CSCI E-89B 강의 노트 전집

Neural Networks, RNN/LSTM/GRU, NLP 기초

정리: InJoo 학습노트

Fall 2025

Contents

Part I

Lecture 1: 아무것도 몰라도 이해되는 신경망 기초

읽기 가이드

이 문서는 **전혀 모르는 사람**을 위해 작성되었다. 수학식은 꼭 필요한 만큼만, 대신 **비유 + 손계산 예시 + 체크리스트**로 이해를 도와준다.

- **핵심 비유:** 신경망은 "입력 재료 → 여러 단계의 조리(층) → 결과 요리"를 만드는 **레시피**이다.
- **핵심 목표:** 예측이 실제와 얼마나 다른지(오차)를 숫자로 재고(손실 함수), 그걸 줄이는 방향으로 레시피(가중치)를 조금씩 조정(경사 하강법)한다.
- **읽는 순서:** 1) 신경망이 뭔지 직관, 2) 층/뉴런/활성화함수, 3) 손실/비용 함수, 4) 순전파/역전파 계산, 5) 경사하강법(배치/미니배치/SGD), 6) 실전 팁 & FAQ.

1 신경망을 아주 직관적으로 이해하기

한 줄 요약

신경망(Neural Network)은 **입력** x를 받아 **여러 층(layer)**을 거치며 **출력** $\hat{y}$를 만드는 **함수들의 합성**이다:

$$\hat{y} = f^{(L)}\big(f^{(L-1)}(\cdots f^{(1)}(x)\cdots)\big).$$

여기서 각 $f^{(\ell)}$은 **선형변환(가중치/편향)**과 **비선형 활성화 함수**로 이루어진다.

왜 굳이 "깊게(Deep)" 써야 할까?

- 얕은(한두 단계) 모델은 **직선/완만한 곡선** 같은 단순한 경계만 만든다.
- 복잡한 문제(예: 이미지/언어)는 **비선형 변환을 여러 번** 적용해 **복잡한 모양의 결정 경계**가 필요하다.
- 층을 쌓으면, **간단한 조각 특징** → **중간 특징** → **고수준 의미** 식으로 표현이 점점 "추상화"된다.

2 뉴런, 층, 활성화 함수

2.1 두 입력, 은닉 2, 출력 1인 2-2-1 미니 네트워크

구성: 입력 $x = (x_1, x_2)$, 은닉층 뉴런 u_1, u_2, 출력 $\hat{y}$. 은닉층은 **ReLU**를, 출력층은 회귀면 **선형**, 이진분류면 **시그모이드**, 다중분류면 **소프트맥스**를 쓴다고 생각하면 된다.

$$u_1 = f\Big(w_{01}^{(1)} + w_{11}^{(1)}x_1 + w_{21}^{(1)}x_2\Big),$$
$$u_2 = f\Big(w_{02}^{(1)} + w_{12}^{(1)}x_1 + w_{22}^{(1)}x_2\Big),$$
$$\hat{y} = g\Big(w_0^{(2)} + w_1^{(2)}u_1 + w_2^{(2)}u_2\Big).$$

여기서 f는 은닉층 활성화, g는 출력층 활성화다.

2.2 활성화 함수(왜 비선형이 필요?)

- **ReLU** ($f(x) = \max(0, x)$): 빠르고 간단, 깊은 네트워크에서도 학습 잘 됨. 기본값으로 생각해도 좋다.
- **Sigmoid** ($\sigma(x) = 1/(1 + e^{-x})$): 출력이 (0,1)이라 **확률**처럼 해석 쉬움(이진분류 출력층에 주로 사용).
- **Softmax:** $\mathrm{softmax}_j(z) = \dfrac{e^{z_j}}{\sum_k e^{z_k}}$ (다중분류 출력층).
- **Tanh:** $(-1, 1)$ 범위. 옛날엔 자주 썼지만 ReLU에 밀림.

핵심: 활성화가 **비선형**이어야 층을 쌓을 의미가 생긴다. 선형만 쌓으면 전체가 결국 또 선형이 된다.

3 손실(LOSS)과 비용(COST) 정확히 구분하기

3.1 손실 함수 $L^{(i)}$ (한 샘플의 틀림 정도)

- **회귀(실수 예측):** $L^{(i)} = \big(\hat{y}^{(i)} - y^{(i)}\big)^2$ (MSE 단일항)
- **이진분류:** $L^{(i)} = -\big(y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})\big)$
- **다중분류:** $L^{(i)} = -\sum_{c=1}^{M} y_c^{(i)} \log \hat{y}_c^{(i)}$ (원-핫 y 가정)

3.2 비용 함수 $J(\mathbf{w})$ (데이터 전체 평균 오차)

$$J(\mathbf{w}) = \frac{1}{m} \sum_{i=1}^{m} L^{(i)}(\mathbf{w}).$$

요약: L = 개별샘플 오류, J = 전체 평균 오류(우리가 최소화 하려는 목표).

4 순전파(Forward) & 역전파(Backprop) — 손계산으로 감 잡기

4.1 설정(회귀)

은닉 ReLU, 출력 **선형**. 입력/가중치/정답을 **일부러** 간단히 잡아 계산이 한 번에 눈에 보이도록 한다.

$$
\begin{aligned}
&x_1 = 1,\ x_2 = 2, \qquad y = 2.0\\
&(\text{은닉}1)\ w_{01}^{(1)} = 0.1,\ w_{11}^{(1)} = 0.5,\ w_{21}^{(1)} = 0.3\\
&(\text{은닉}2)\ w_{02}^{(1)} = -0.1,\ w_{12}^{(1)} = 0.4,\ w_{22}^{(1)} = 0.1\\
&(\text{출력})\ w_{0}^{(2)} = 0.2,\ w_{1}^{(2)} = 1.0,\ w_{2}^{(2)} = 0.5
\end{aligned}
$$

1) 순전파

$$
\begin{aligned}
z_1 &= 0.1 + 0.5(1) + 0.3(2) = 1.2 \Rightarrow u_1 = \max(0, 1.2) = 1.2\\
z_2 &= -0.1 + 0.4(1) + 0.1(2) = 0.4 \Rightarrow u_2 = 0.4\\
\hat{y} &= 0.2 + 1.0 \cdot 1.2 + 0.5 \cdot 0.4 = 1.6\\
L &= (\hat{y} - y)^2 = (1.6 - 2)^2 = 0.16
\end{aligned}
$$

2) 역전파(미분)

핵심은 **연쇄법칙(Chain Rule)**. $\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial w}$.

출력층:

$$
\frac{\partial L}{\partial \hat{y}} = 2(\hat{y} - y) = 2(-0.4) = -0.8
$$

$$
\frac{\partial L}{\partial w_0^{(2)}} = -0.8, \quad \frac{\partial L}{\partial w_1^{(2)}} = -0.8 \cdot u_1 = -0.96, \quad \frac{\partial L}{\partial w_2^{(2)}} = -0.8 \cdot u_2 = -0.32.
$$

은닉층: 먼저 은닉 출력에 대한 민감도:

$$
\frac{\partial L}{\partial u_1} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial u_1} = -0.8 \cdot w_1^{(2)} = -0.8, \qquad \frac{\partial L}{\partial u_2} = -0.8 \cdot w_2^{(2)} = -0.4.
$$

ReLU의 도함수 $f'(z) = \mathbf{1}\{z > 0\}$. 여기선 $z_1, z_2 > 0 \Rightarrow f'(z_1) = f'(z_2) = 1$.

$$
\frac{\partial L}{\partial z_1} = -0.8, \quad \frac{\partial L}{\partial z_2} = -0.4.
$$

이제 은닉 가중치:

$$
\begin{aligned}
&\frac{\partial L}{\partial w_{01}^{(1)}} = -0.8 \cdot 1 = -0.8, \quad \frac{\partial L}{\partial w_{11}^{(1)}} = -0.8 \cdot x_1 = -0.8, \quad \frac{\partial L}{\partial w_{21}^{(1)}} = -0.8 \cdot x_2 = -1.6,\\
&\frac{\partial L}{\partial w_{02}^{(1)}} = -0.4 \cdot 1 = -0.4, \quad \frac{\partial L}{\partial w_{12}^{(1)}} = -0.4 \cdot x_1 = -0.4, \quad \frac{\partial L}{\partial w_{22}^{(1)}} = -0.4 \cdot x_2 = -0.8.
\end{aligned}
$$

3) 가중치 업데이트(경사 하강법)

$$w \leftarrow w - \alpha \frac{\partial L}{\partial w}.$$

학습률을 $\alpha = 0.02$로 작게 잡아 **손실이 실제로 줄어드는지** 확인한다.

업데이트:

$$\Delta w = -\alpha \nabla L \Rightarrow \begin{cases} w_0^{(2)} \leftarrow 0.2 + 0.016 = 0.216 \\ w_1^{(2)} \leftarrow 1.0 + 0.0192 = 1.0192 \\ w_2^{(2)} \leftarrow 0.5 + 0.0064 = 0.5064 \\ w_{01}^{(1)} \leftarrow 0.1 + 0.016 = 0.116 \\ w_{11}^{(1)} \leftarrow 0.5 + 0.016 = 0.516 \\ w_{21}^{(1)} \leftarrow 0.3 + 0.032 = 0.332 \\ w_{02}^{(1)} \leftarrow -0.1 + 0.008 = -0.092 \\ w_{12}^{(1)} \leftarrow 0.4 + 0.008 = 0.408 \\ w_{22}^{(1)} \leftarrow 0.1 + 0.016 = 0.116 \end{cases}$$

업데이트 후 순전파(손실 감소 확인):

$$\begin{aligned} z_1 &= 0.116 + 0.516(1) + 0.332(2) = 1.296 \Rightarrow u_1 = 1.296 \\ z_2 &= -0.092 + 0.408 + 0.232 = 0.548 \Rightarrow u_2 = 0.548 \\ \hat{y} &= 0.216 + 1.0192 \cdot 1.296 + 0.5064 \cdot 0.548 \approx 1.8144 \\ L &= (1.8144 - 2)^2 \approx 0.0345 \quad (\textbf{처음}\ 0.16 \rightarrow \textbf{감소}) \end{aligned}$$

교훈: α가 너무 크면 오히려 손실이 커질 수 있다(발산/오버슈팅). 작게부터 시도하고 모니터링하자.

5 경사 하강법(최적화) — 세 가지 모드

- **배치 GD**: 전체 데이터(m개) 평균 그라디언트로 한 번 업데이트. 정확하지만 느림.
- **미니배치 GD**: s개씩 묶어 평균 그라디언트로 업데이트($1 < s < m$). 현업 표준.
- **SGD**: $s = 1$. 가볍고 빠르지만 요동이 큼.

모두 공통 업데이트식은

$$w \leftarrow w - \alpha \cdot \frac{1}{s} \sum_{i \in \text{batch}} \nabla L^{(i)}(w).$$

6 실전 체크리스트(왕초보용)

1. **데이터 분할:** train/validation/test를 시간 순서 존중 또는 무작위로 적절히 나눈다.
2. **스케일링:** 입력을 표준화/정규화하면 학습이 더 안정적.

3. **초기화:** He/Xavier 등 합리적인 초기화(프레임워크 기본값 활용).

4. **기준선:** 평균 예측/최빈값/지속성 같은 **베이스라인**과 꼭 비교.

5. **얼리 스토핑:** 검증 손실이 나빠지면 학습 중단.

6. **학습률 스케줄:** 처음엔 조금 크게, 점점 줄이는 전략도 유효.

7 (선택) 분류 출력층 한눈 요약

이진분류(라벨 0/1)

출력층 활성화 $g = \sigma$ (시그모이드), 손실은 **Binary Cross-Entropy**.

$$\hat{y} = \sigma(z), \quad L = -\big(y \log \hat{y} + (1-y) \log(1-\hat{y})\big).$$

다중분류(라벨 1-of-K)

출력층 활성화 $g = \text{softmax}$, 손실은 **Categorical Cross-Entropy**.

$$\hat{\mathbf{y}} = \text{softmax}(\mathbf{z}), \quad L = -\sum_{c=1}^{K} y_c \log \hat{y}_c.$$

8 자주 헷갈리는 것들(FAQ)

Q1. *Loss*와 *Cost* 차이? 손실(Loss)은 **개별 샘플**의 오차, 비용(Cost)은 **전체 평균** 오차(최소화 대상).

Q2. 기울기(그라디언트)가 0이면 항상 최적? 아니다. **안장점/최대점**일 수도 있다. 실전에서는 검증성능/학습곡선으로 판단.

Q3. 왜 활성화가 꼭 비선형이어야 하나? 선형만 겹치면 전체가 결국 **한 번의 선형변환**과 다를 바 없다. 복잡한 패턴을 못 배운다.

Q4. 학습률은 어떻게 정하지? 작게 시작(예: $10^{-3} \sim 10^{-2}$)해서 **학습곡선**을 보고 조정. 발산하면 더 작게, 너무 느리면 조금 키운다.

Q5. 미니배치 크기 s는? 하드웨어/데이터에 따라 다르지만 32, 64, 128이 무난. 너무 크면 **평탄부**에 갇히기도.

마무리 요약(핵심만 쏙)

- 신경망은 **선형 + 비선형** 변환을 층층이 쌓아 **복잡한 함수**를 근사한다.
- **손실**은 한 샘플, **비용**은 전체 평균. 우리는 비용을 **내리도록** 가중치를 업데이트한다.

- 순전파로 예측, 역전파로 기울기, 경사하강법으로 업데이트. **학습률/미니배치**가 실전 핵심 노브.

Part II

Lecture 2: RNN, LSTM, GRU, Bidirectional RNN — 왕초보도 이해되는 시퀀스 딥러닝

읽기 가이드

- **왜 필요한가?** 문장, 음성, 주가처럼 순서가 중요한 데이터는 이전 정보가 현재/다음 예측에 큰 영향을 준다.
- **핵심 아이디어** RNN은 "이전 시점의 기억"을 들고 다니며 현재 입력을 처리한다.
- **오늘 로드맵** 1) RNN 기본, 2) 기울기 소실/폭발 문제(BPTT), 3) LSTM/GRU로 해결, 4) 입력/출력 구조, 5) 양방향 RNN, 6) 실전 팁/FAQ.

9 RNN(순환 신경망) 기본

9.1 한 줄 요약

피드포워드(정적) 네트워크와 달리, RNN은 시점 t의 은닉상태 h_t가 **이전 상태** h_{t-1}와 **현재 입력** x_t에 의해 결정된다:

$$h_t = \phi\big(W_{xh}x_t + W_{hh}h_{t-1} + b_h\big), \quad y_t = \psi\big(W_{hy}h_t + b_y\big).$$

여기서 ϕ는 보통 tanh나 ReLU, ψ는 작업(회귀/분류)에 맞는 출력 활성화다.

직관적 비유

메모장을 들고 강의를 듣는다고 생각하자. 매 시점(t)마다 강의 슬라이드(x_t)를 보고 메모($h_{t-1} \to h_t$)를 갱신한다. 최종 답(y_t)은 현재 메모 h_t에 달려 있다.

9.2 손계산 미니 예시(짧은 시퀀스)

설정: 스칼라 입력/은닉(이해 용이). $\phi = \tanh$.

$$h_0 = 0, \quad W_{xh} = 0.8,\ W_{hh} = 0.5,\ b_h = 0,$$
$$x_1 = 1.0,\ x_2 = 0.5.$$

순전파:

$$h_1 = \tanh(0.8 \cdot 1.0 + 0.5 \cdot 0) = \tanh(0.8) \approx 0.664$$
$$h_2 = \tanh(0.8 \cdot 0.5 + 0.5 \cdot 0.664) = \tanh(0.4 + 0.332) \approx \tanh(0.732) \approx 0.625.$$

핵심: **과거 정보** h_1이 h_2에 스며든다.

10 BPTT와 기울기 소실/폭발

10.1 BPTT(Backpropagation Through Time)

순환구조를 시간축으로 펼친 뒤(복사본을 늘어놓듯), 일반 역전파를 적용한다. 손실 $L = \sum_t L_t$에 대해

$$\frac{\partial L}{\partial W_{hh}} = \sum_t \frac{\partial L}{\partial h_t}\frac{\partial h_t}{\partial W_{hh}}, \quad \frac{\partial L}{\partial h_t} = \frac{\partial L_t}{\partial h_t} + \frac{\partial L}{\partial h_{t+1}}\frac{\partial h_{t+1}}{\partial h_t}.$$

즉, **미래의 그라디언트**가 **현재로 역전파**된다(연쇄법칙).

10.2 왜 소실/폭발이 생기나?

단순화해 $\phi'(z) \approx \kappa$ (평균적 도함수 크기), $\|W_{hh}\| \approx \rho$라 하면,

$$\left\| \frac{\partial h_t}{\partial h_{t-k}} \right\| \approx (\kappa\rho)^k.$$

- $(\kappa\rho)^k \ll 1$: k가 커질수록 **기울기 소실** (장기 의존 학습 실패)
- $(\kappa\rho)^k \gg 1$: **기울기 폭발** (훈련 불안정/발산)

10.3 대응 전략(바닐라 RNN에서 할 수 있는 것)

- **가중치 초기화** (Xavier/He), **정규화**(LayerNorm), **드롭아웃**
- **Gradient Clipping** (예: $\|g\|_2 > \tau$이면 $g \leftarrow \tau \frac{g}{\|g\|_2}$)
- **짧은 BPTT truncation** (예: 100 스텝씩 끊어 역전파)

하지만 **구조적 해결책**이 더 강력: *LSTM/GRU*.

11 LSTM: 게이트로 기억을 관리

11.1 핵심 아이디어

셀 상태 C_t라는 별도의 "고속도로"를 만들어, 필요한 정보는 오래 유지하고 불필요한 정보는 잊어버리게 한다. 이를 위해 **세 가지 게이트**를 둔다.

$$\begin{aligned}
f_t &= \sigma(W_f[h_{t-1}, x_t] + b_f) && \text{(Forget: 무엇을 지울까?)} \\
i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) && \text{(Input: 무엇을 쓸까?)} \\
\tilde{C}_t &= \tanh(W_C[h_{t-1}, x_t] + b_C) && \text{(새 후보 메모)} \\
C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t && \text{(셀 상태 갱신)} \\
o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) && \text{(Output: 무엇을 드러낼까?)} \\
h_t &= o_t \odot \tanh(C_t) && \text{(은닉 상태)}
\end{aligned}$$

여기서 σ는 시그모이드, $\odot$는 원소별 곱.

왜 소실이 줄어드나?

$C_t = f_t \odot C_{t-1} + \cdots$ 구조 덕분에, $f_t \approx 1$인 경로가 생기면 **그라디언트가 C-경로를 따라 비교적 손실 없이 전달**된다(컨트롤된 잔차 경로).

11.2 숫자 예시(아주 단순화)

스칼라 C, h로 직관만 잡자.

$$
\begin{aligned}
&C_0 = 0,\ h_0 = 0,\ x_1 = 1.0,\ x_2 = 0.5,\\
&f_1 = 0.9,\ i_1 = 0.7,\ \tilde{C}_1 = 0.8 \Rightarrow C_1 = 0.9 \cdot 0 + 0.7 \cdot 0.8 = 0.56,\\
&o_1 = 0.6 \Rightarrow h_1 = 0.6 \cdot \tanh(0.56) \approx 0.6 \cdot 0.509 = 0.305.
\end{aligned}
$$

다음 스텝에서 f_2가 0.95처럼 크게 나오면, C_1에 담긴 정보가 C_2로 잘 보존된다.

12 GRU: LSTM을 더 단순하게

12.1 구조

GRU는 **셀 상태 C_t 없이** 은닉 h_t 하나만 쓴다. 게이트는 두 개:

$$
\begin{aligned}
z_t &= \sigma(W_z[h_{t-1}, x_t] + b_z) && \text{(Update: 얼마를 유지/덮어쓸까?)}\\
r_t &= \sigma(W_r[h_{t-1}, x_t] + b_r) && \text{(Reset: 과거를 얼마나 지울까?)}\\
\tilde{h}_t &= \tanh\big(W[r_t \odot h_{t-1}, x_t] + b\big)\\
h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t.
\end{aligned}
$$

직관

z_t가 작으면 과거 유지, 크면 새 정보로 덮어씀. r_t는 후보 계산 시 과거 얼마나 참고할지 조절. LSTM보다 **파라미터가 적고 계산이 빠른** 경향.

13 입출력 시퀀스 패턴(문제별 배선)

- **일대일(one-to-one):** 전통적 신경망. 고정 크기 입력 → 고정 크기 출력
- **일대다(one-to-many):** 이미지 캡셔닝. 이미지 하나 → 문장 시퀀스
- **다대일(many-to-one):** 감성분석. 문장 시퀀스 → 감성 라벨 하나
- **다대다(many-to-many):** 기계번역, 시퀀스 라벨링

Part III
Lecture 3: 아무것도 몰라도 이해되는 NLP 기초와 토큰화

읽기 가이드

- **목표:** NLP가 무엇을 하고, 왜 어려운지, 어디에 쓰는지, 그리고 토큰화(*Tokenization*)가 왜 첫 단계인지를 완전 기초부터 이해한다.
- **진행:** (1) NLP 정의 & 난점(모호성), (2) 활용 분야, (3) 기초 전처리 중 토큰화 원리와 실전 팁.
- **톤:** 수학식 최소화, 대신 직관과 예시, 바로 써먹는 규칙 중심.

14 NLP란 무엇인가? (What is NLP?)

자연어 처리(NLP, Natural Language Processing)는 **인간 언어를 컴퓨터가 이해하고, 해석하고, 생성**하도록 만드는 기술이다.
NLP는 언어학(형태론, 구문론, 의미론 등)과 인공지능(머신러닝, 딥러닝)을 **융합**한다.

한 줄 정의

$$\text{NLP} = (\text{언어학 지식}) + (\text{AI/ML 기법}) \Rightarrow \text{언어를 데이터로 다루는 방법}$$

직관적 예시

- **스팸 필터:** 이메일 본문을 읽고 스팸 여부(0/1)를 분류
- **감성 분석:** 리뷰가 긍정/부정인지 예측
- **기계 번역:** 한국어 ↔ 영어 문장 변환
- **질의응답:** "파리는 어느 나라 수도야?" ⇒ "프랑스"

15 왜 NLP가 어려운가? (언어의 모호성)

인간 언어는 **모호**하다. 같은 문장도 문맥에 따라 뜻이 달라질 수 있다.

모호한 제목(헤드라인) 예시

- "*Enraged Cow Injures Farmer with Ax*"
  (소가 분노해서 농부를 도끼로 다치게 했다고? 도끼는 누구의?)
- "*Teacher Strikes Idle Children*"
  (교사가 파업을 했다는 뜻? 아이들을 때렸다는 뜻?)
- "*Stolen Painting Found by Tree*"
  (나무가 그림을 찾았다고? 나무 옆에서 발견되었다는 뜻?)

핵심 컴퓨터에게는 **문맥/상식/세상 지식**이 부족하기 때문에, 기호(문자열)만 보고 의미를 정확히 파악하기가 어렵다. 그래서 전처리와 표현학습(예: 임베딩), 그리고 강력한 모델링(딥러닝)이 중요하다.

16 NLP의 대표적 응용 분야(개념 지도)

현대 NLP는 매우 넓다. 아래는 **가장 자주 등장하는 작업들**이다.

1. **텍스트 분류** (스팸 탐지, 주제 분류, 감성 분석)
2. **개체명 인식(NER)** (사람/조직/지명 등 고유명 추출)
3. **정보검색(IR)** (검색엔진, 문서 랭킹: TF–IDF, BM25 등)
4. **요약(Summarization)** (추출식/생성식)
5. **기계 번역(MT)** (NMT, Transformer)
6. **질의응답(Q&A)** (정답 스팬 추출/생성)
7. **음성 인식(ASR)** (음성 → 텍스트: 음향+언어 모델)
8. **OCR** (이미지 속 문자 → 텍스트)
9. **대화형 에이전트/챗봇** (NLU+NLG)

공통 전처리 위의 거의 모든 작업은 **텍스트 전처리**에서 시작한다. 그 첫 단추가 **토큰화**다.

17 기초 전처리 1: Tokenization(토큰화)

17.1 정의

토큰화는 원시 텍스트를 **더 작은 단위(토큰)**로 나누는 과정이다. 토큰은 문장, 단어, 서브워드, 문자가 될 수 있다.
이 단계에서 우리는 **어휘집(vocabulary)**을 만들고, 이후 파이프라인(표현학습, 모델링)의 입구를 준비한다.

17.2 왜 중요한가?

- **모델 입력의 기준**이 된다. (무엇을 하나의 단위로 볼 것인가?)
- **어휘 크기**와 **희소성(OOV)**을 좌우한다. (일반화 성능과 직결)
- 후속 작업(구문분석, 표제어 추출, 품사 태깅)의 **정확도**에 영향

17.3 대표 토큰화 방식과 예시

(1) 단어 토큰화(Word) 문장을 단어 단위로 분할한다.

"Karl Benz invented the first car." ⇒ ["Karl", "Benz", "invented", "the", "first", "car", "."]

(2) 문장 토큰화(Sentence) 문단을 문장 단위로 분할한다.

"Henry Ford created assembly lines. This revolutionized car production." ⇒ $\begin{bmatrix}\text{"Henry Ford created ass}\\ \text{"This revolutionized car}\end{bmatrix}$

(3) 서브워드 토큰화(Subword) 단어를 더 작은 서브워드로 쪼갠다(BPE, WordPiece 등).

"Hydrogen-powered" ⇒ ["Hydrogen", "-", "powered"]

서브워드는 희소 단어(OOV)를 줄이고, 접사/합성어를 유연하게 다룬다. 현대 대형 언어모델(예: BERT, GPT)은 주로 서브워드를 사용한다.

17.4 실전에서 자주 부딪히는 케이스

- **구두점:** 마침표/쉼표/따옴표 등은 별도 토큰으로 분리하는 것이 일반적이다.
- **축약형:** `don't`, `it's`는 `do n't`, `it 's`로 나눌지 말지 규칙이 다르다.
- **고유명사:** `New York`, `San Francisco` 같은 복합어는 단어 2개지만 하나의 개체다.
- **하이픈/합성어:** `state-of-the-art`, `Hydrogen-powered` 처리 규칙 정의 필요.
- **언어별 특성:** 한국어/일본어처럼 공백이 확실한 경계가 아닌 언어는 형태소 분석기 기반 토큰화가 흔하다.

17.5 NLTK로 단어/문장 토큰화(아이디어)

```
# (아이디어 예시) NLTK
import nltk
nltk.download('punkt')

from nltk.tokenize import word_tokenize, sent_tokenize

text = "Henry Ford created assembly lines. This changed the industry."

print(word_tokenize(text))
# ['Henry', 'Ford', 'created', 'assembly', 'lines', '.', 'This', 'changed', 'the', 'i

print(sent_tokenize(text))
# ['Henry Ford created assembly lines.', 'This changed the industry.']
```

17.6 서브워드 토큰화가 필요한 순간

- **어휘 부풀림**(*vocab explosion*)을 막고 싶을 때
- 드물게 등장하는 신조어/합성어/전문어가 많을 때
- 사전 기반(OOV 민감) 단어 토큰화로는 일반화가 떨어질 때

17.7 토큰화 품질을 올리는 5가지 체크리스트

1. **정규화(Normalization):** 대소문자 통일, 유니코드 정규화(NFC/NFKC), 공백 정리
2. **숫자/기호 정책:** 숫자를 모두 <num>로 치환할지, 기호를 남길지 결정
3. **언어별 규칙:** 영어(축약형), 한국어(조사/어미), 독어(합성어) 등 맞춤 룰
4. **도메인 적합성:** 법률/의료/코드 등 분야별 특수 토큰(약어, 섹션표기) 고려
5. **재현성:** 토크나이저 버전/옵션을 고정해 실험 & 배포 일관성 확보

정리 (Part 1)

- NLP는 **언어학** + **AI**의 융합 분야로, 스팸/감성/번역/요약/QA 등 광범위하게 쓰인다.
- 언어는 **모호**하고 **문맥 의존**적이어서, 전처리와 표현학습이 매우 중요하다.
- **토큰화**는 모든 파이프라인의 첫 단계이며, 단어/문장/서브워드 중 작업에 맞는 단위를 고르는 것이 핵심이다.