인공지능의 이해

Understanding Artificial Intelligence

유준수 (Joon Soo Yoo, Ph.D.)

Fall 2026

Defense & Advanced Security Lab (DAS Lab)
Department of Advanced Defense Technology and Industry
Jeonbuk National University

Introduction to Generative AI

From Prediction to Generation

Week 09

1. From Week 07 to Week 09

중간고사 이전에는 Data로부터 Pattern을 학습하고, Prediction하는 과정을 배웠다.

$$\text{Data} \rightarrow \text{Training} \rightarrow \text{Model} \rightarrow \text{Prediction}$$

- Regression: 연속적인 숫자를 예측한다.
- Classification: Class 또는 Category를 예측한다.
- CNN: Image의 Local Pattern을 학습하여 분류한다.

오늘의 질문

AI가 정답을 분류하는 것을 넘어, 새로운 문장이나 이미지를 만드는 것은 어떻게 가능할까?

1-1. Today's Mission

1. Generative AI의 의미와 활용 사례 이해
2. Classification과 Generation의 차이 이해
3. Token과 Tokenization의 역할 이해
4. Next Token Prediction의 원리 이해
5. Temperature와 Sampling의 역할 이해
6. Transformer와 LLM의 전체 구조 살펴보기

학습 목표

지금까지 배운 Neural Network, Probability, Prediction 개념을 Text Generation과 연결하여 설명할 수 있다.

2. What Is Generative AI?

Generative AI는 학습한 Data의 Pattern을 바탕으로 새로운 Content를 생성하는 AI이다.

Input	Generated Output
Text	Text
Text	Image
Text	Code
Audio	Text
Image + Text	Text

핵심

Generative AI는 주어진 Input을 분석하는 것뿐 아니라, 새로운 Text, Image, Code 등의 Content를 생성한다.

2-1. AI Around Us

우리는 이미 다양한 Generative AI를 사용하고 있다.

- ChatGPT: 질문에 대한 답변 생성
- Coding Assistant: 설명을 바탕으로 Code 생성
- Image Generator: Prompt를 바탕으로 Image 생성
- Summarizer: 긴 Document를 짧은 문장으로 요약

생각해보기

ChatGPT에게 다음 두 가지를 요청한다고 하자.

1. 이 문장이 긍정적인지 부정적인지 판단해줘.
2. 긍정적인 문장을 새롭게 작성해줘.

두 요청은 어떤 점에서 다른가?

3. Classification vs. Generation

지금까지 배운 Classification을 생각해보자.

$$\text{Image} \rightarrow \text{CNN} \rightarrow \text{Cat}$$

Generative AI에서는 새로운 Content를 만든다.

$$\text{Text Prompt} \rightarrow \text{Generative Model} \rightarrow \text{New Text}$$

핵심

Classification은 주어진 Class 중 하나를 선택한다.
Generation은 새로운 Content를 생성한다.

3-1. Is Generation Different from Prediction?

Generative AI도 Prediction을 활용할 수 있다.

예를 들어 다음 문장을 생각해보자.

The cat is sitting on the _____

다음에 올 단어는 무엇일까?

- mat
- floor
- chair
- moon

핵심

Autoregressive Language Model은 다음 Token을 Prediction하는 과정을 반복하여 새로운 문장을 생성한다.

3-2. Connection to Neural Networks

Week 06에서는 Neural Network가 Input으로부터 Prediction을 만드는 과정을 배웠다.

$$\text{Input} \rightarrow \text{Neural Network} \rightarrow \text{Prediction}$$

Language Model에서도 유사한 구조를 생각할 수 있다.

$$\text{Text Context} \rightarrow \text{Language Model} \rightarrow \text{Next Token Scores}$$

핵심

Generative AI에서도 Weight, Training, Neural Network, Probability 등의 개념이 사용된다.

4. What Is a Language Model?

Language Model은 Text에 나타나는 Pattern을 학습하여 Token Sequence의 확률적 관계를 모델링한다.

예를 들어 다음 문장을 생각해보자.

I am studying artificial _____

- 앞부분의 Text가 Context가 된다.
- Model은 다음 Token에 대한 Score를 계산한다.
- Score를 Probability로 변환할 수 있다.
- 선택된 Token을 이용하여 문장을 이어갈 수 있다.

핵심

Language Model은 Context를 이용하여 다음 Token의 Probability를 계산할 수 있다.

5. What Is a Token?

Computer가 Text를 처리하려면 Text를 계산 가능한 형태로 변환해야 한다.

$$\text{Text} \rightarrow \text{Tokens} \rightarrow \text{Token IDs}$$

예를 들어 다음 문장을 생각해보자.

I love artificial intelligence.

개념적으로 다음과 같이 나눌 수 있다.

[I][love][artificial][intelligence][.]

핵심

Token은 Language Model이 Text를 처리하기 위해 사용하는 기본 단위이다.

5-1. Token Is Not Always a Word

Token은 반드시 하나의 단어와 일치하지 않는다.

Tokenizer에 따라 다음과 같은 단위를 사용할 수 있다.

- Word
- Subword
- Character
- Byte 기반 단위

따라서 같은 문장이라도 Tokenizer에 따라 Token의 개수가 달라질 수 있다.

주의

앞 슬라이드의 Token 분할은 개념 설명을 위한 예시이다.
실제 Tokenizer에서는 서로 다른 결과가 나타날 수 있다.

5-2. Interactive Demo: Tokenization

실제 Tokenizer를 사용하여 문장을 나누어보자.

Interactive Visualization

OpenAI Tokenizer – Click Here

다음 문장을 각각 입력해보자.

1. I love artificial intelligence.
2. Artificial intelligence is amazing.
3. 저는 인공지능을 공부하고 있습니다.

질문

영어와 한국어 문장의 Token 수는 어떻게 다른가?
띄어쓰기나 문장부호를 바꾸면 결과도 달라지는가?

5-3. From Tokens to Numbers

Tokenizer는 각 Token을 정수 ID로 나타낼 수 있다.

$$\text{Token} \rightarrow \text{Token ID}$$

예를 들어 다음과 같은 가상의 Mapping을 생각해보자.

Token	Token ID
I	15
love	82
AI	104

핵심

Token ID는 Token을 구분하기 위한 숫자이다.
ID의 크기 자체가 단어의 의미나 중요도를 나타내지는 않는다.

5-4. What Is an Embedding?

Token ID만으로는 단어의 의미적 관계를 직접 표현하기 어렵다.

Embedding은 Token을 Vector로 변환한다.

$$\text{Token ID} \rightarrow \text{Embedding Vector}$$

$$i \rightarrow \mathbf{e}_i \in \mathbb{R}^d$$

- 각 Token은 숫자로 이루어진 Vector로 표현된다.
- Embedding은 Training 과정에서 학습될 수 있다.
- Model은 이러한 Vector를 이용하여 계산한다.

핵심

Embedding은 Token을 Neural Network가 처리할 수 있는 숫자 표현으로 변환한다.

5-5. From Token IDs to Embeddings

Token을 숫자로 표현하는 방법을 비교해보자.

예를 들어 Vocabulary가 4개라고 하자.

Token	ID	Binary	One-hot
I	2	010	[0,0,1,0]
love	1	001	[0,1,0,0]
AI	3	011	[0,0,0,1]
you	0	000	[1,0,0,0]

실제 Neural Network에서는 학습 가능한 Embedding Vector를 사용한다.

$$\text{Token ID } 2 \longrightarrow \mathbf{e}_2 = [0.2, -0.5, 0.8]$$

핵심

Binary와 One-hot은 Token을 구분하는 표현이다.
Learned Embedding은 Training을 통해 학습되는 Vector이며, Token 간 관계를 표현할 수 있다.

6. Next Token Prediction

다음 문장을 생각해보자.

The cat is sitting on the ...

Model이 다음과 같은 Probability를 계산했다고 가정하자.

Next Token	Probability
mat	0.50
floor	0.30
chair	0.15
moon	0.05

핵심

Language Model은 가능한 다음 Token에 대한 Probability Distribution을 계산할 수 있다.

6-1. Connection to Softmax

Week 06에서는 Neural Network의 Output에 Softmax를 적용하여 Class Probability를 구했다.

Language Model에서도 유사한 방법을 사용한다.

$$\text{Context} \rightarrow \text{Neural Network} \rightarrow \text{Logits} \rightarrow \text{Softmax}$$

$$P(t_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

- z_i: Token i의 Logit
- $P(t_i)$: Token i가 선택될 확률
- 모든 Token Probability의 합은 1

핵심

MNIST에서는 10개 Class를 다뤘지만, Language Model에서는 많은 Token 후보를 다룬다.

6-2. Softmax Calculation Example

Language Model이 다음 Logit을 출력했다고 하자.

$$\text{Context} \rightarrow \text{Neural Network} \rightarrow z = [2, 1, 0]$$

Softmax는 Logit을 Probability로 변환한다.

$$P_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Token	Logit	$\exp(z_i)$	Probability
A	2	7.389	0.665
B	1	2.718	0.245
C	0	1.000	0.090
Total	–	11.107	1.000

예를 들어 Token A의 Probability는

$$P(A) = \frac{e^2}{e^2 + e^1 + e^0} = \frac{7.389}{11.107} \approx 0.665$$

핵심

Logit은 Probability가 아니다.
Softmax를 적용하면 각 Token의 Probability를 계산할 수 있으며, 전체 합은 1이 된다.

6-3. Mini Problem: Probability

어떤 Language Model이 다음 확률을 출력했다고 하자.

Token	Probability
A	0.60
B	0.25
C	0.10
D	0.05

Questions

1. 가장 높은 Probability를 가진 Token은?
2. 항상 A를 선택하면 결과가 달라질 수 있을까?
3. Probability에 따라 선택하면 어떤 일이 생길까?

7. From Prediction to Generation

Language Model은 다음 Token을 한 번 예측하는 것으로 끝나지 않는다.

Context → Predict Next Token

↓

Select Token → Append to Context

↓

Predict Again

핵심

다음 Token을 예측하고 Context에 추가하는 과정을 반복하면 새로운 문장을 생성할 수 있다.

7-1. Autoregressive Generation

다음은 Text Generation의 단순화된 예시이다.

1. Prompt: *The cat*
2. Next Token: *is*
3. Updated Context: *The cat is*
4. Next Token: *sleeping*
5. Updated Context: *The cat is sleeping*
6. Next Token: *.*

핵심

이전 Token을 이용하여 다음 Token을 생성하는 방식을 Autoregressive Generation이라고 한다.

7-2. When Does Generation Stop?

Text Generation이 무한히 계속되는 것은 아니다.

대표적인 종료 조건은 다음과 같다.

- End-of-Sequence Token이 생성되는 경우
- Maximum New Tokens에 도달한 경우
- Application에서 정의한 종료 조건을 만족한 경우

생각해보기

같은 Prompt에서 Maximum New Tokens를 10과 100으로 설정하면 어떤 차이가 발생할까?

8. How Do We Select the Next Token?

Model이 Probability Distribution을 계산했다고 하자.

다음 Token을 선택하는 방법은 여러 가지가 있다.

- Greedy Decoding: 가장 높은 Probability의 Token을 선택한다.
- Sampling: Probability Distribution을 이용하여 Token을 추출한다.

핵심

같은 Model과 Context에서도 Token Selection 방법에 따라 서로 다른 문장이 생성될 수 있다.

8-1. What Is Temperature?

Temperature는 Sampling에 사용하는 Probability Distribution을 조절한다.

$$P_i(T) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$$

- z_i: Token i의 Logit
- T: Temperature ($T > 0$)
- $P_i(T)$: 조정된 Token Probability

핵심

Temperature는 Model의 Weight를 변경하지 않는다.
이미 계산된 Logit으로부터 Sampling Probability를 조절한다.

8-2. Low vs. High Temperature

Temperature에 따라 Probability Distribution의 형태가 달라진다.

Temperature	Distribution	Sampling
Low	More concentrated	More predictable
Medium	Moderate	Balanced
High	More uniform	More diverse

- Low Temperature: 높은 Probability의 Token에 더 집중한다.
- High Temperature: 낮은 Probability의 Token도 선택될 가능성이 커진다.

주의

Temperature가 높다고 항상 더 좋은 문장이 생성되는 것은 아니다.
너무 높으면 문장의 일관성이 떨어질 수 있다.

8-3. Example: Temperature and Probability

다음과 같은 Logit이 있다고 가정하자.

$$z = [2,\ 1,\ 0]$$

Temperature를 변경하면 다음과 같이 Probability가 달라진다.

Temperature	Token A	Token B	Token C
$T = 0.5$	0.867	0.117	0.016
$T = 1.0$	0.665	0.245	0.090
$T = 2.0$	0.506	0.307	0.186

생각해보기

1. 어느 Temperature에서 Token A가 가장 자주 선택될까?
2. 어느 Temperature에서 Token C가 선택될 가능성이 높을까?
3. Temperature를 바꾸면 가장 높은 확률의 Token도 바뀔까?

8-4. Mini Problem: Sampling

어떤 Language Model이 다음 Probability를 출력했다.

Token	Probability
cat	0.60
dog	0.25
bird	0.10
car	0.05

Questions

1. Greedy Decoding은 어떤 Token을 선택할까?
2. Sampling에서는 dog가 선택될 수 있을까?
3. 같은 Prompt로 여러 번 생성하면 결과가 달라질 수 있을까?

핵심

Greedy Decoding은 가장 높은 확률의 Token을 선택한다.
Sampling은 Probability에 따라 다양한 Token을 선택할 수 있다.

9. What Is a Large Language Model?

Large Language Model (LLM)은 대규모 Text Data를 이용하여 학습한 Language Model이다.

$$\text{Large Text Dataset} \rightarrow \text{Training} \rightarrow \text{LLM}$$

LLM은 다양한 Text Processing Task에 활용된다.

- Question Answering
- Text Summarization
- Translation
- Code Generation
- Conversation

핵심

LLM은 학습한 Language Pattern을 바탕으로 주어진 Context에 적절한 Text를 생성할 수 있다.

9-1. How Does an LLM Learn?

Autoregressive Language Model의 대표적인 Training Objective는 Next Token Prediction이다.

예를 들어 다음 문장을 생각해보자.

I love artificial intelligence

Context	Target Token
I	love
I love	artificial
I love artificial	intelligence

핵심

Training에서는 실제 다음 Token을 정답으로 사용하여 Prediction Error를 줄이도록 Model을 학습한다.

9-2. Training vs. Text Generation

LLM의 Training과 Generation은 서로 다른 과정이다.

Stage	Purpose
Training	Data를 이용하여 Model Parameter를 학습한다.
Generation	학습된 Model로부터 새로운 Token을 생성한다.

$$\text{Training Data} \rightarrow \text{Learned Weights}$$

$$\text{Prompt} \rightarrow \text{Trained LLM} \rightarrow \text{Generated Text}$$

핵심

일반적인 Text Generation 과정에서는 Model의 Weight를 다시 학습하지 않는다.

10. Why Do We Need a Transformer?

Language Model은 다음 Token을 예측할 때 앞에 등장한 Text의 정보를 활용해야 한다.

다음 문장을 생각해보자.

Example

The animal did not cross the road because **it** was tired.

여기서 it은 무엇을 가리킬까?

- 문장 안의 Token들은 서로 관계를 가진다.
- 멀리 떨어진 Token도 중요한 정보를 제공할 수 있다.
- Model은 Context의 관계를 처리해야 한다.

핵심

Transformer는 Attention Mechanism을 활용하여 Token 사이의 관계를 처리하는 Neural Network 구조이다.

10-1. What Is Attention?

Attention은 다른 Token의 정보를 얼마나 반영할지 결정하는 Mechanism이다.

예를 들어 다음 문장을 생각해보자.

The cat ate the fish because it was hungry.

생각해보기

1. 여기서 it은 무엇을 가리키는가?
2. it의 의미를 이해하려면 어떤 단어를 참고해야 할까?
3. 모든 단어가 똑같이 중요할까?

핵심

Attention은 Context 안의 Token들이 서로의 정보를 활용하도록 돕는다.

10-2. Transformer Architecture

Transformer는 Attention을 중심으로 Token Representation을 처리한다.

Input Tokens

↓

Token Embeddings + Position Information

↓

Transformer Blocks

↓

Output Representations

핵심

Transformer Block에는 Attention과 Feed-Forward Network 등의 구성 요소가 포함된다.
GPT 계열의 Autoregressive LLM은 주로 Decoder-only Transformer를 사용한다.

10-3. Interactive Transformer Explainer

Transformer의 내부 동작을 Interactive Visualization으로 살펴보자.

Interactive Visualization

Transformer Explainer – Click Here

다음 요소들을 확인해보자.

1. Input Tokens
2. Token Embeddings
3. Attention
4. Transformer Blocks
5. Next Token Probability

질문

Transformer는 이전에 배운 MLP와 어떤 점에서 비슷하고 어떤 점에서 다를까?

10-4. Visualizing an LLM

LLM의 내부 계산 구조를 3D Visualization으로 살펴보자.

Interactive Visualization

LLM Visualization – Click Here

다음 내용을 관찰해보자.

1. Token이 Vector로 변환되는 과정
2. Transformer Layer를 통과하는 과정
3. Attention을 이용한 정보 처리
4. Output Token Probability 계산

주의

이번 Week에서는 전체적인 구조를 이해하는 것이 목표이다.
Attention의 세부 수학적 계산은 다음 Week에서 살펴본다.

11. Does an LLM Always Tell the Truth?

LLM이 자연스러운 문장을 생성한다고 해서 항상 사실에 맞는 답변을 만드는 것은 아니다.

예를 들어 다음과 같은 문제가 발생할 수 있다.

- 존재하지 않는 논문이나 출처를 제시한다.
- 잘못된 숫자나 날짜를 생성한다.
- 질문의 Context를 잘못 해석한다.
- 그럴듯하지만 사실과 다른 설명을 만든다.

핵심

LLM의 자연스러운 Text Generation 능력과 답변의 Factual Correctness는 서로 다르다.

11-1. What Is Hallucination?

Hallucination은 Generative AI가 사실과 다르거나 근거가 없는 내용을 그럴듯하게 생성하는 현상을 의미한다.

Example

사용자가 존재하지 않는 논문을 질문했을 때, Model이 실제로 존재하는 것처럼 제목, 저자, 출판연도를 만들어낼 수 있다.

- 높은 Confidence처럼 보이는 문장도 틀릴 수 있다.
- 자연스러운 문장이라고 사실인 것은 아니다.
- 중요한 정보는 외부 자료로 검증해야 한다.

핵심

Language Model의 Output은 검증 없이 항상 신뢰할 수 있는 정답이 아니다.

11-2. Why Does Hallucination Matter?

AI가 사용되는 분야에 따라 잘못된 정보의 영향은 달라진다.

Application	Possible Risk
Education	Incorrect explanation
Healthcare	Incorrect medical information
Finance	Incorrect financial analysis
Cybersecurity	Incorrect threat assessment
Robotics	Unsafe decision

생각해보기

ChatGPT가 작성한 문장을 그대로 제출하는 것과 내용을 검증한 후 활용하는 것은 어떤 차이가 있을까?

12. From Classification to Generation

지금까지 배운 AI의 흐름을 연결해보자.

Topic	Input	Output
Regression	Features	Number
Classification	Features	Class
CNN	Image	Class
Language Model	Text Context	Next Token
Text Generation	Prompt	New Text

핵심

Generative AI는 기존 Machine Learning과 완전히 분리된 기술이 아니다.
Neural Network, Training, Prediction, Probability 등의 개념을 바탕으로 발전하였다.

12-1. What We Learned

이번 Week에서는 Generative AI의 기본적인 동작 원리를 살펴보았다.

- Generative AI는 새로운 Content를 생성한다.
- Tokenizer는 Text를 Token으로 변환한다.
- Embedding은 Token을 Vector로 표현한다.
- Language Model은 Next Token Probability를 계산한다.
- Autoregressive Generation은 Token Prediction을 반복한다.
- Temperature는 Sampling Distribution을 조절한다.
- Transformer는 Attention을 활용하여 Context를 처리한다.
- LLM의 Output은 항상 사실이라고 보장할 수 없다.

핵심

Text Generation은 Next Token Prediction과 Token Selection의 반복으로 이해할 수 있다.

14. Next Week: Transformer and Attention

이번 Week에서는 Language Model이 Context를 이용하여 Next Token을 예측한다는 것을 배웠다.

하지만 아직 중요한 질문이 남아 있다.

Next Question

문장에 수많은 Token이 있을 때, Model은 어떤 Token의 정보를 중요하게 활용해야 하는지 어떻게 결정할까?

다음 Week에서는 다음 내용을 살펴본다.

- Transformer Architecture
- Self-Attention
- Query, Key, Value
- Positional Information
- Transformer 기반 Language Model

Thank You

Questions?