Lee, W. Reade, R. Sukthankar, G. Toderici *et al.*, "The 2nd YouTube-8M large-scale video understanding challenge," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 0–0.

[36] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 248–255.

[37] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[38] Ford Motor Company, *Factory Facts from Ford*. Detroit, Michigan, USA: Ford Motor Company, 1915.

[39] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological Review*, vol. 65, no. 6, p. 386, 1958.

[40] ——, *Principles of Neurodynamics: Perceptions and the theory of brain mechanisms*. Spartan, 1962.

[41] W. James, *The Principles of Psychology*. Henry Holt and Company, 1890, vol. 1.

[42] ——, *The Principles of Psychology*. Henry Holt and Company, 1890, vol. 2.

[43] B. F. Skinner, *The Behavior of Organisms: An experimental analysis*. Appleton-Century, 1938.

[44] B. F. Skinner and C. B. Ferster, *Schedules of Reinforcement*. B. F. Skinner Foundation, 1957.

[45] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[46] H. Landahl, W. S. McCulloch, and W. Pitts, "A statistical consequence of the logical calculus of nervous nets," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 135–137, 1943.

[47] N. Wiener, *Cybernetics or Control and Communication in the Animal and the Machine*. MIT press, 1948.

[48] D. Hebb, *The Organization of Behavior*. Wiley, New York, 1949.

[49] F. A. Hayek, *The Sensory Order: An inquiry into the foundations of theoretical psychology*. University of Chicago Press, 1952.

[50] B. Farley and W. Clark, "Simulation of self-organizing systems by digital computer," *Transactions of the IRE Professional Group on Information Theory*, vol. 4, no. 4, pp. 76–84, 1954.

A Early Findings and Postulations

Development of various AI/ML models has their roots from the early experimental findings and postulations on animal/human behavioral development. In this appendix, some notable findings and postulations will be presented.

A.1 Behavioral Association

One early postulation is from the studies conducted by some psychologists in the nineteenth century. Psychologist William James postulated from their findings that behavioral response of a human to a stimulus is not random. It is based upon his/her the associations among the responses and the actual outcomes from his/her past experiences [41, 42]. Each association, like a memory, could let a human to have an expectation on the outcome of a response to a stimulus.

A.2 Behavioral Reinforcement

In early twentieth century, psychologist Burrhus Frederic Skinner conducted a number of experimental researches on operant conditioning to animals. As a result, Skinner postulated that human action as dependent on consequences of previous actions. If the outcome to a response to a stimulus is bad, the chance that the response will be repeated would be decreased. If the outcome to a response to a stimulus is good, the chance that the response will be repeated would be elevated. This is the principle of reinforcement [43, 44].

A.3 Threshold Logic Neuron Model

In early 1940s, Warren S. McCulloch and Walter Pitts modelled a neuron as a threshold logic unit and presented a mathematical foundation on neuronal networks with threshold logic neurons [45, 46]. Their results influenced their collaborator Norbert Wiener to postulate that the intelligent behavior of a human is the result of a number of feedback mechanisms occurred in that human [47].

A.4 Association in Neuronal Level

Another notable postulation is from psychologist Donald Hebb in his seminal monograph entitled *The Organization of Behavior* [48]. This monograph was published in 1949. It is a year later than the publication of the book *Cybernetic* [47] authored by Norbert Wiener. In *The Organization of Behavior*, Donald Hebb postulated that the association between two neighbor neurons will be strengthened if both neurons fire at the same time. In contrast to the postulation from William James in [41, 42], it is the first time in the history that the postulation of association is in the neuronal level.

A.5 Neuronal Map Postulation

One more postulation is from an economist Friedrich A. Hayek, who is also the recipient of the 1974 Nobel Memorial Prize in Economic Sciences, in a monograph entitled *The Sensory Order* [49]. In the monograph, Hayek postulated that our mental structure has many sensory maps. The sensory information perceived will influence a brain to form mental maps which capture the orders of the sensory information – sensory orders.

A.6 Influences

These early postulations and findings have laid the very foundation on human behavior development. A neuron could be mathematically defined as a threshold logic unit (TLU), so-called the McCulloch-Pitts neuron. Learning is a behavioral adaptation process – a process of *association*, a process of *respondent conditioning*, a process of *operant conditioning – reinforcement* or an *adaptation process with feedback mechanism*. While Warren S. McCulloch and Walter Pitts had laid the theoretical foundation on the networks of TLU neurons [45] in 1940s, theoretical foundation on the learning process was still unknown.

Starting from 1950, researchers started to postulate various mathematical models for human learning. Some of them have the psychological or biological origins. Computer simulation on self-organizing systems were investigated [50]. Hardware neural analog systems were finally realized.

- Marvin L. Minsky, *A Neural-Analogue Calculator Based upon a Probability Model of Reinforcement*, Harvard University Psychological Laboratories, Cambridge, Massachusetts, January 8, 1952.

It describes the hardware of the a machine named Stochastic Neural Analog Reinforcement Calculator (SNARC), the first artificial neural network that applied the idea of reinforcement to design the learning algorithm. Theoretical foundation on the mathematical models for a brain and neural networks were thus laid by Marvin L. Minsky in his PhD dissertation.

- Marvin L. Minsky, *Neural Nets and the Brain Model Problem*, Ph.D. Dissertation in the Department of Mathematics, Princeton, 1954. (Many new theories and theorems about learning in neural networks, secondary reinforcement, circulating dynamic storage and synaptic modifications.)

Unfortunately, these two early works by Marvin L. Minsky have never been published. The definitions of the learning algorithms for these two neural machines are unknown. Otherwise, it could have better linkage of the contributions of Marvin L. Minsky to the present learning theory.