AI Application Lecture 1

Real-World Problems and the Formulation of Machine Learning

SUZUKI, Atsushi

Outline

Introduction

1. Introduction: Purpose

The purpose of this course is to learn the fundamentals of AI in the application context, especially **Machine Learning** using **Neural Networks**, from scratch.

1.1 Learning Outcomes

At the end of this lecture, students should be able to:

- Formulate tasks that humanity seeks to solve as a relationship between **input** and **output**.

1.1 Learning Outcomes

At the end of this lecture, students should be able to:

- Formulate tasks that humanity seeks to solve as a relationship between **input** and **output**.
- Mutually convert various real-world **entities** that humanity deals with and their corresponding **numerical sequences**.

1.1 Learning Outcomes

At the end of this lecture, students should be able to:

- Formulate tasks that humanity seeks to solve as a relationship between **input** and **output**.
- Mutually convert various real-world **entities** that humanity deals with and their corresponding **numerical sequences**.
- Formulate **Artificial Intelligence** as a **function determination problem**.

1.2 Approach of This Course

This course will start with fundamental topics such as the definitions of machine learning and neural networks. Some students may have already studied these, but there is a reason for deliberately starting with basic definitions.

1.2 Approach of This Course

This course will start with fundamental topics such as the definitions of machine learning and neural networks. Some students may have already studied these, but there is a reason for deliberately starting with basic definitions.

The field of AI and machine learning is developing very rapidly, and specific technologies and models that are widely known today may well be obsolete by the time you graduate.

1.2 Approach of This Course

This course emphasizes the understanding of more general and universal concepts that will be applicable in the future, rather than learning individual technologies.

1.2 Approach of This Course

This course emphasizes the understanding of more general and universal concepts that will be applicable in the future, rather than learning individual technologies.

Also, many AI-related courses first teach the individual components that make up AI and then introduce application examples. However, this lecture adopts the reverse approach.

1. First, we will treat AI as a functional **black box** and start by learning "how to use it."
2. Then, as necessary, we will **white-box** the contents of the black box and uncover its mechanisms.

1.2 Approach of This Course

This approach has the following advantages.

- **Promoting multifaceted understanding:** Leads to a deeper understanding.

1.2 Approach of This Course

This approach has the following advantages.

- **Promoting multifaceted understanding:** Leads to a deeper understanding.
- **Acquiring practical knowledge:** You will at least acquire the practical knowledge of "how to use" AI.

1.2 Approach of This Course

This approach has the following advantages.

- **Promoting multifaceted understanding:** Leads to a deeper understanding.
- **Acquiring practical knowledge:** You will at least acquire the practical knowledge of "how to use" AI.
- **Gaining universal knowledge:** The "framework for using AI" is relatively resistant to change.

1.2 Approach of This Course

This approach has the following advantages.

- **Promoting multifaceted understanding:** Leads to a deeper understanding.
- **Acquiring practical knowledge:** You will at least acquire the practical knowledge of "how to use" AI.
- **Gaining universal knowledge:** The "framework for using AI" is relatively resistant to change.
- **Addressing realistic constraints:** Details of large-scale models are often trade secrets, and there is no "panacea learning method for all problems."

Preliminaries: Mathematical Notations

2. Preliminaries: Mathematical Notations

We will organize the basic mathematical notations used in this lecture.

- **Definition:**
  - $(\mathrm{LHS}) \coloneqq (\mathrm{RHS})$: Indicates that the left-hand side is defined by the right-hand side.
  - For example, $a \coloneqq b$ indicates that a is defined as b.

2. Preliminaries: Mathematical Notations

We will organize the basic mathematical notations used in this lecture.

- **Definition**:
  - $(\mathrm{LHS}) \coloneqq (\mathrm{RHS})$: Indicates that the left-hand side is defined by the right-hand side.
  - For example, $a \coloneqq b$ indicates that a is defined as b.
- **Set**:
  - Sets are often denoted by uppercase calligraphic letters. E.g., $\mathcal{A}$.
  - $x \in \mathcal{A}$: Indicates that the element x belongs to the set $\mathcal{A}$.
  - $\{\}$: The empty set.

- **Set Notation Examples:**
  - $\{a, b, c\}$: The set consisting of elements a, b, c (Roster notation).
  - $\{x \in \mathcal{A} | P(x)\}$: The set of elements in $\mathcal{A}$ for which the proposition $P(x)$ is true (set-builder notation).

- **Set Notation Examples:**
  - $\{a, b, c\}$: The set consisting of elements a, b, c (Roster notation).
  - $\{x \in \mathcal{A} | P(x)\}$: The set of elements in $\mathcal{A}$ for which the proposition $P(x)$ is true (set-builder notation).
- **Common Sets of Numbers:**
  - $\mathbb{R}$: The set of all real numbers.
  - $\mathbb{R}_{>0}$: The set of all positive real numbers.
  - $\mathbb{R}_{\geq 0}$: The set of all non-negative real numbers.
  - $\mathbb{Z}$: The set of all integers.
  - $\mathbb{Z}_{>0}$: The set of all positive integers.
  - $\mathbb{Z}_{\geq 0}$: The set of all non-negative integers.
  - $[1, k]_{\mathbb{Z}} \coloneqq \{1, 2, \ldots, k\}$: The set of integers from 1 to k.

- **Function:**
  - $f : \mathcal{X} \to \mathcal{Y}$: f is a **map** from a **domain** $\mathcal{X}$ to a **codomain** $\mathcal{Y}$.
  - $y = f(x)$: The output of f for an input $x \in \mathcal{X}$ is $y \in \mathcal{Y}$.

2. Preliminaries: Mathematical Notations

- **Function:**
  - $f : \mathcal{X} \to \mathcal{Y}$: f is a **map** from a **domain** $\mathcal{X}$ to a **codomain** $\mathcal{Y}$.
  - $y = f(x)$: The output of f for an input $x \in \mathcal{X}$ is $y \in \mathcal{Y}$.
- **Vector:**
  - In this course, a vector is a column of numbers.
  - Denoted by bold italic lowercase letters (e.g., $\boldsymbol{v}$).
  - $\boldsymbol{v} \in \mathbb{R}^n$: an n-dimensional real vector.
  - The i-th element is denoted as v_i.

$$\boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

- **Matrix:** Denoted by bold italic uppercase letters (e.g., $\boldsymbol{A}$).
  - $\boldsymbol{A} \in \mathbb{R}^{m,n}$: an $m \times n$ real matrix.
  - The element in the i-th row and j-th column is denoted as $a_{i,j}$.

$$\boldsymbol{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix}$$

- **Matrix Transpose:** The transpose of $\boldsymbol{A}$ is denoted as $\boldsymbol{A}^\top$.
  - If $\boldsymbol{A} \in \mathbb{R}^{m,n}$, then $\boldsymbol{A}^\top \in \mathbb{R}^{n,m}$.

$$\boldsymbol{A}^\top = \begin{bmatrix} a_{1,1} & a_{2,1} & \cdots & a_{m,1} \\ a_{1,2} & a_{2,2} & \cdots & a_{m,2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1,n} & a_{2,n} & \cdots & a_{m,n} \end{bmatrix}$$

- **Matrix Transpose:** The transpose of $\boldsymbol{A}$ is denoted as $\boldsymbol{A}^\top$.
  - If $\boldsymbol{A} \in \mathbb{R}^{m,n}$, then $\boldsymbol{A}^\top \in \mathbb{R}^{n,m}$.

$$\boldsymbol{A}^\top = \begin{bmatrix} a_{1,1} & a_{2,1} & \cdots & a_{m,1} \\ a_{1,2} & a_{2,2} & \cdots & a_{m,2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1,n} & a_{2,n} & \cdots & a_{m,n} \end{bmatrix}$$

- **Vector Transpose:** A vector is a matrix with 1 column.
  - The transpose of a column vector is a row vector.

$$\boldsymbol{v}^\top = \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix} \in \mathbb{R}^{1,n}$$

2. Preliminaries: Mathematical Notations

- **Tensor:**
  - In this lecture, we treat tensors as multidimensional arrays.
  - A vector is a 1st-order tensor; a matrix is a 2nd-order tensor.
  - Tensors of 3rd order or higher are denoted by an underlined, bold, italic, uppercase letter, e.g., $\underline{\boldsymbol{A}}$.
  - Note: For those familiar with abstract tensors, we assume a fixed standard basis, identifying a tensor with its component representation.

2. Preliminaries: Mathematical Notations

- **Tensor:**
  - In this lecture, we treat tensors as multidimensional arrays.
  - A vector is a 1st-order tensor; a matrix is a 2nd-order tensor.
  - Tensors of 3rd order or higher are denoted by an underlined, bold, italic, uppercase letter, e.g., $\underline{\boldsymbol{A}}$.
  - Note: For those familiar with abstract tensors, we assume a fixed standard basis, identifying a tensor with its component representation.
- **Note on Implementation:**
  - Theoretically, we use real numbers ($\mathbb{R}$).
  - In practice, computers use finite subsets (e.g., floating-point numbers).
  - These subsets are approximations. Their discreteness and finiteness can cause different properties from $\mathbb{R}$.

What is AI?

3. What is AI?

This course is about "AI applications," but what is "Artificial Intelligence" in the first place? Defining this term is extremely difficult.

3. What is AI?

This course is about "AI applications," but what is "Artificial Intelligence" in the first place? Defining this term is extremely difficult.

- **Artificial**: Realized using a **computer**.
- **Intelligence**: No unified definition exists, even among experts.

3. What is AI?

This course is about "AI applications," but what is "Artificial Intelligence" in the first place? Defining this term is extremely difficult.

- **Artificial**: Realized using a **computer**.
- **Intelligence**: No unified definition exists, even among experts.

From an engineering perspective, what is important is whether that technology can contribute to human society.

3. What is AI?

This course will not delve into any philosophical discussions, but will define AI in a very practical way as follows.

Definition (Definition of AI in This Lecture)

Artificial Intelligence (AI) is a procedure implemented on a computer that solves some task that humanity wants to solve.

Task Formulation: Problems as Input-Output Relationships

4. Task Formulation: Problems as Input-Output Relationships

Many of the tasks that humans want computers to solve can be formulated as a problem of "returning an appropriate output given a certain input."

4. Task Formulation: Problems as Input-Output Relationships

Many of the tasks that humans want computers to solve can be formulated as a problem of "returning an appropriate output given a certain input."

This is the idea of viewing a task as a kind of transformation rule, that is, a correspondence between input and output. Let’s look at some examples.

4. Task Formulation: Examples

- **Chat AI:**
  - **Input:** A question or command sentence. (e.g., "What's the weather like today?")
  - **Output:** The AI's response text. (e.g., "The weather in Tokyo is sunny.")

4. Task Formulation: Examples

- **Chat AI:**
  - **Input:** A question or command sentence. (e.g., "What's the weather like today?")
  - **Output:** The AI's response text. (e.g., "The weather in Tokyo is sunny.")
- **Autonomous Driving:**
  - **Input:** Sensor information (camera images, LiDAR, GPS, etc.).
  - **Output:** Vehicle control signals (steering, accelerator, brake).

4. Task Formulation: Examples

- **Chat AI:**
  - **Input:** A question or command sentence. (e.g., "What's the weather like today?")
  - **Output:** The AI's response text. (e.g., "The weather in Tokyo is sunny.")
- **Autonomous Driving:**
  - **Input:** Sensor information (camera images, LiDAR, GPS, etc.).
  - **Output:** Vehicle control signals (steering, accelerator, brake).
- **Text-to-Image (Image Generation AI):**
  - **Input:** A descriptive sentence (**prompt**). (e.g., "A cat flying in space, oil painting style")
  - **Output:** Generated image data.

4. Task Formulation: Examples

- **Medical Diagnosis from Images:**
  - **Input:** Medical image data (X-rays, CT scans, etc.).
  - **Output:** A diagnosis label (e.g., "positive" or "negative").

4. Task Formulation: Examples

- **Medical Diagnosis from Images:**
  - **Input:** Medical image data (X-rays, CT scans, etc.).
  - **Output:** A diagnosis label (e.g., "positive" or "negative").
- **Fraud Detection:**
  - **Input:** Credit card transaction information (time, location, amount, etc.).
  - **Output:** A judgment result (e.g., "likely fraudulent" or "normal").

4. Task Formulation: Examples

- **Medical Diagnosis from Images:**
  - **Input:** Medical image data (X-rays, CT scans, etc.).
  - **Output:** A diagnosis label (e.g., "positive" or "negative").
- **Fraud Detection:**
  - **Input:** Credit card transaction information (time, location, amount, etc.).
  - **Output:** A judgment result (e.g., "likely fraudulent" or "normal").

As can be seen from these examples, even tasks that seem completely different share a common structure of "input $\rightarrow$ output."

Numerical Representation of Real-World Entities

5. Numerical Representation of Real-World Entities

Fundamentally, what computers can handle are **bit strings** of 0s and 1s, or the **byte strings** and numbers they form.

5. Numerical Representation of Real-World Entities

Fundamentally, what computers can handle are **bit strings** of 0s and 1s, or the **byte strings** and numbers they form.

Therefore, to process the various inputs and outputs we saw in the previous section on a computer, they must first be converted into numerical data. This conversion rule is defined by humans according to the task.

5.1 Natural Language

A sentence is a sequence of characters. The rule that maps each character to a specific byte sequence is called a **character encoding**.

- **ASCII (American Standard Code for Information Interchange):**
  - Maps alphabets, numbers, symbols, etc., to 7-bit (usually 1-byte) integers.
  - For example, ‘A‘ is 65, and ‘B‘ is 66.

5.1 Natural Language: ASCII Table

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
32	20	(space)	56	38	8	80	50	P	104	68	h
33	21	!	57	39	9	81	51	Q	105	69	i
34	22	"	58	3A	:	82	52	R	106	6A	j
35	23	#	59	3B	;	83	53	S	107	6B	k
36	24	$	60	3C	<	84	54	T	108	6C	l
37	25	%	61	3D	=	85	55	U	109	6D	m
38	26	&	62	3E	>	86	56	V	110	6E	n
39	27	'	63	3F	?	87	57	W	111	6F	o
40	28	(	64	40	@	88	58	X	112	70	p
41	29	)	65	41	A	89	59	Y	113	71	q
42	2A	*	66	42	B	90	5A	Z	114	72	r
43	2B	+	67	43	C	91	5B	[	115	73	s
44	2C	,	68	44	D	92	5C	\	116	74	t
45	2D	-	69	45	E	93	5D	]	117	75	u
46	2E	.	70	46	F	94	5E	^	118	76	v
47	2F	/	71	47	G	95	5F	_	119	77	w
48	30	0	72	48	H	96	60	`	120	78	x
49	31	1	73	49	I	97	61	a	121	79	y
50	32	2	74	4A	J	98	62	b	122	7A	z
51	33	3	75	4B	K	99	63	c	123	7B	{
52	34	4	76	4C	L	100	64	d	124	7C	|
53	35	5	77	4D	M	101	65	e	125	7D	}
54	36	6	78	4E	N	102	66	f	126	7E	~
55	37	7	79	4F	O	103	67	g			

5.1 Natural Language: UTF-8

- **UTF-8 (Unicode Transformation Format-8):**
  - Includes characters from all over the world.
  - Uses a variable-length byte sequence (1 to 4 bytes).
  - e.g., ‘A’ is 1 byte, while ‘日’ and ‘𠮷’ are 3 and 4 bytes.

5.1 Natural Language: UTF-8

- **UTF-8 (Unicode Transformation Format-8):**
  - Includes characters from all over the world.
  - Uses a variable-length byte sequence (1 to 4 bytes).
  - e.g., 'A' is 1 byte, while '日' and '𝌆' are 3 and 4 bytes.

Example (UTF-8 representations of letters and characters)

- '日' (U+65E5) → UTF-8 (hex): `E6 97 A5` (3 bytes)
- Greek 'Ω' (U+03A9) → UTF-8 (hex): `CE A9` (2 bytes)
- Emoji '𝌆' (U+1F602) → UTF-8 (hex): `F0 9F 98 82` (4 bytes)

With this rule, any text can be converted into a unique byte sequence.

5.1 Natural Language: Remark on Emoji

Remark (On the Technical Significance of Emoji)

Emoji, originating from Japan, are now used worldwide. Their popularization has motivated worldwide support for multi-byte characters.

5.1 Natural Language: Remark on Emoji

Remark (On the Technical Significance of Emoji)

Emoji, originating from Japan, are now used worldwide. Their popularization has motivated worldwide support for multi-byte characters.

For instance, people in the Chinese character culture sphere (China, Japan, Korea, Vietnam) have benefited from this, as it improved support for characters like Chinese characters. Before emoji became popular, supporting these character sets was not a high priority for the predominantly English-speaking tech world.

5.1 Natural Language: Remark on Emoji

Remark (On the Technical Significance of Emoji)

The word Emoji comes from the Japanese word "絵文字" (e-moji), meaning "picture character," not from "emotion." The practice of assigning character codes to pictures dates back to the 1950s in Japan.

5.1 Natural Language: Remark on Emoji

Remark (On the Technical Significance of Emoji)

The word Emoji comes from the Japanese word "絵文字" (e-moji), meaning "picture character," not from "emotion." The practice of assigning character codes to pictures dates back to the 1950s in Japan.

Later, on Japanese mobile phones in the 2000s before the spread of smartphones, emoji were so abundant that one could communicate using them alone.

5.1 Natural Language: Remark on Emoji

Remark (On the Technical Significance of Emoji)

The word Emoji comes from the Japanese word "絵文字" (e-moji), meaning "picture character," not from "emotion." The practice of assigning character codes to pictures dates back to the 1950s in Japan.

Later, on Japanese mobile phones in the 2000s before the spread of smartphones, emoji were so abundant that one could communicate using them alone.

However, the contribution of the Gmail team in assigning Unicode points was also significant for the global spread of emoji.

5.2 Images

A color image is a collection of **pixels**. The color of each pixel can be represented by a combination of the intensities of Red, Green, and Blue (RGB).

5.2 Images

A color image is a collection of **pixels**. The color of each pixel can be represented by a combination of the intensities of Red, Green, and Blue (RGB).

- **Bitmap representation:**
  - An image of height H and width W can be represented as a 3rd-order integer tensor $\underline{\boldsymbol{I}} \in \mathbb{Z}^{3 \times H \times W}$.
  - Each of the three $H \times W$ matrices corresponds to the R, G, and B channels, with integer values from 0 to 255.

5.2 Images: Example

Consider a small 2×2 pixel image:

- Top-left: Red (R:255, G:0, B:0)
- Top-right: Green (R:0, G:255, B:0)
- Bottom-left: Blue (R:0, G:0, B:255)
- Bottom-right: White (R:255, G:255, B:255)

5.2 Images: Example

Consider a small 2×2 pixel image:

- Top-left: Red (R:255, G:0, B:0)
- Top-right: Green (R:0, G:255, B:0)
- Bottom-left: Blue (R:0, G:0, B:255)
- Bottom-right: White (R:255, G:255, B:255)

This image is represented as a $3 \times 2 \times 2$ tensor $\underline{\boldsymbol{I}}$, where the channels $\underline{\boldsymbol{I}}_R, \underline{\boldsymbol{I}}_G, \underline{\boldsymbol{I}}_B$ are:

$$\underline{\boldsymbol{I}}_R = \begin{bmatrix} 255 & 0 \\ 0 & 255 \end{bmatrix}, \quad \underline{\boldsymbol{I}}_G = \begin{bmatrix} 0 & 255 \\ 0 & 255 \end{bmatrix}, \quad \underline{\boldsymbol{I}}_B = \begin{bmatrix} 0 & 0 \\ 255 & 255 \end{bmatrix}$$

5.3 Video & 5.4 Audio

- **Video**: A sequence of time-varying images.
  - Add a time dimension to the image tensor.
  - A video of T frames is a 4th-order tensor $\underline{\boldsymbol{V}} \in \mathbb{Z}^{3 \times T \times H \times W}$.

5.3 Video & 5.4 Audio

- **Video**: A sequence of time-varying images.
  - Add a time dimension to the image tensor.
  - A video of T frames is a 4th-order tensor $\underline{\boldsymbol{V}} \in \mathbb{Z}^{3\times T\times H\times W}$.
- **Audio**: A time-varying wave.
  - **PCM (Pulse Code Modulation):** The analog signal is converted to a sequence of integers through:
    - **Sampling**: Measuring amplitude at regular time intervals.
    - **Quantization**: Approximating each measurement with an integer value.

5.4 Audio: PCM

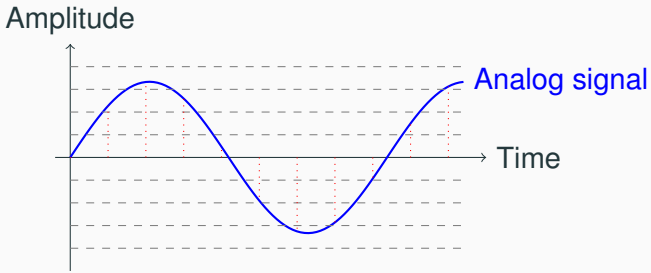

Figure 1: Schematic of sampling and quantization of an audio signal

5.4 Audio: PCM

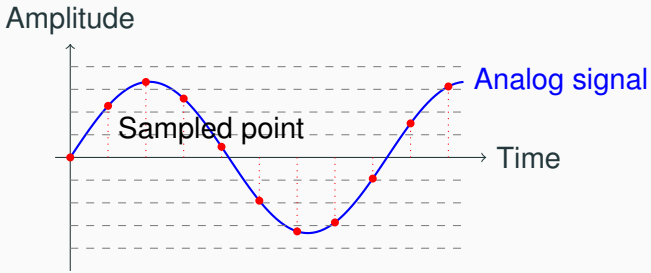

Figure 1: Schematic of sampling and quantization of an audio signal

5.4 Audio: PCM

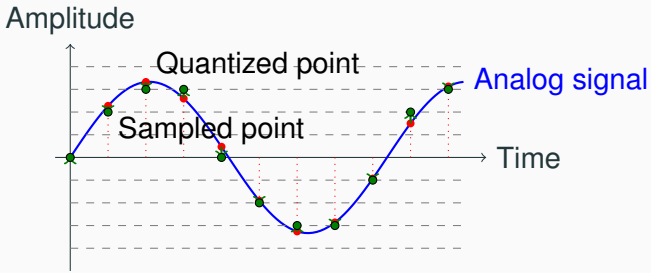

Figure 1: Schematic of sampling and quantization of an audio signal

5. Numerical Representation: Summary

In the case of stereo audio, this process is performed for the left and right channels respectively, resulting in two integer sequences $\boldsymbol{l}, \boldsymbol{r} \in \mathbb{Z}^D$.

5. Numerical Representation: Summary

In the case of stereo audio, this process is performed for the left and right channels respectively, resulting in two integer sequences $\boldsymbol{l}, \boldsymbol{r} \in \mathbb{Z}^D$.

In this way, various real-world entities can be converted into numerical data (vectors or tensors) based on clear rules.

Redefining AI: The Function Determination Problem

6. Redefining AI: The Function Determination Problem

Now, let's summarize the discussion so far.

1. The task we want to solve can be seen as a correspondence of "input entity $\rightarrow$ output entity."

6. Redefining AI: The Function Determination Problem

Now, let's summarize the discussion so far.

1. The task we want to solve can be seen as a correspondence of "input entity $\rightarrow$ output entity."
2. Input and output entities can be converted into "input numerical data $\rightarrow$ output numerical data" according to appropriate rules.

6. Redefining AI: The Function Determination Problem

Now, let's summarize the discussion so far.

1. The task we want to solve can be seen as a correspondence of "input entity $\rightarrow$ output entity."
2. Input and output entities can be converted into "input numerical data $\rightarrow$ output numerical data" according to appropriate rules.

Through these two steps, the problems we want AI to solve ultimately come down to the problem of "calculating the appropriate output numerical data given some input numerical data."

This is none other than the **problem of finding a function** from the set of all input numerical data (**input space**) to the set of all output numerical data (**output space**).

Definition (Engineering Goal of AI)

To solve a task is to construct a function f on a computer that mimics an unknown ideal function $f^* : \mathcal{X} \to \mathcal{Y}$ that represents the input-output relationship of that task.

6.1 Input and Output Spaces in Each Application Example

- **Chat AI:**
  - Input space $\mathcal{X}$: Set of UTF-8 byte strings.
  - Output space $\mathcal{Y}$: Set of UTF-8 byte strings.

6.1 Input and Output Spaces in Each Application Example

- **Chat AI:**
  - Input space $\mathcal{X}$: Set of UTF-8 byte strings.
  - Output space $\mathcal{Y}$: Set of UTF-8 byte strings.
- **Autonomous Driving:**
  - Input space $\mathcal{X} \subset \mathbb{R}^{D_{in}}$: Vectors of sensor readings.
  - Output space $\mathcal{Y} \subset \mathbb{R}^{D_{out}}$: Vectors of control signals.

6.1 Input and Output Spaces in Each Application Example

- **Chat AI:**
  - Input space $\mathcal{X}$: Set of UTF-8 byte strings.
  - Output space $\mathcal{Y}$: Set of UTF-8 byte strings.
- **Autonomous Driving:**
  - Input space $\mathcal{X} \subset \mathbb{R}^{D_{in}}$: Vectors of sensor readings.
  - Output space $\mathcal{Y} \subset \mathbb{R}^{D_{out}}$: Vectors of control signals.
- **Text-to-Image:**
  - Input space $\mathcal{X}$: Set of byte strings (prompts).
  - Output space $\mathcal{Y} = \mathbb{Z}^{3\times H\times W}$: Set of image tensors.

6.1 Input and Output Spaces in Each Application Example

- **Medical Diagnosis from Images:**
  - Input space $\mathcal{X} \subset \mathbb{Z}^{C \times H \times W}$: Set of medical image tensors.
  - Output space $\mathcal{Y} = \{0, 1\}$ (0: negative, 1: positive).

6.1 Input and Output Spaces in Each Application Example

- **Medical Diagnosis from Images:**
  - Input space $\mathcal{X} \subset \mathbb{Z}^{C \times H \times W}$: Set of medical image tensors.
  - Output space $\mathcal{Y} = \{0, 1\}$ (0: negative, 1: positive).
- **Fraud Detection:**
  - Input space $\mathcal{X} \subset \mathbb{R}^D$: Real vectors of D transaction features.
  - Output space $\mathcal{Y} = \{0, 1\}$ (0: normal, 1: fraudulent).

6.1 Input and Output Spaces in Each Application Example

- **Medical Diagnosis from Images:**
  - Input space $\mathcal{X} \subset \mathbb{Z}^{C \times H \times W}$: Set of medical image tensors.
  - Output space $\mathcal{Y} = \{0, 1\}$ (0: negative, 1: positive).
- **Fraud Detection:**
  - Input space $\mathcal{X} \subset \mathbb{R}^D$: Real vectors of D transaction features.
  - Output space $\mathcal{Y} = \{0, 1\}$ (0: normal, 1: fraudulent).

Thus, we can see that any task can be uniformly described as a problem of determining a function f from an input space $\mathcal{X}$ to an output space $\mathcal{Y}$.

Summary and Future Outlook

7.1 Today's Summary

In this lecture, we have organized the diverse tasks that AI tackles from a mathematical perspective.

- **Formulating tasks as input-output:** Many tasks can be formulated as determining an output for a given input.

7.1 Today's Summary

In this lecture, we have organized the diverse tasks that AI tackles from a mathematical perspective.

- **Formulating tasks as input-output:** Many tasks can be formulated as determining an output for a given input.
- **Numerical representation of entities:** Real-world entities are converted into computer-manageable numerical data (vectors and tensors).

7.1 Today's Summary

In this lecture, we have organized the diverse tasks that AI tackles from a mathematical perspective.

- **Formulating tasks as input-output:** Many tasks can be formulated as determining an output for a given input.
- **Numerical representation of entities:** Real-world entities are converted into computer-manageable numerical data (vectors and tensors).
- **AI and the function determination problem:** Solving a task with AI is equivalent to determining an appropriate function $f : \mathcal{X} \to \mathcal{Y}$.

7.2 Next Time

We have seen that solving a task can be equated with "determining a function."
So, how can we realize this function f on a computer?

7.2 Next Time

We have seen that solving a task can be equated with "determining a function." So, how can we realize this function f on a computer?

In particular, when the ideal function f^* is unknown, how can we make f "learn" to be close to it?

7.2 Next Time

We have seen that solving a task can be equated with "determining a function." So, how can we realize this function f on a computer?

In particular, when the ideal function f^* is unknown, how can we make f "learn" to be close to it?

In the upcoming lectures, we will learn about neural networks, which are the concrete components of this function f, and uncover how they learn complex input-output relationships.