

## □ 강의 개요

### 학습 목표:

- 이 강의의 핵심 개념을 이해합니다
- 실전에 적용할 수 있는 지식을 습득합니다

주요 키워드: [자동으로 채워질 예정]

선행 지식: 기본적인 컴퓨터 사용 능력

## Contents

<b>1</b>	<b>학습 로드맵 및 핵심 용어</b>	<b>3</b>
1.1	용어 정리	3
<b>2</b>	<b>강의 운영 및 평가</b>	<b>4</b>
2.1	소통 채널 및 강의 세션	4
2.2	평가 기준 및 정책	4
2.3	과제 제출 가이드	5
<b>3</b>	<b>신경망 입문 (Introduction to Neural Networks)</b>	<b>6</b>
3.1	선형 회귀에서 신경망으로	6
3.2	순방향 신경망 (Feedforward Neural Network, FNN)	6
3.3	활성화 함수 (Activation Functions)	7
<b>4</b>	<b>신경망 훈련 (Training Neural Networks)</b>	<b>8</b>
4.1	1단계: 손실 함수 (Loss Function) 정의	8
4.1.1	회귀 (Regression) 문제의 손실 함수	8
4.1.2	분류 (Classification) 문제의 손실 함수	8
4.2	2단계: 최적화 알고리즘 (Optimization Algorithm) 선택	9
<b>5</b>	<b>1페이지 요약</b>	<b>11</b>

# CSCI E-89B 자연어 처리 입문

1주차: 신경망 기초 및 훈련 방법론

Dmitry Kurochkin 교수 강의 기반 정리

October 26, 2025

■ 강의명: CSCI E-89B: 자연어 처리 입문

■ 주차: Lecture 01

■ 교수명: Dmitry Kurochkin

■ 목적: Lecture 01의 핵심 개념 학습

## □ 핵심 요약

본 문서는 하버드 익스텐션 스쿨의 CSCI E-89B 자연어 처리 입문 강의 내용을 통합 정리한 노트입니다.

- 강의 운영 방식, 평가 기준, 과제 제출법 등 학사 정보를 명확히 안내합니다.
- 딥러닝의 기초가 되는 신경망(Neural Network)의 개념을 다룹니다.
- 신경망의 핵심 구성 요소인 활성화 함수(Activation Function)의 종류와 역할을 설명합니다.
- 모델을 학습시키는 원리인 손실/비용 함수(Loss/Cost Function)와 최적화 알고리즘(Optimization Algorithm)을 배웁니다.
- 이론적 개념을 실제 코드로 구현하는 Keras 예제를 포함합니다.

# 1 학습 로드맵 및 핵심 용어

## □ 핵심 정보

### 학습 로드맵

1. 기초 다지기: 강의 운영 방식을 숙지하고, 신경망의 기본 아이디어를 이해합니다.
2. 핵심 개념: 활성화 함수, 손실 함수, 비용 함수의 정의와 역할을 명확히 구분합니다.
3. 훈련 원리: 경사 하강법 (GD), 확률적 경사 하강법 (SGD), 미니배치 경사 하강법의 차이를 비교하고 이해합니다.
4. 실습 적용: Keras 코드를 통해 신경망을 구축하고, 다양한 손실 함수와 옵티마이저를 적용하는 방법을 익힙니다.
5. 심화: 최종 프로젝트를 염두에 두고, 다양한 문제(회귀, 분류)에 어떤 함수와 알고리즘이 적합할지 고민합니다.

## 1.1 용어 정리

Table 1: 자연어 처리 및 신경망 핵심 용어

용어	쉬운 설명	원어	비고
신경망	인간의 뇌 신경을 모방한 데이터 학습 모델	Neural Network (NN)	입력
활성화 함수	뉴런의 활성/비활성을 결정하는 비선형 함수	Activation Function	ReLU
손실 함수	단일 데이터에 대한 모델 예측과 실제 값의 차이	Loss Function	예: 가
비용 함수	전체 데이터셋에 대한 손실의 평균	Cost Function	훈련
경사 하강법	비용 함수를 최소화하기 위해 파라미터를 업데이트하는 방법	Gradient Descent	기울
원-핫 인코딩	범주형 데이터를 0과 1의 벡터로 변환하는 기법	One-Hot Encoding	고양
하이퍼파라미터	모델이 학습하지 않고, 사용자가 직접 설정하는 값	Hyperparameter	학습
에포크	전체 훈련 데이터셋을 한 번 모두 사용한 훈련 주기	Epoch	

## 2 강의 운영 및 평가

### 2.1 소통 채널 및 강의 세션

효율적인 학습과 소통을 위해 목적에 따라 다른 채널을 사용합니다.

#### 소통 채널 가이드

- **Piazza (피아자):** 강의 내용, 과제에 대한 공식적인 질문을 위한 공간입니다. 교수 및 조교(TA)가 답변하며, 다른 학생들과 질문과 답변을 공유할 수 있습니다.
- **WhatsApp (왓츠앱):** 학생들 간의 자유로운 토론 및 정보 교류를 위한 비공식 채널입니다. 교수나 조교가 항상 확인하지는 않으므로, 공식적인 답변이 필요하다면 Piazza를 사용해야 합니다.
- **Canvas Inbox (캔버스 인박스):** 개인적인 사안에 대해 교수나 조교에게 직접 연락할 때 사용합니다.

Table 2: 주간 세션 일정

세션 종류	요일 (예상)	주요 내용
강의 (Lecture)	화요일	핵심 이론 및 개념 설명
조교 세션 1 (TA Session)	수요일 또는 목요일	이론 복습, 예제 풀이 (1)
강사 세션 (Instructor's Section)	금요일	Python 코드 구현 예제, 심화 토론
조교 세션 2 (TA Session)	토요일 또는 일요일	이론 복습, 예제 풀이 (2)

모든 세션은 녹화되어 제공되므로 실시간 참여가 어려워도 학습이 가능합니다. 단, 두 조교 세션은 서로 다른 내용을 다루므로 중복되지 않습니다.

### 2.2 평가 기준 및 정책

최종 성적은 과제, 퀴즈, 최종 프로젝트의 점수를 합산하여 산출됩니다.

Table 3: 성적 평가 비중

항목	비중
주간 과제 (Homework Assignments)	65%
주간 퀴즈 (Quizzes)	20%
최종 프로젝트 (Final Project)	15%

#### 주의사항

##### 제출 기한 정책

- **퀴즈:** 절대 지각 제출 불가. 퀴즈 마감 직후 다음 수업에서 해설이 진행되기 때문에, 시스템적으로 재응시 기회를 제공하기 어렵습니다. 7일의 충분한 기간 내에 제출해야 합니다.
- **과제:** 매주 일요일 23:59 (보스턴 시간 기준) 마감. 지각 제출 시 정해진 비율에 따라 감점

됩니다 (1일 지각 시 10%, 2일 20% ... 5일 100%).

- **최종 프로젝트:** 마감 기한이 매우 엄격하며, 연장이 거의 불가능합니다. 특별한 사유가 있을 시, 사전에 Extension School을 통해 공식적인 절차를 밟아야 합니다. 이 강의에서는 가장 낮은 점수의 퀴즈나 과제를 제외하는 정책이 적용되지 않습니다.

## 2.3 과제 제출 가이드

과제는 코드와 보고서를 함께 제출해야 합니다.

### 과제 제출 체크리스트

#### 1. 보고서 (Report) 작성:

- MS Word 또는 PDF 형식으로 제출합니다.
- 문제 해결의 핵심이 되는 코드 일부, 결과(플롯, 표), 그리고 결과에 대한 간단한 논의를 포함해야 합니다.
- Jupyter Notebook에서 직접 PDF로 변환하여 제출하는 것도 허용됩니다.

#### 2. 소스 코드 (Source Code) 제출:

- 조교가 코드를 직접 실행하고 검토할 수 있도록 원본 코드를 반드시 제출해야 합니다. (예: .ipynb 파일)
- 파일이 여러 개일 경우, 하나의 .zip 파일로 압축하여 제출합니다.

#### 3. 최종 제출물 확인: 보고서 파일 1개와 소스 코드 파일(또는 ZIP 파일) 1개를 모두 제출했는지 확인합니다.

### 3 신경망 입문 (Introduction to Neural Networks)

#### 3.1 선형 회귀에서 신경망으로

전통적인 머신러닝 모델인 선형 회귀 (Linear Regression)는 데이터의 패턴을 잘 표현하기 위해 사람이 직접 **특성(feature)**을 설계해야 했습니다. 예를 들어, 비선형 관계를 표현하기 위해 입력값  $x$  뿐만 아니라  $x^2, x^3$  같은 항을 직접 추가하는 방식입니다.

$$\hat{y} = w_0 + w_1 \cdot x + w_2 \cdot x^2 + w_3 \cdot x^3$$

하지만 이미지나 텍스트 같은 복잡한 데이터에서는 사람이 유의미한 특성을 직접 설계하기가 거의 불가능합니다. **신경망(Neural Network)**은 이러한 특성을 데이터로부터 **자동으로 학습**하는 모델입니다. 여러 개의 층(layer)을 쌓아, 데이터의 표현(representation)을 점진적으로 학습해 나갑니다.

#### 3.2 순방향 신경망 (Feedforward Neural Network, FNN)

가장 기본적인 신경망 구조로, 입력층에서 출력층으로 정보가 한 방향으로만 흐릅니다. 수학적으로는 **함수들의 중첩(nested functions)**으로 표현할 수 있습니다.

$$\hat{y} = f^{(L)}(f^{(L-1)}(\dots f^{(1)}(x)))$$

여기서 각 함수  $f^{(l)}$ 은 보통 **비선형 활성화 함수**가 적용된 **선형 변환**의 형태를 가집니다. 예를 들어, 2개의 입력( $x_1, x_2$ )과 1개의 은닉층(hidden layer)을 갖는 간단한 신경망의 출력  $\hat{y}$ 는 다음과 같이 계산됩니다.

$$\begin{aligned} u_1 &= f(w_{01}^{(1)} + w_{11}^{(1)}x_1 + w_{21}^{(1)}x_2) \\ u_2 &= f(w_{02}^{(1)} + w_{12}^{(1)}x_1 + w_{22}^{(1)}x_2) \\ \hat{y} &= f(w_0^{(2)} + w_1^{(2)}u_1 + w_2^{(2)}u_2) \end{aligned}$$

#### □ 핵심 정보

**왜 중간에 선형 변환을 사용할까?** 신경망의 핵심은 비선형성을 표현하는 것이지만, 각 단계에서 선형 변환( $w \cdot x + b$ )을 사용하는 이유는 **미분 가능성** 때문입니다. 최적화 과정에서 경사 하강법을 사용하려면 모델을 파라미터로 미분해야 하는데, 선형 함수는 미분이 매우 간단합니다. 연쇄 법칙(chain rule)을 통해 복잡한 신경망 전체의 미분도 효율적으로 계산할 수 있습니다.

### 3.3 활성화 함수 (Activation Functions)

활성화 함수는 신경망에 비선형성(non-linearity)을 부여하는 핵심 요소입니다. 만약 활성화 함수가 없다면, 여러 층을 쌓더라도 결국 하나의 선형 변환과 같아져 복잡한 패턴을 학습할 수 없습니다.

#### 주의사항

생물학적 뉴런과의 비유 활성화 함수라는 이름은 뇌의 생물학적 뉴런에서 유래했습니다. 뉴런은 여러 다른 뉴런으로부터 신호를 받아, 그 신호의 합이 특정 임계값(threshold)을 넘으면 활성화(activate)되어 다음 뉴런으로 신호를 전달합니다. 초기 인공 신경망은 이를 모방하여 계단 함수(step function)를 사용했지만, 이 함수는 미분이 불가능한 지점이 있어 경사 하강법에 적합하지 않습니다.

Table 4: 주요 활성화 함수 비교

함수	수식	특징	주요 용도
<b>ReLU</b> (Rectified Linear Unit)	$\max(0, x)$	계산이 빠르고, 널리 사용됨. 음수 입력에 대해 0을 출력.	은닉층 (Hidden layers)
<b>Leaky ReLU</b>	$\max(0.1x, x)$	ReLU의 변형. 음수 입력에도 작은 기울기(0.1)를 가짐.	은닉층
<b>Sigmoid</b>	$\frac{1}{1+e^{-x}}$	출력을 (0, 1) 사이로 압축.	이진 분류(Binary classification) 출력층
<b>Softmax</b>	$\frac{e^{z_i}}{\sum_j e^{z_j}}$	다차원 입력의 출력을 합이 1인 확률 분포로 변환.	다중 클래스 분류(Multi-class) 출력층

```

1 import keras
2 from keras import models, layers
3
4 model = models.Sequential()
5 # 은닉층 1: 개 16 뉴런, ReLU 활성화함수
6 model.add(layers.Dense(16, activation='relu', input_shape=(900,)))
7 # 출력층: 개 2 뉴런, Softmax 활성화함수다중 (분류)
8 model.add(layers.Dense(2, activation='softmax'))
9
10 model.summary()

```

Listing 1: Keras를 이용한 활성화 함수 지정 예시

## 4 신경망 훈련 (Training Neural Networks)

신경망 훈련의 목표는 모델의 예측값( $\hat{y}$ )과 실제 정답( $y$ ) 사이의 오차를 최소화하는 파라미터(가중치  $w$ 와 편향  $b$ )를 찾는 것입니다. 이 과정은 손실 함수, 비용 함수, 최적화 알고리즘 세 가지 요소로 구성됩니다.

### 4.1 1단계: 손실 함수 (Loss Function) 정의

손실 함수는 하나의 데이터 샘플에 대한 모델의 오차를 측정하는 함수입니다. 어떤 문제를 푸느냐에 따라 적절한 손실 함수를 선택해야 합니다.

#### 손실 함수 (Loss) vs. 비용 함수 (Cost)

- **손실 함수 (Loss Function):** 사과 하나가 얼마나 썩었는지 보는 것. 즉, 개별 데이터 포인트  $(x^{(i)}, y^{(i)})$  하나에 대한 예측 오차  $L^{(i)}(w)$ 를 측정합니다.
- **비용 함수 (Cost Function):** 사과 상자 전체가 평균적으로 얼마나 썩었는지 보는 것. 즉, 전체 데이터셋에 대한 손실의 평균  $J(w) = \frac{1}{m} \sum_{i=1}^m L^{(i)}(w)$ 를 측정합니다. 훈련의 실제 목표는 이 비용 함수를 최소화하는 것입니다.

#### 4.1.1 회귀 (Regression) 문제의 손실 함수

연속적인 값을 예측하는 문제에 사용됩니다.

- **제곱 오차 (Squared Error):**  $L(w) = (\hat{y} - y)^2$ 
  - 오차가 클수록 패널티를 더 크게 부여합니다.
  - 수학적으로 다루기 쉽고 미분이 용이하여 널리 쓰입니다.
- **절대 오차 (Absolute Error):**  $L(w) = |\hat{y} - y|$ 
  - 이상치 (outlier)에 덜 민감합니다.
  - 최솟값 지점에서 미분이 불가능한 단점이 있습니다.

#### 4.1.2 분류 (Classification) 문제의 손실 함수

데이터를 특정 카테고리로 분류하는 문제에 사용됩니다.

- **교차 엔트로피 (Cross-Entropy):**  $L(w) = -\sum_{j=1}^M y_j \log(\hat{y}_j)$ 
  - 모델이 예측한 확률 분포( $\hat{y}$ )와 실제 레이블의 원-핫 인코딩 분포( $y$ ) 사이의 차이를 측정합니다.
  - 모델이 정답을 높은 확률로 맞추면 손실이 0에 가까워지고, 틀린 답을 높은 확률로 예측하면 손실이 무한대에 가깝게 커집니다.



## 주의사항

분류 문제에 제공 오차를 쓰면 안 되는 이유 분류 문제의 레이블(예: [1, 0])과 모델의 확률 예측(예: [0.9, 0.1]) 사이에 제공 오차를 사용하면, 비용 함수의 형태가 매우 비선형적이고 복잡해져 나쁜 지역 최솟값(bad local minima)에 빠지기 쉽습니다. 이는 최적화 과정을 매우 불안정하게 만들어 좋은 성능을 얻기 어렵게 합니다. 따라서 분류 문제에는 교차 엔트로피를 사용하는 것이 표준입니다.

## 4.2 2단계: 최적화 알고리즘 (Optimization Algorithm) 선택

최적화 알고리즘은 비용 함수  $J(w)$ 를 최소화하기 위해 파라미터  $w$ 를 반복적으로 업데이트하는 방법입니다. 대부분의 알고리즘은 경사 하강법(Gradient Descent)에 기반합니다.

Table 5: 주요 경사 하강법 알고리즘 비교

알고리즘	업데이트 단위	장점	단점
경사 하강법 (GD)	전체 데이터셋	안정적, 전역 최솟값 수렴(볼록 함수)	계산 비용 매우 높음,
확률적 경사 하강법 (SGD)	데이터 1개	계산 빠름, 지역 최솟값 탈출 용이	매우 불안정, 수렴 속
미니배치 경사 하강법	데이터 N개 (배치)	GD와 SGD의 장점 절충, 병렬화 용이	배치 크기라는 하이퍼

## □ 핵심 정보

미니배치와 "수프 스푼" 비유 큰 솥에 끓고 있는 수프의 간을 볼 때, 수프 전체를 마실 필요도 없고, 숟가락 크기가 솥의 크기에 비례할 필요도 없습니다. 적당한 크기의 스푼 하나면 충분합니다. 마찬가지로, 거대한 데이터셋(수프 솥)의 전체적인 경사(간)를 추정할 때, 적절한 크기의 미니배치(스푼)만 사용해도 충분히 효율적입니다. 데이터셋이 100만 개든 1억 개든, 32나 64 크기의 미니배치가 효과적인 이유입니다.

파라미터 업데이트 규칙:

$$w_{\text{new}} := w_{\text{old}} - \alpha \nabla J(w_{\text{old}})$$

여기서  $\alpha$ 는 학습률(learning rate)로, 얼마나 큰 보폭으로 이동할지를 결정하는 중요한 하이퍼 파라미터입니다.

```

1 model.compile(
2     optimizer='adam', # Adam 옵티마이저사용
3     loss='categorical_crossentropy', # 다중분류용교차엔트로피
4     metrics=['accuracy'] # 훈련중정확도를모니터링
5 )
6
7 # 모델훈련
8 history = model.fit(
9     X_train, y_train,
10    batch_size=128, # 미니배치크기
11    epochs=35, # 전체데이터셋반복횟수
12    validation_data=(X_test, y_test)

```

13 )

## Listing 2: Keras에서 손실 함수와 옵티마이저 지정하기

Adam은 현재 가장 널리 쓰이는 진보된 옵티마이저 중 하나입니다.

## 5 1페이지 요약

### 핵심 개념 킷 리뷰

#### 1. 신경망의 기본 구조

- **역할:** 데이터로부터 특성을 자동으로 학습하는 모델.
- **구조:** 입력층 → 은닉층(들) → 출력층
- **원리:** 비선형 활성화 함수와 선형 변환의 중첩.

$$\hat{y} = f(W \cdot x + b)$$

#### 2. 활성화 함수

- **역할:** 모델에 비선형성을 부여하여 복잡한 패턴 학습을 가능하게 함.
- **은닉층용: ReLU** ( $\max(0, x)$ )가 가장 보편적.
- **출력층용:**
  - 이진 분류: **Sigmoid**
  - 다중 분류: **Softmax**

#### 3. 손실 함수와 비용 함수

- **손실 함수:** 단일 데이터의 오차 측정.
- **비용 함수:** 전체 데이터셋의 평균 손실.
- **회귀 문제:** 제곱 오차(MSE), 절대 오차(MAE)
- **분류 문제:** 교차 엔트로피

#### 4. 최적화 알고리즘

- **목표:** 비용 함수를 최소화하는 파라미터 찾기.
- **핵심:** 경사 하강법.
- **실용적 선택:** 미니배치 경사 하강법
- **주요 하이퍼파라미터:** 학습률 ( $\alpha$ ), 배치 크기.