DistilBERT를 활용한 텍스트 분류

MLP Lab

Contents

02 텍스트 분류

01 데이터 로드 및 분석

1-1. 데이터 불러오기 및 데이터프레임으로 변환

```python
from datasets import load_dataset

emotions = load_dataset('emotion')
```

- set_format 메서드를 사용해 데이터프레임 변환

```python
import pandas as pd

emotions.set_format(type='pandas')
df = emotions['train'][:]
df.head()
```

	text	label
0	i didnt feel humiliated	0
1	i can go from feeling so hopeless to so damned...	0
2	im grabbing a minute to post i feel greedy wrong	3
3	i am ever feeling nostalgic about the fireplac...	2
4	i am feeling grouchy	3

01 데이터 로드 및 분석

1-1. 데이터 불러오기 및 데이터프레임으로 변환

```
from datasets import load_dataset

emotions = load_dataset('emotion')
```

• set_format 메서드를 사용해 데이터프레임 변환

```
import pandas as pd

emotions.set_format(type='pandas')
df = emotions['train'][:]
df.head()
```

	text	label
0	i didnt feel humiliated	0
1	i can go from feeling so hopeless to so damned...	0
2	im grabbing a minute to post i feel greedy wrong	3
3	i am ever feeling nostalgic about the fireplac...	2
4	i am feeling grouchy	3

01 데이터 로드 및 분석

1-2. label_name 필드 추가

- int2str 메서드를 사용해 label ID를 label_name으로 매핑

```
def label_int2str(row):
    return emotions['train'].features['label'].int2str(row)

df['label_name'] = df['label'].apply(label_int2str)
df.head()
```

	text	label	label_name
0	i didnt feel humiliated	0	sadness
1	i can go from feeling so hopeless to so damned...	0	sadness
2	im grabbing a minute to post i feel greedy wrong	3	anger
3	i am ever feeling nostalgic about the fireplac...	2	love
4	i am feeling grouchy	3	anger

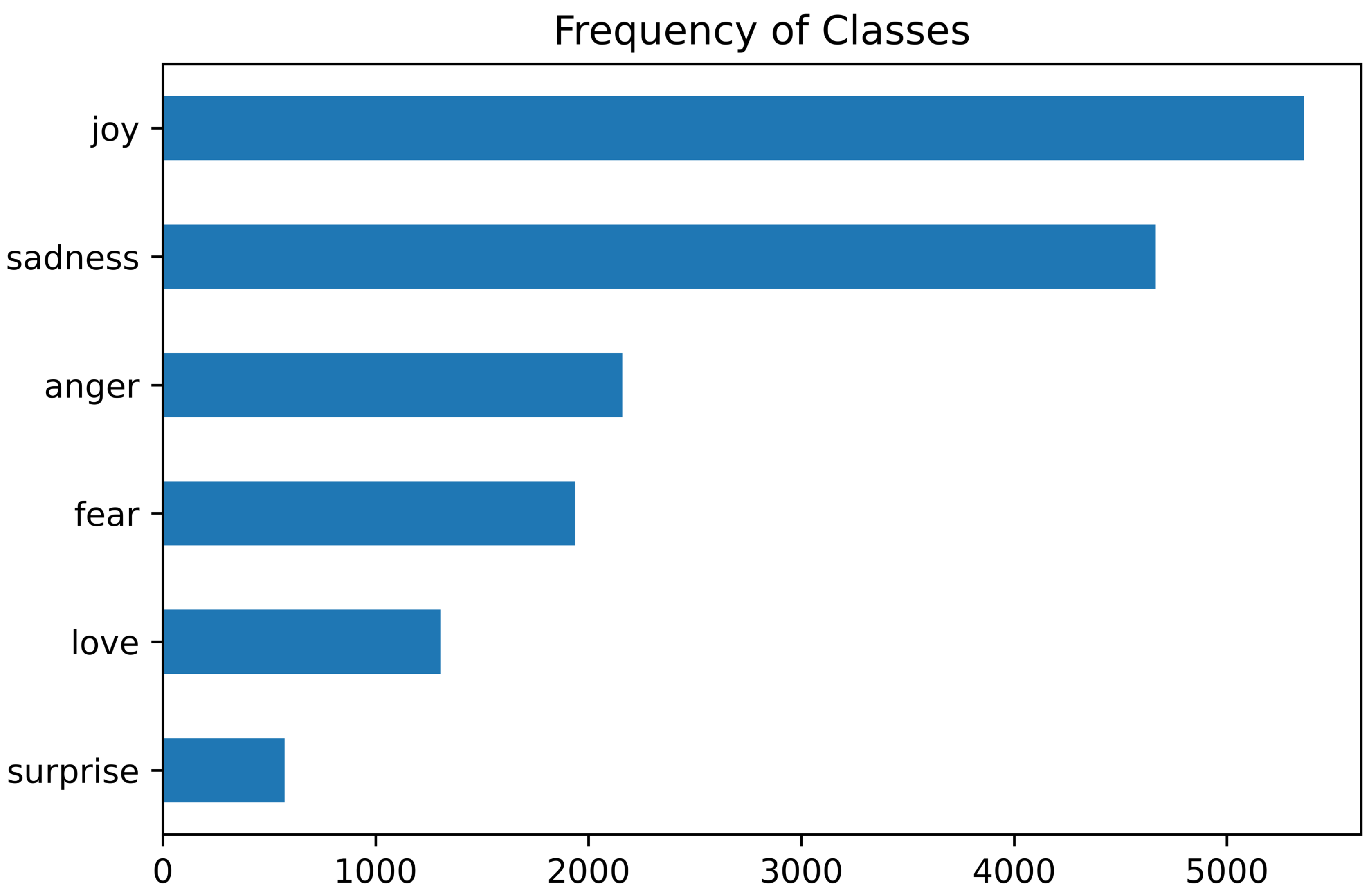
01 데이터 로드 및 분석

1-3. Label 별 분포 확인하기 (EDA)

- 데이터 시각화와 관련된 다양한 도구들을 제공하는 matplotlib를 사용해 시각화

```
import matplotlib.pyplot as plt
%matplotlib inline

df['label_name'].value_counts(ascending=True).plot.barh()
plt.title('Frequency of Class')
plt.show()
```

Label 별 막대그래프로 분포를 살펴보니 클래스 불균형 문제가 발견됨 클래스 불균형은 분류기의 성능을 저하시킬 수 있음



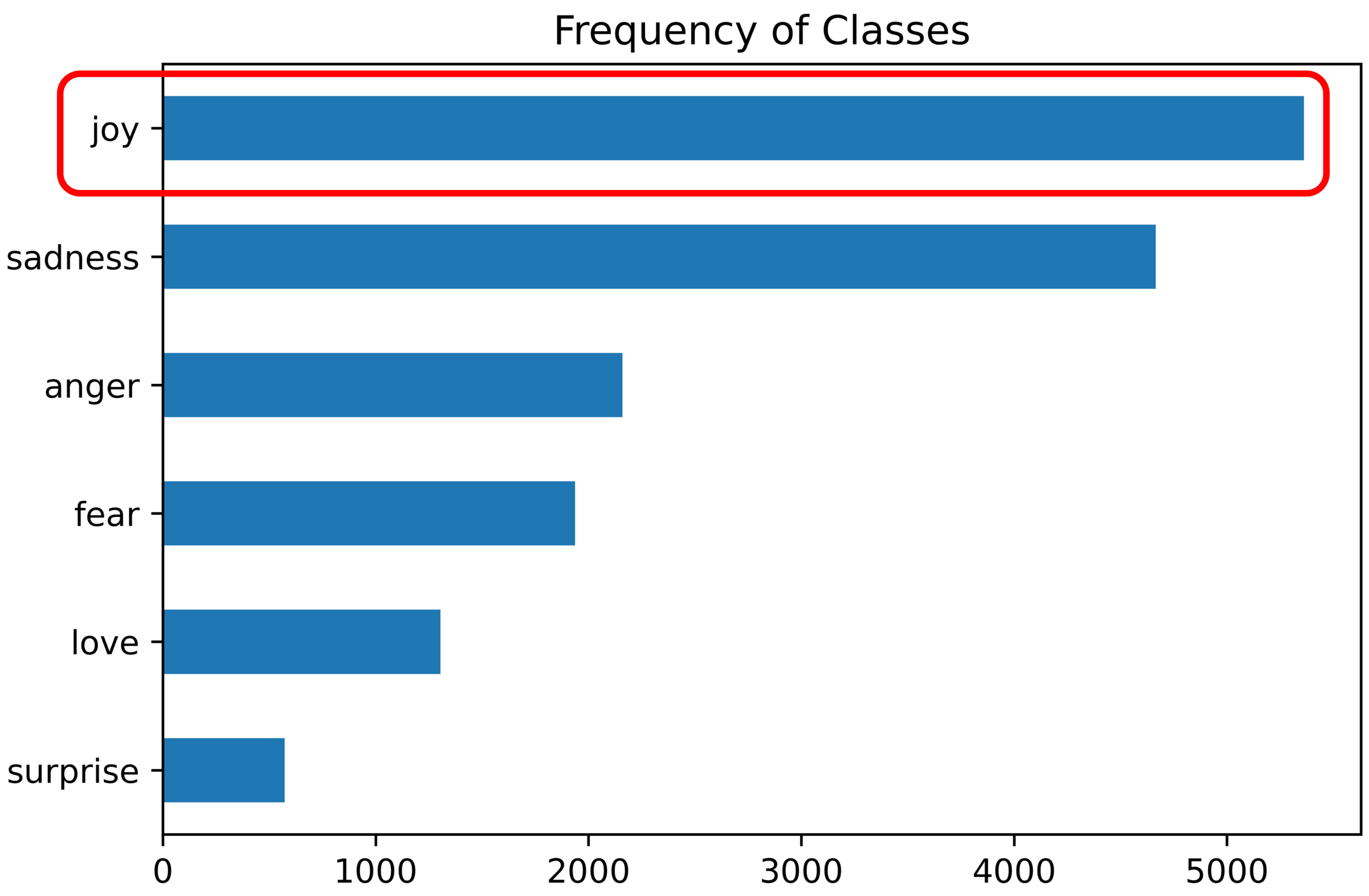
01 데이터 로드 및 분석

1-3. Label 별 분포 확인하기 (EDA)

- 데이터 시각화와 관련된 다양한 도구들을 제공하는 matplotlib를 사용해 시각화

```
import matplotlib.pyplot as plt
%matplotlib inline

df['label_name'].value_counts(ascending=True).plot.barh()
plt.title('Frequency of Class')
plt.show()
```

Label 별 막대그래프로 분포를 살펴보니 클래스 불균형 문제가 발견됨 클래스 불균형은 분류기의 성능을 저하시킬 수 있음

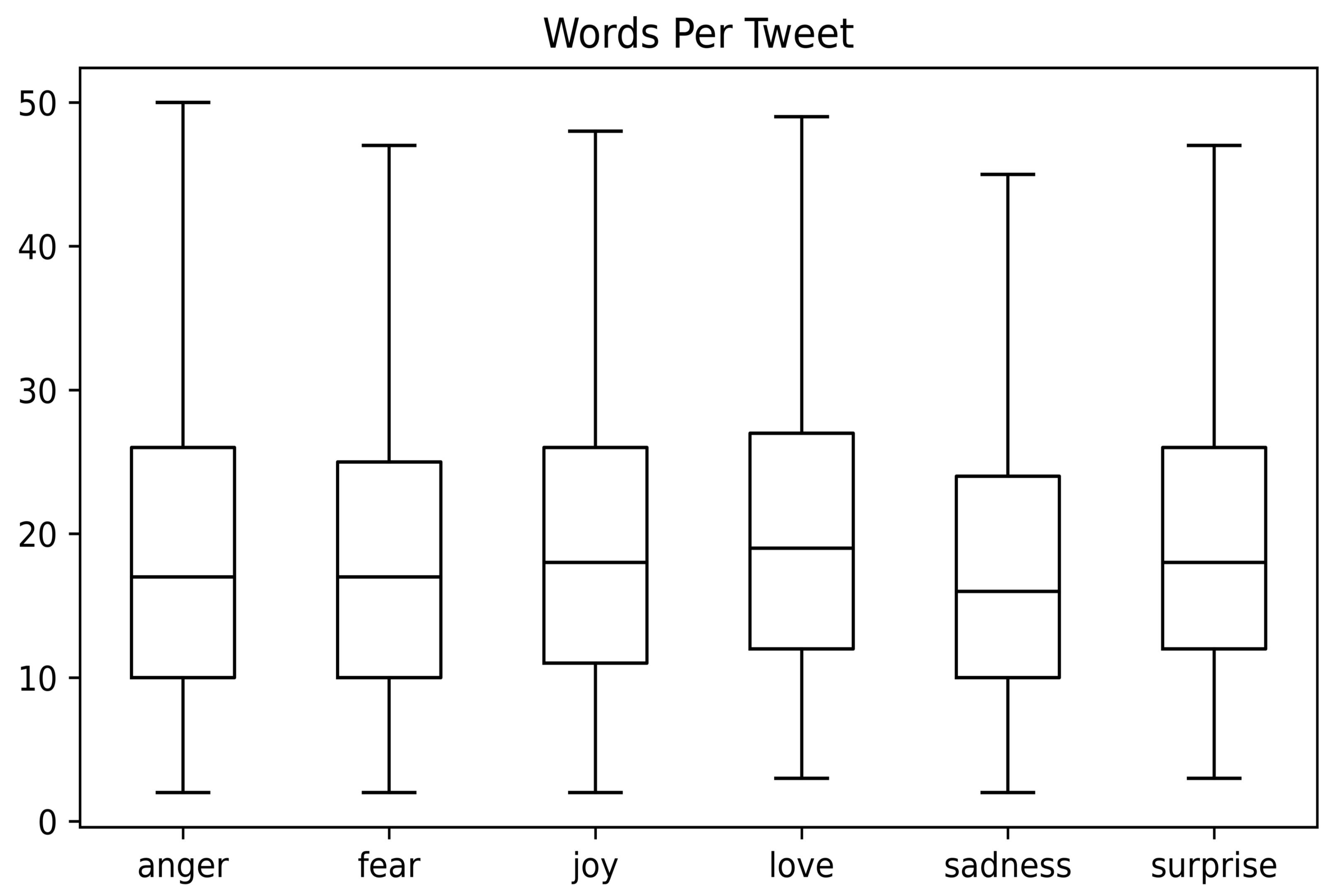


01 데이터 로드 및 분석

1-3. 각 데이터의 단어 수 분포 확인하기 (EDA)

```
df['Tweet Per Words'] = df['text'].str.split().apply(len)
df.boxplot('Tweet Per Words',by='label_name',showfliers=False,grid=False)
plt.show()
```

Boxplot으로 각 데이터(트윗)별 단어 수 분포를 확인해보니 Label 별로 15개 정도에서 중간값을 나타내고 있고 최대 50개 정도 되는것을 확인할 수 있음



단어 수를 확인하는 이유 → 모델의 최대 문맥 길이를 고려해야하기 때문에

02 Model

- DistilBERT

= “Distil”(축소된) + BERT

✓ 기존 BERT의 경량화 모델

DistilBERT는 기존 BERT 모델에 지식증류 기법을 통해 학습한 모델이며 기존 BERT 모델에 비해 40% 정도 가벼워졌고 BERT의 97%정도의 성능을 보이며 60%정도 빨라졌다.

지식증류(Knowledge distillation)
-> 큰 모델(BERT)에서 증류한 지식을 작은모델(DistilBERT)에 전이하는 일련의 과정

DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter

Victor SANH, Lysandre DEBUT, Julien CHAUMOND, Thomas WOLF
Hugging Face
{victor,lysandre,julien,thomas}@huggingface.co

Abstract

As Transfer Learning from large-scale pre-trained models becomes more prevalent in Natural Language Processing (NLP), operating these large models in on-the-edge and/or under constrained computational training or inference budgets remains challenging. In this work, we propose a method to pre-train a smaller general-purpose language representation model, called DistilBERT, which can then be fine-tuned with good performances on a wide range of tasks like its larger counterparts. While most prior work investigated the use of distillation for building task-specific models, we leverage knowledge distillation during the pre-training phase and show that it is possible to reduce the size of a BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster. To leverage the inductive biases learned by larger models during pre-training, we introduce a triple loss combining language modeling, distillation and cosine-distance losses. Our smaller, faster and lighter model is cheaper to pre-train and we demonstrate its capabilities for on-device computations in a proof-of-concept experiment and a comparative on-device study.

02 Model

- Temperature Softmax

ü 기존 Softmax 와 Cross-Entropy 를 사용해 label의 확률은 1에 가깝게, 정답이 아닌 label은 0에 가까운 값(near-zero)으로 만든다.

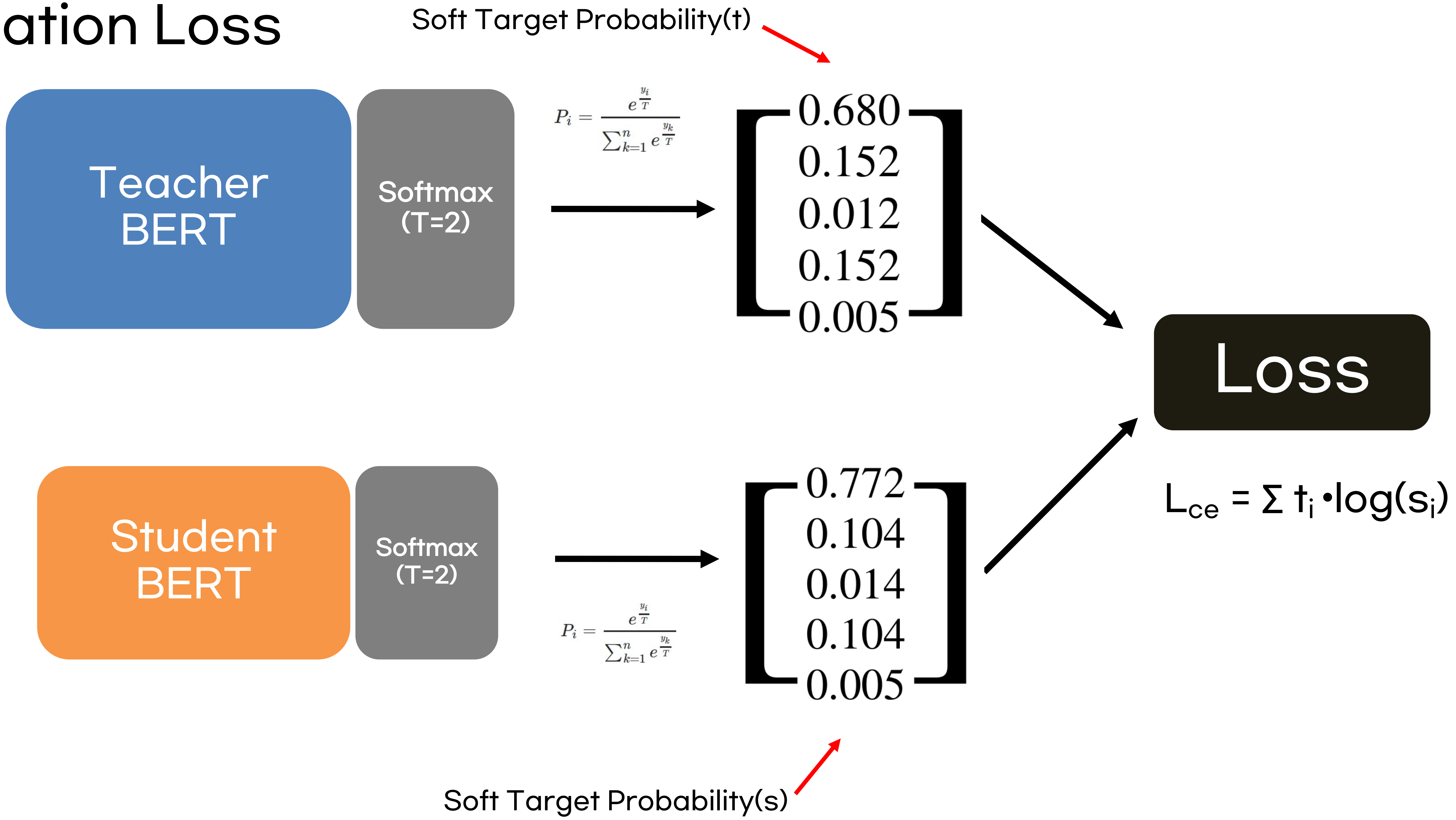
ü 하지만 정답이 아닌 Label 에서도 배울 지식이 있다면?
ex) 바둑이 -> 스포츠(정답),강아지(오답),스파게티(오답)

✓ 이러한 지식들을 배우기 위해 Temperature Softmax 를 사용

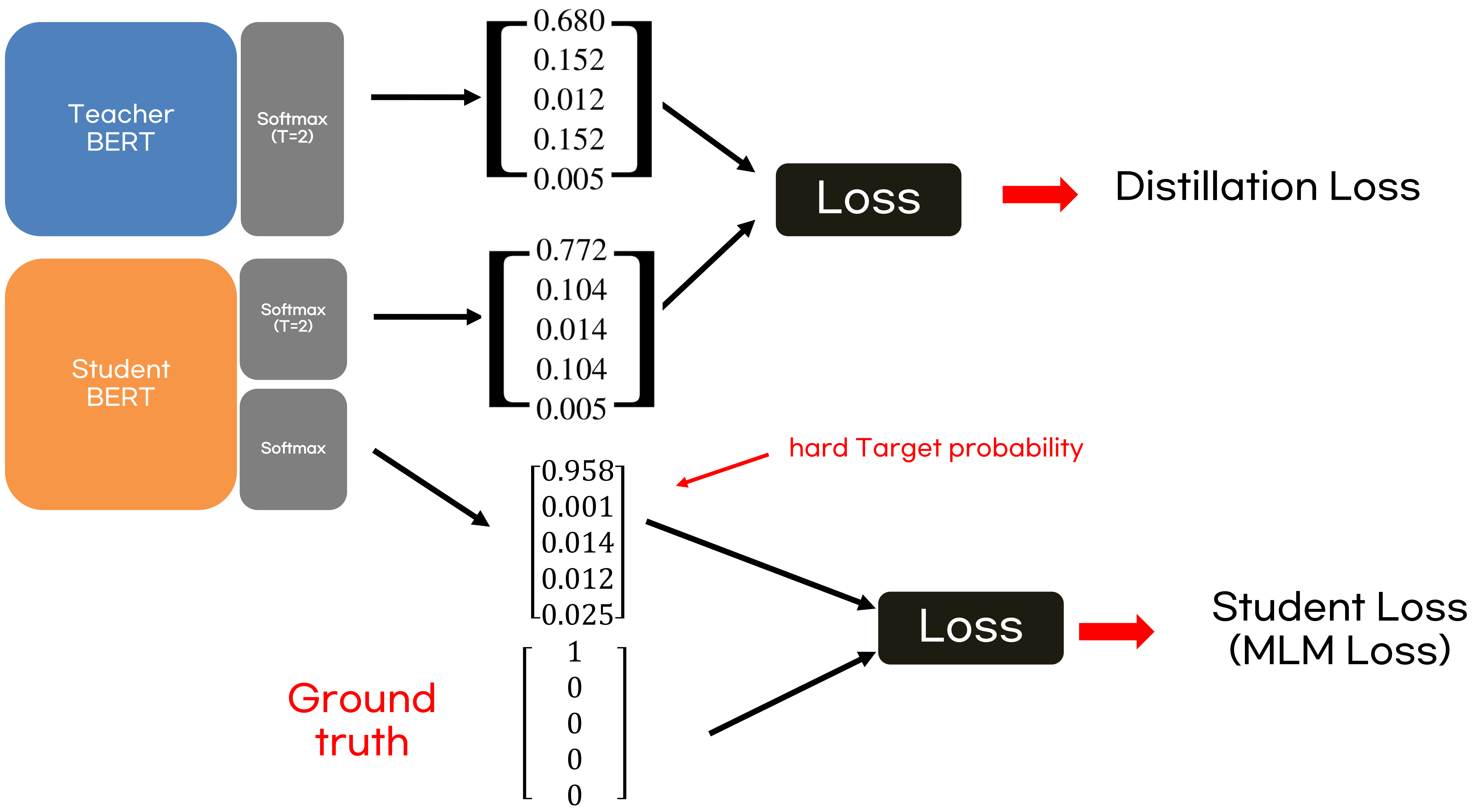
$$P_i = \frac{e^{\frac{y_i}{T}}}{\sum_{k=1}^{n} e^{\frac{y_k}{T}}}$$

02 DistilBERT

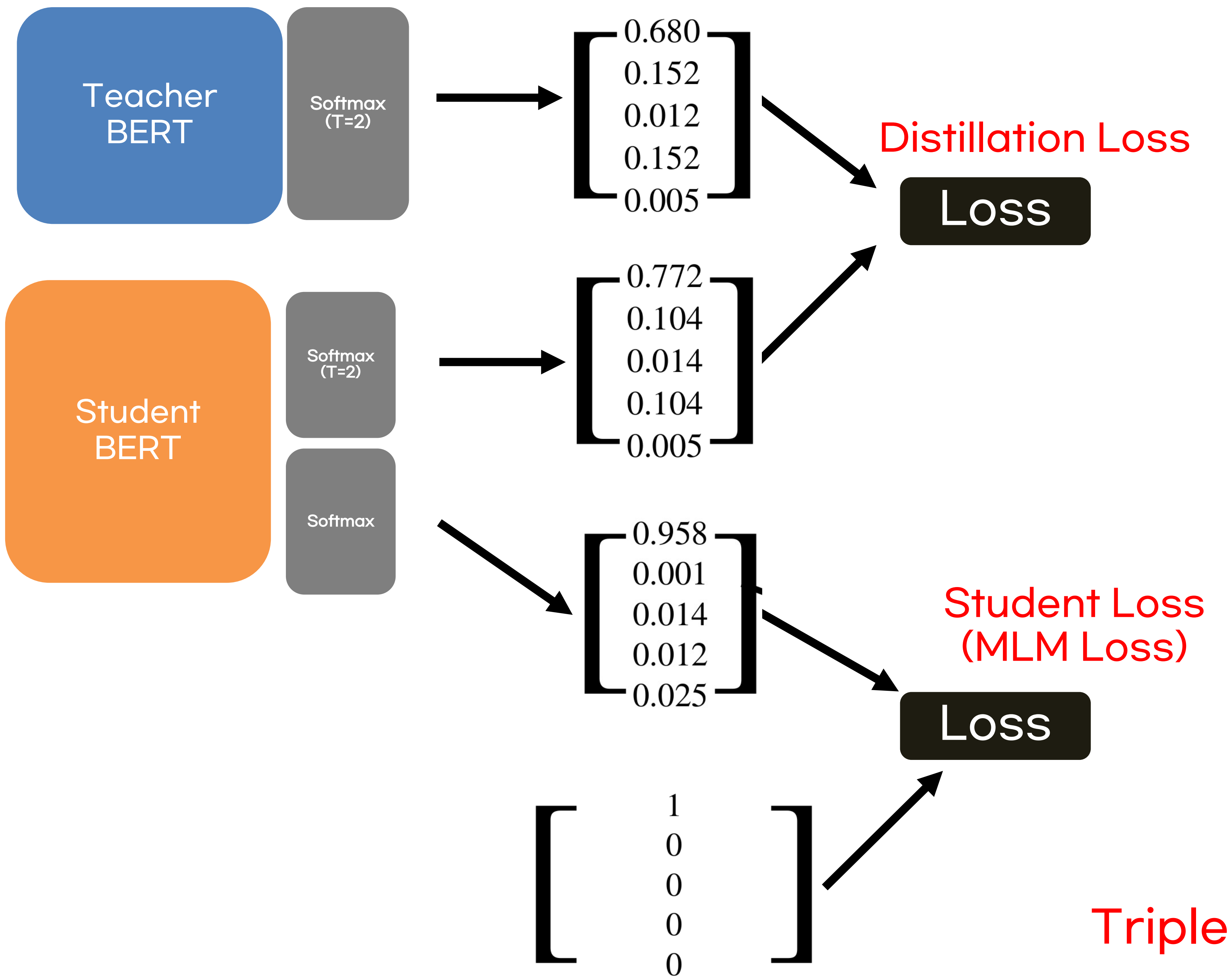
- Distillation Loss

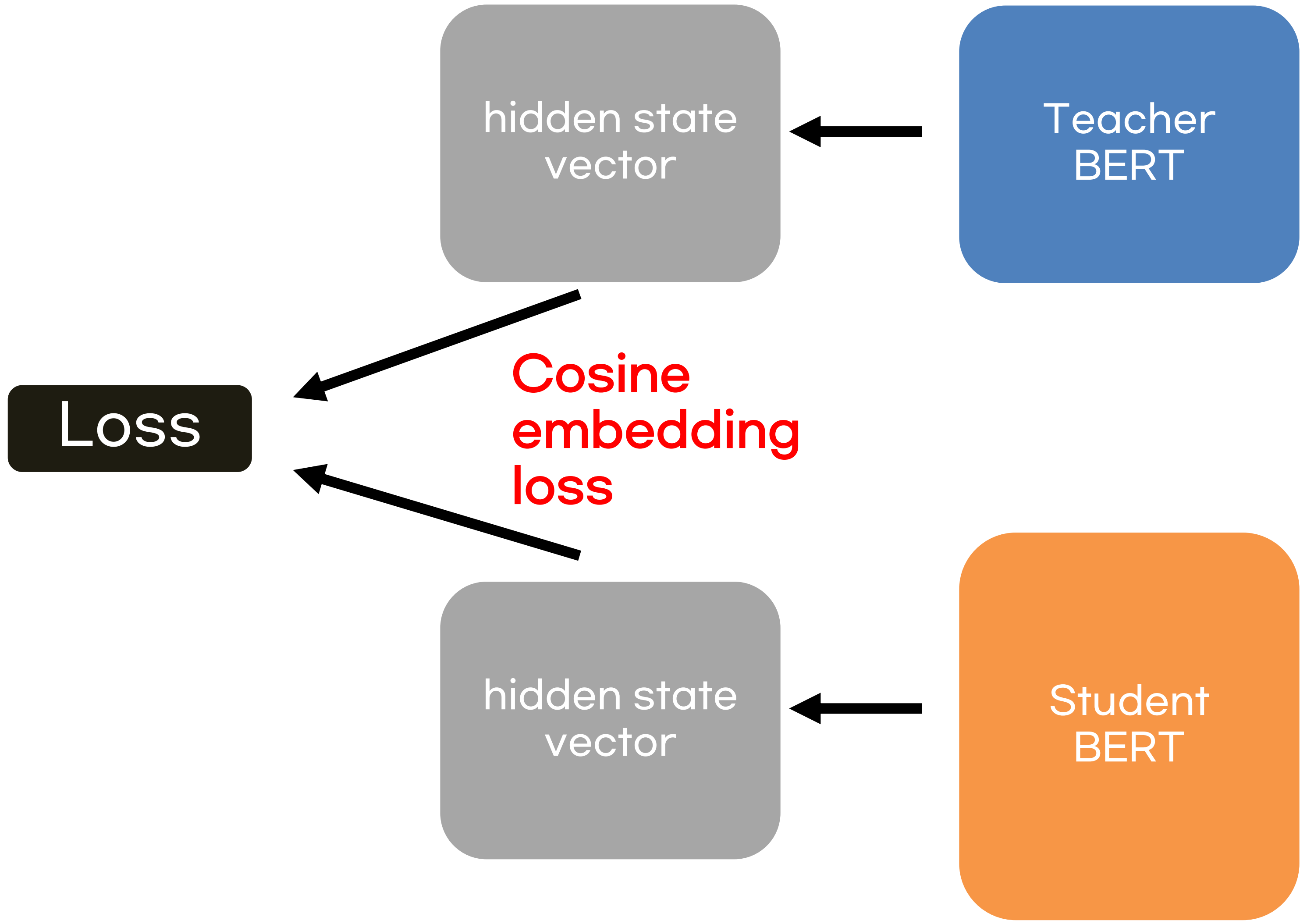


02 DistilBERT



02 DistilBERT





$$\text{Triple Loss} = \alpha * L_{\text{Distillation}\neq} + \beta * L_{\text{student}} + \gamma * L_{\text{cosine}}$$
$$(\alpha+\beta+\gamma=1)$$

03 Tokenizer

```
from transformers import AutoTokenizer

 model = “distilbert-base-uncased”
 tokenizer = AutoTokenizer.from_pretrained(model_ckpt)
```

```
tokenizer.vocab_size # 단어 사전의 개수
>> 30522

tokenizer.model_max_length # 모델 최대 문맥 길이
>> 512

tokenizer.model_input_names # 모델이 기대하는 입력값
>> [ ‘input_ids’ , ‘attention_mask’]
```

03 Tokenizer

```
tokens = tokenizer.convert_ids_to_tokens(encoded_text.input_ids) # Token ID -> Token
print(tokens)

>>> [ ‘ [CLS] ', ‘ token ', ‘ ##izing ', ‘ text ', ‘ is ', ‘ a ', ‘ core ', ‘ task ’, ‘ of ', ‘ nl ', ‘ ##p ', ‘ . ', ‘ [SEP] ’ ]
```

사전학습된 토크나이저에 따라 다를 수 있음

[UNK] - 100 어휘 사전에 없는 단어

[CLS] - 101 문장의 시작

[SEP] - 102 문장의 구별

[MASK] - 103 단어 마스크

03 Tokenize

```
def tokenize(batch):
	return tokenizer(batch['text'], padding=True,truncation=True)
```

Padding : 배치 내 가장 긴 샘플에 맞춰 0으로 패딩

Truncation : 최대 문맥 크기에 맞춰 샘플을 자름
(최대 문맥 크기 → tokenizer.model_max_length)

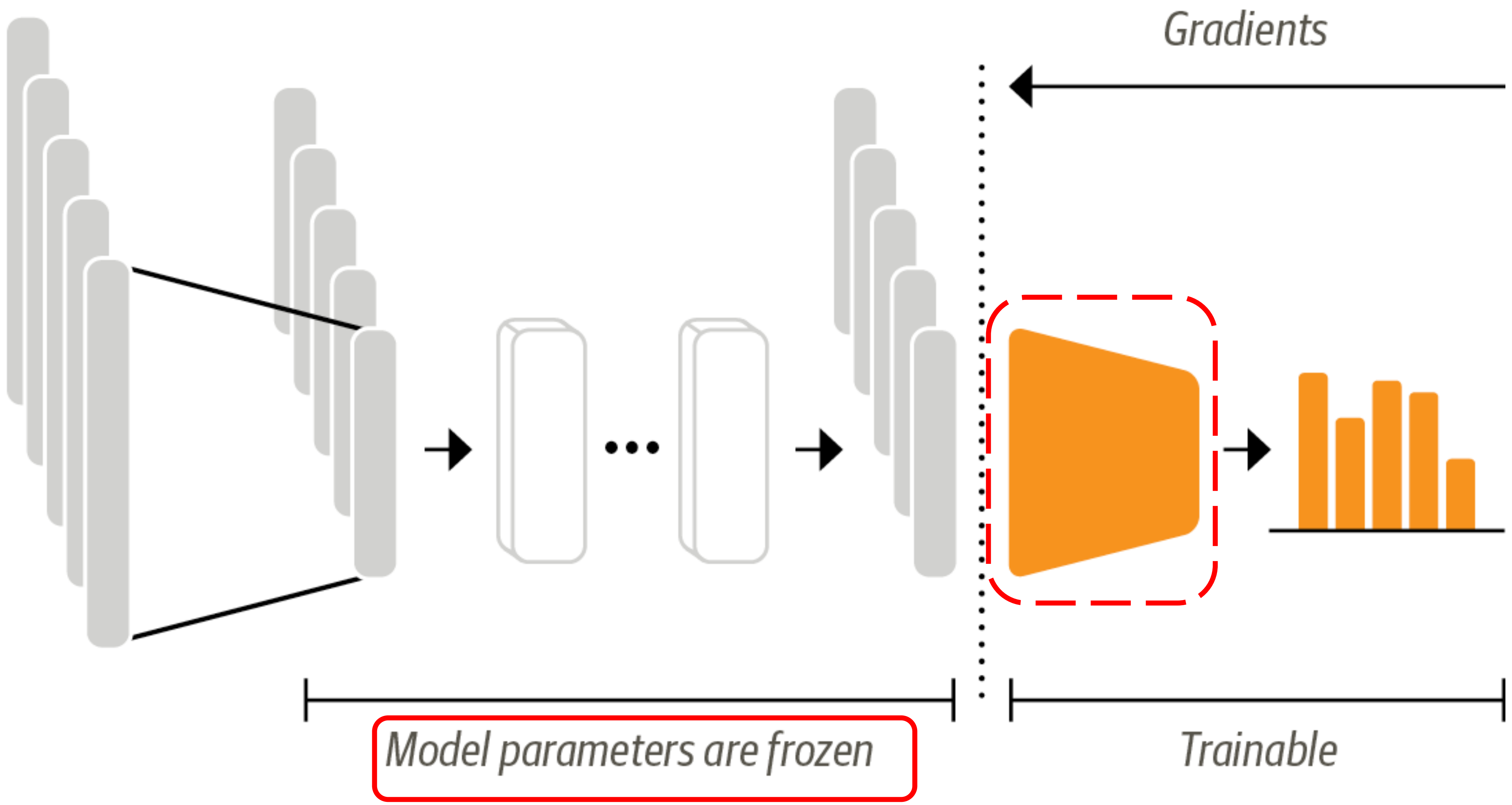
```
print(tokenize(emotions["train"][:2]))

>>> {
	'input_ids': [
	[101, 1045, 2134, 2102, 2514, 26608, 102, 0, 0, 0, 0, 0],
	[101, 10047, 9775, 1037, 3371, 2000, 2695, 1045, 2514, 20505, 3308, 102]
			] ,

	'attention_mask': [
	[1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0],
	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
			]
   }
```

Input_ids : encoding ids
(Token ID)

attention_mask : 패딩 토큰 구분
(실제 정보를 담고있는 토큰을 알려줌)

04 특성 정보 추출

- 특성 추출기로 사용하기

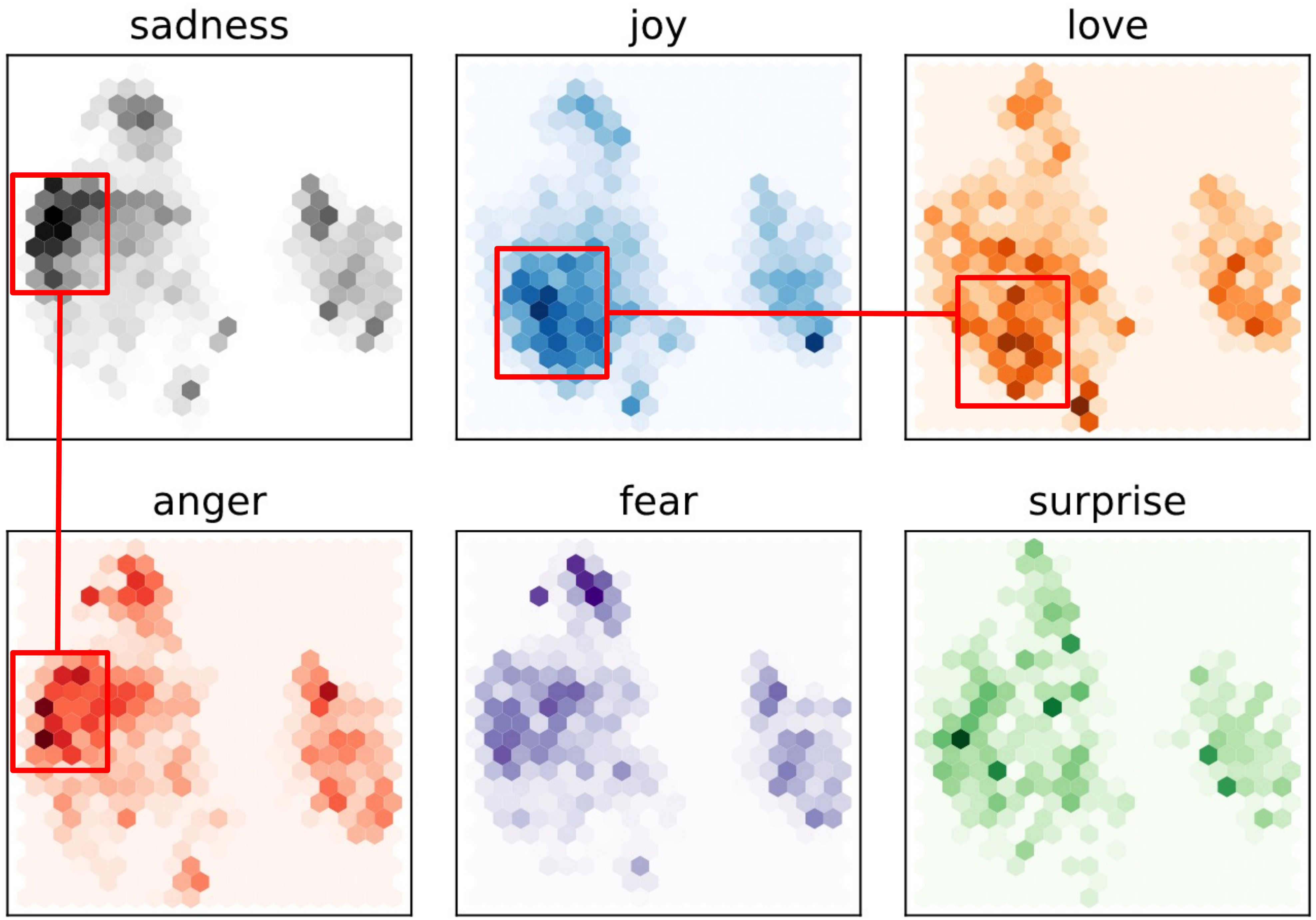


04 특성 정보 추출

✓ [CLS] : 분류 작업에서 전체 시퀀스의 정보가 담겨있음

```
def extract_hidden_states(batch):
    # 모델 입력을 GPU로 옮깁니다. Device 설정                    [ "input_ids", "attention_mask" ]
    inputs = { k : v.to(device) for k,v in batch.items() if k in tokenizer.model_input_names }
    # 마지막 은닉 상태를 추출합니다.
    with torch.no_grad():
        last_hidden_state = model(**inputs).last_hidden_state
    # [CLS] 토큰에 대한 벡터를 반환합니다.
    return {"hidden_state": last_hidden_state[:,0].cpu().numpy()}
    # map 함수를 적용하기 위해서는 파이썬이나 Numpy 객체 필요

emotions_hidden = emotion_encoded.map(extract_hidden_state,batched=True)
```

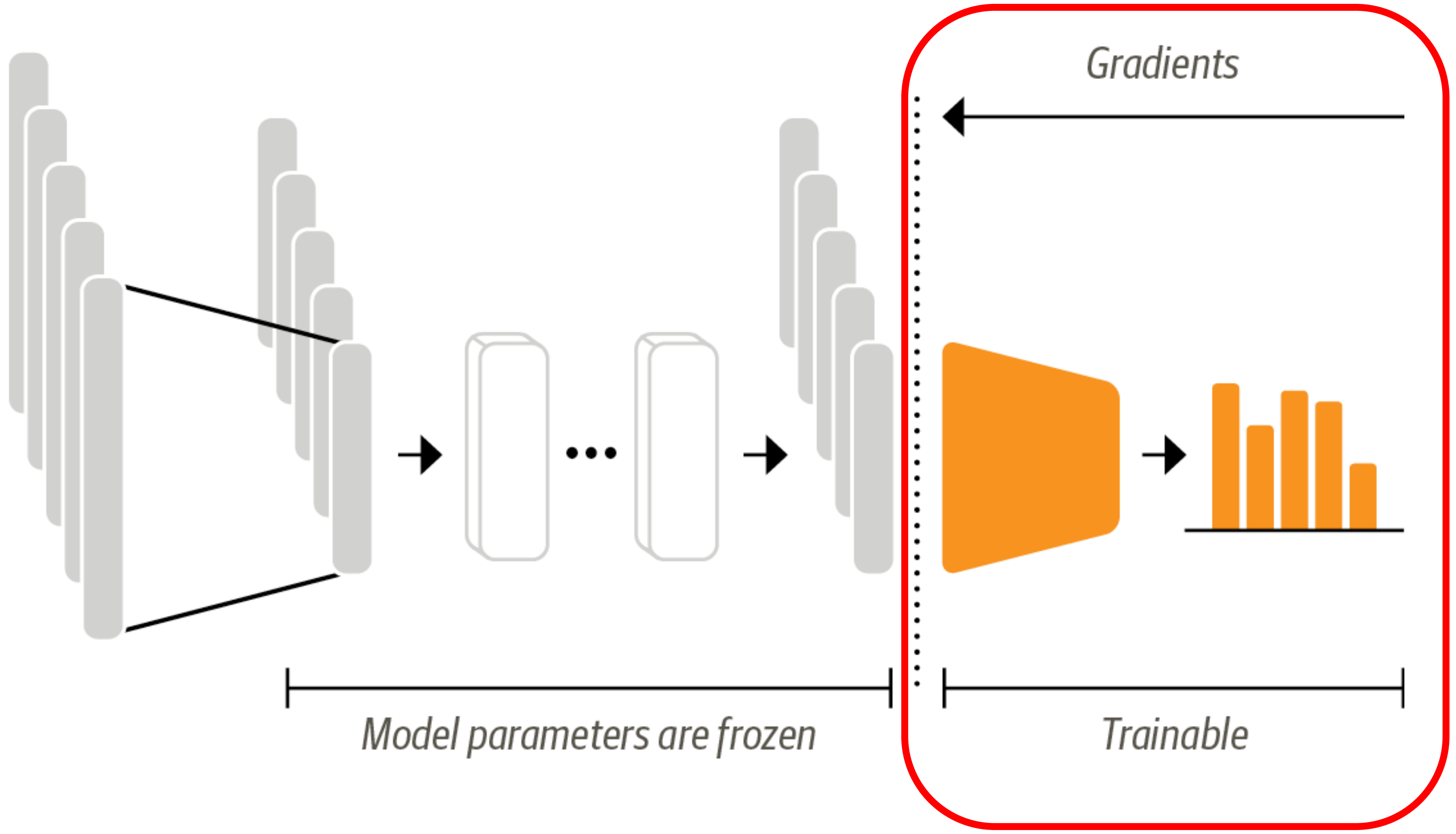
04 특성 정보 추출



04 특성 정보 추출

✓ 특성 정보를 활용한 기계학습(Logistic Regression)

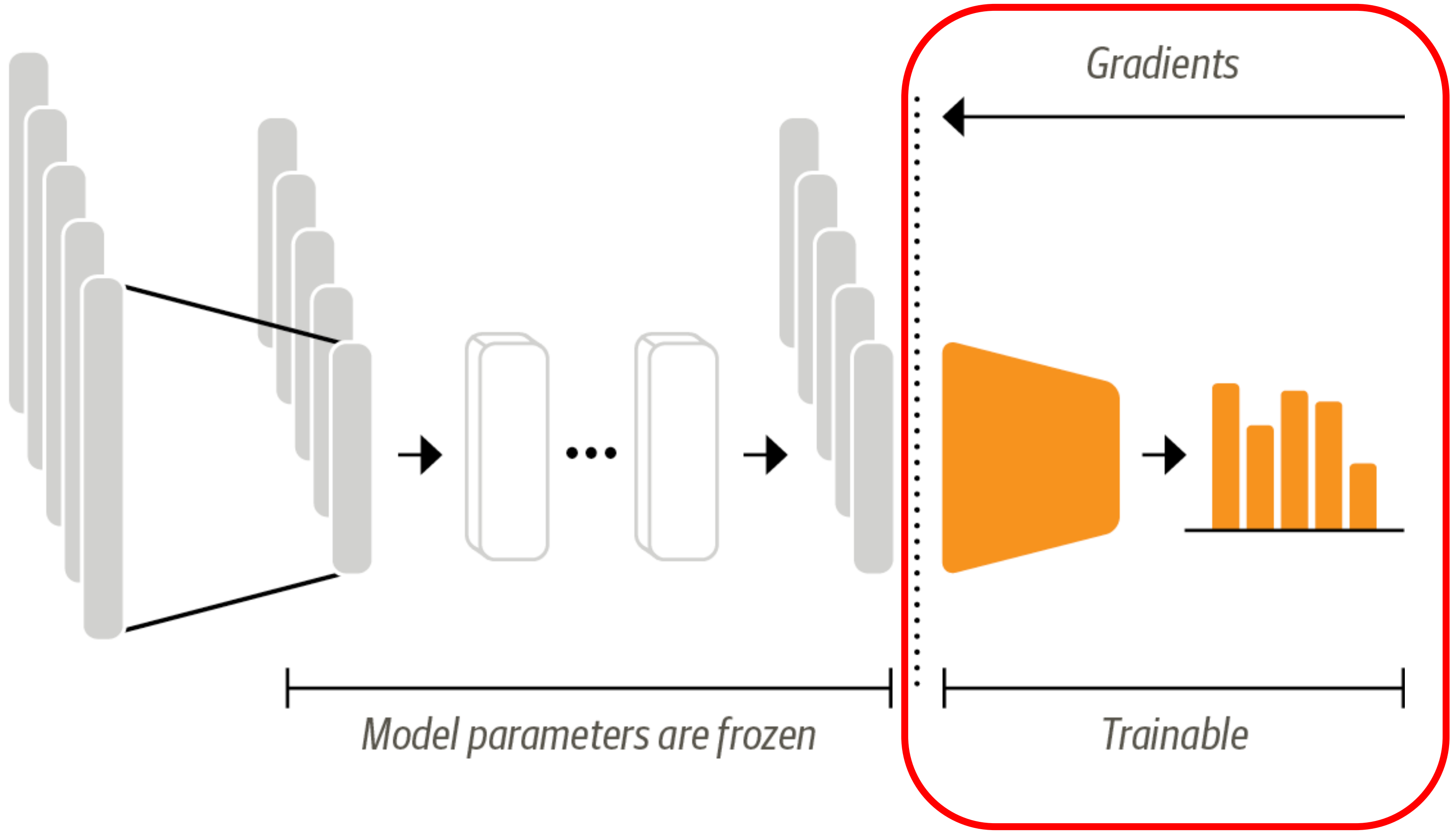
```
from sklearn.linear_model import LogisticRegression

lr_clf = LogisticRegression(max_iter=3000)
lr_clf.fit(X_train, y_train) # X_train -> R[CLS]
lr_clf.score(X_valid, y_valid)

>>> 0.633
```



04 특성 정보 추출

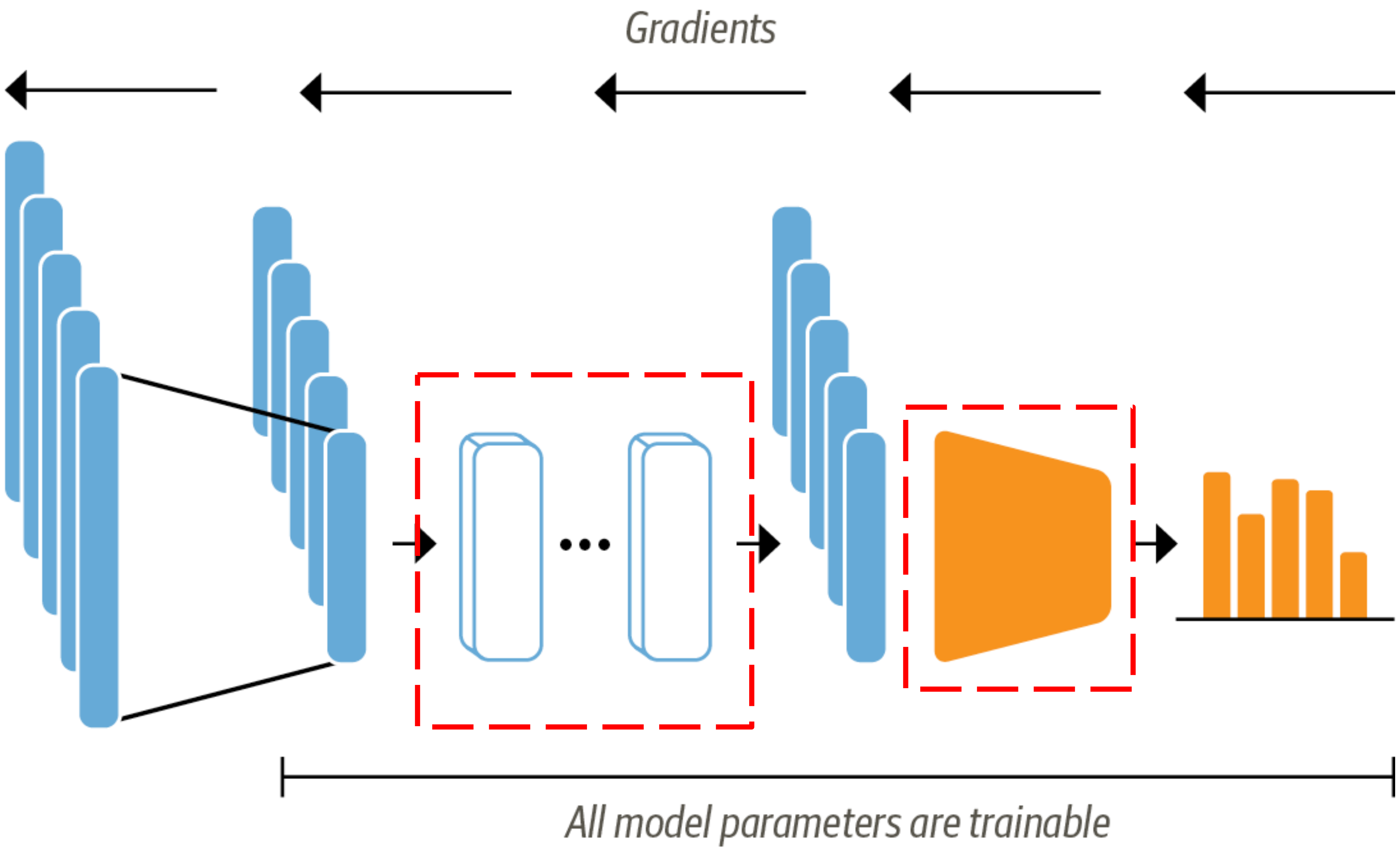
✓ 특성 정보를 활용한 기계학습(Logistic Regression)

```
from sklearn.linear_model import LogisticRegression
from imblearn.over_sampling import SMOTE
smote = SMOTE(strategy='minority')
X_resampled ,y_resampled = smote.fit_resample(X_train,y_train)

lr_clf = LogisticRegression(max_iter=3000)
lr_clf.fit(X_resampled, y_resampled)
lr_clf.score(X_valid, y_valid)

>>> 0.621
```



05 fine-tuning

```
from transformers import AutoForSequenceClassification

num_labels = 6
model =AutoForSequenceClassification.from_pretrained(model_ckpt,num_labels=num_labels).to(device)
```



05 fine-tuning

```
from transformers import Trainer, TrainingArguments

batch_size = 64
model_name = f"{model_ckpt}-finetuned-emotion"
training_args = TrainingArguments(      output_dir=model_name,
                                        num_train_epochs=2,
                                        learning_rate=2e-5,
                                        per_device_train_batch_size=batch_size,
                                        per_device_eval_batch_size=batch_size,
                                        weight_decay=0.01,
                                        evaluation_strategy="epoch",
                                        disable_tqdm=False,
                                        logging_steps=200,
                                        push_to_hub=True,
                                        save_strategy="epoch",
                                        load_best_model_at_end=True,
                                        log_level="error")
```

TrainingArguments : 학습에 사용할 파라미터들 설정

05 fine-tuning

```
from transformers import Trainer, TrainingArguments

trainer = Trainer(
            model = model,
            args = training_args,
            compute_metrics = compute_metrics
            train_dataset = emotion_encoded['train'],
            eval_dataset = emotion_encoded['validation']
                  )

trainer.train()
```

```
<compute_metrics>
from sklearn.metrics import f1_score,accuracy

def compute_metrics(preds):
        labels = preds.input_ids
        pred = preds.predictions.argmax(dim=-1)
        f1 = f1_score(pred,labels,averaged='weighted')
        acc = accuracy(pred,labels)
```

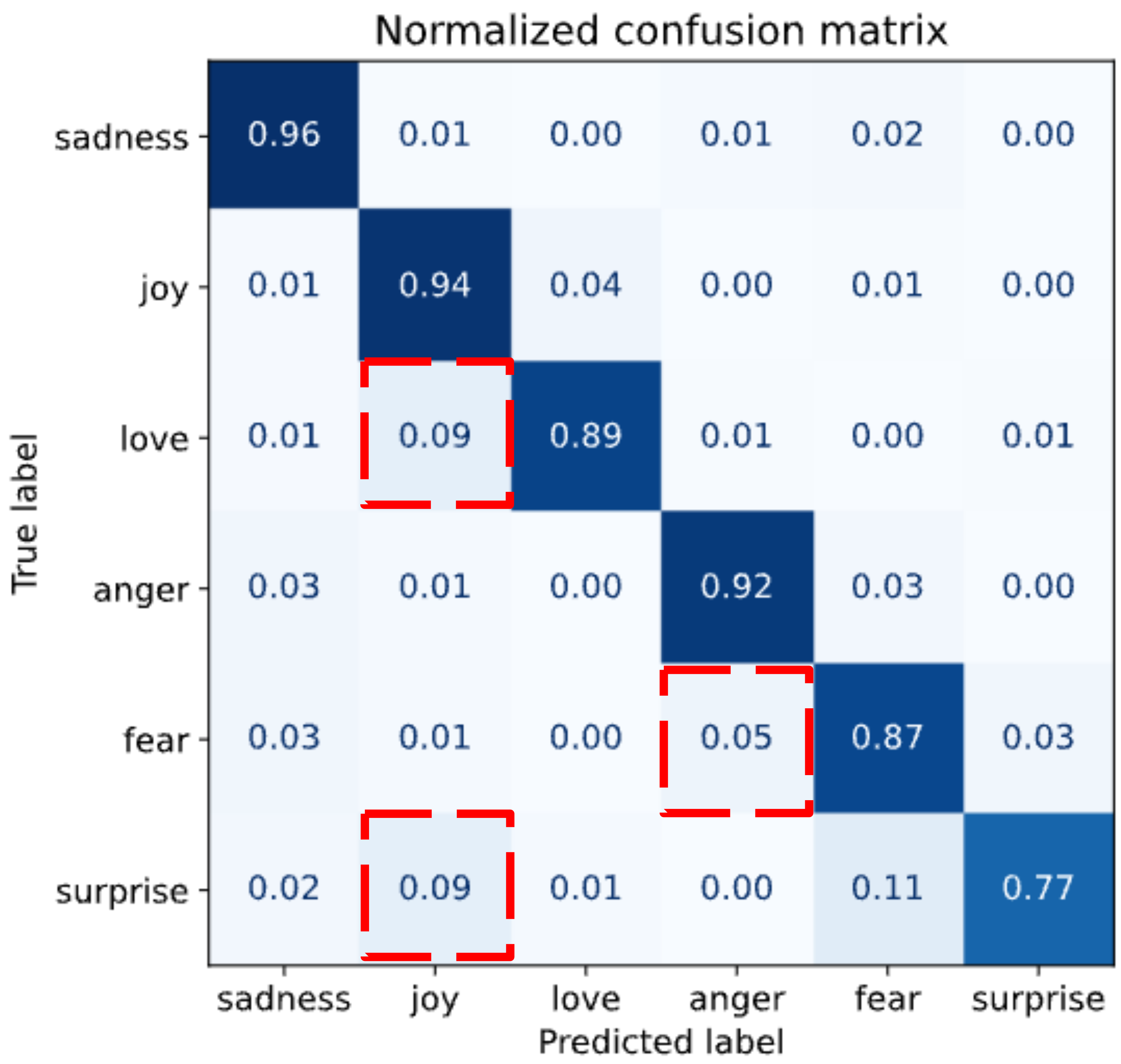
Epoch	Training Loss	Validation Loss	Accuracy	F1
1	0.871400	0.338466	0.904000	0.901318
2	0.261400	0.226248	0.926000	0.925926

06 결과 분석

✓ love : joy 혼동 0.09

✓ surprise : joy 혼동 0.09

✓ anger : fear 혼동 0.09



06 결과 분석

✓ Labeling 오류가 많았음

- text는 sadness에 가까우나 label은 joy

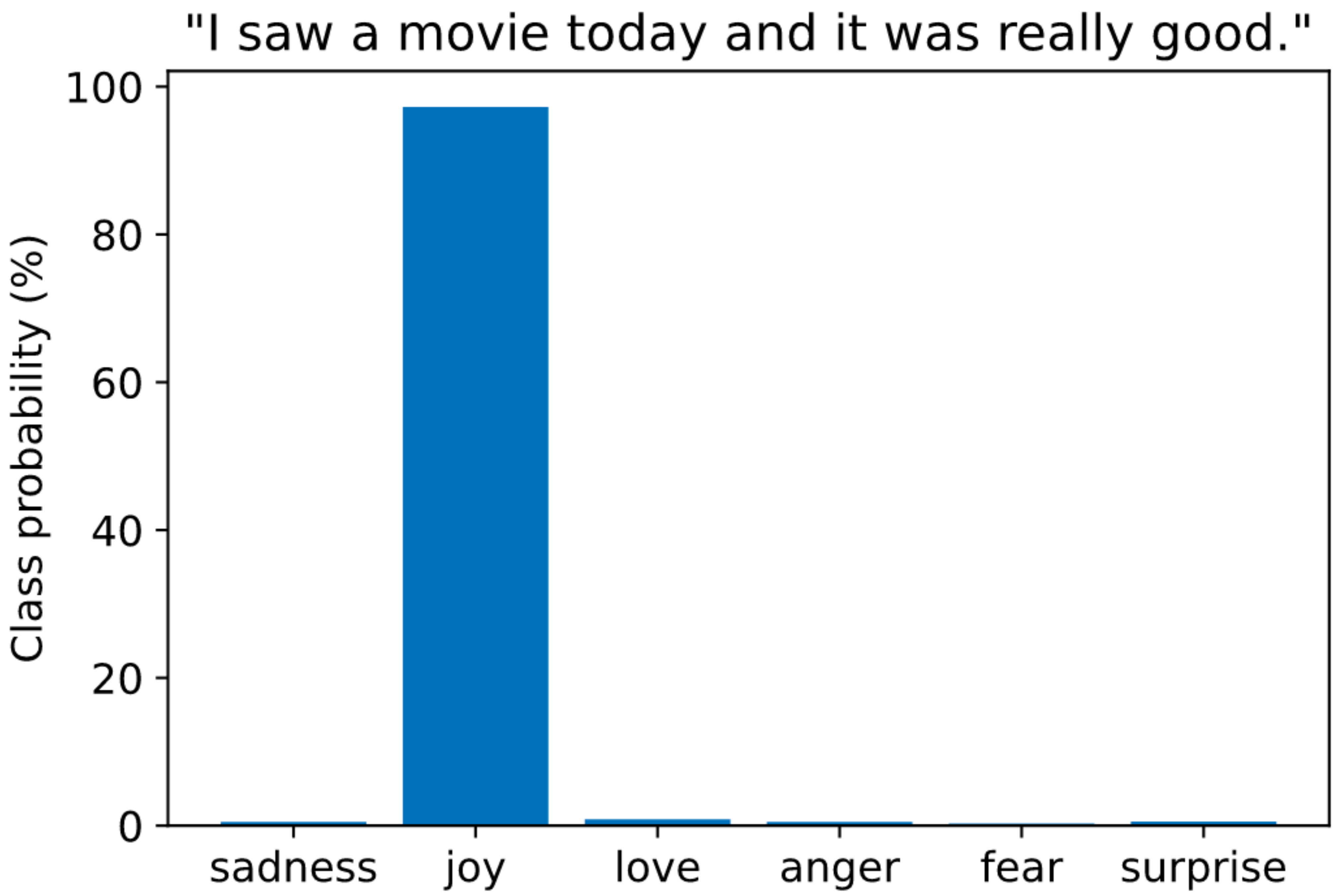
	text	label	predicted_label	loss
1950	i as representative of everything thats wrong ...	surprise	sadness	5.521721
1801	i feel that he was being overshadowed by the s...	love	sadness	5.442623
882	i feel badly about reneging on my commitment t...	love	sadness	5.134108
1963	i called myself pro life and voted for perry w...	joy	sadness	5.073962
1274	i am going to several holiday parties and i ca...	joy	sadness	5.059187
765	i feel super awkward and out of place right now	joy	sadness	5.048173
1509	i guess this is a memoir so it feels like that...	joy	fear	4.929876
465	i would eventually go in to these stores but i...	joy	fear	4.862169
1870	i guess i feel betrayed because i admired him ...	joy	sadness	4.725090
1111	im lazy my characters fall into categories of ...	joy	fear	4.653110

06 결과 분석

✓ sadness에 대한 예측이 가장 강함

	text	label	predicted_label	loss
69	i have no extra money im worried all of the ti...	sadness	sadness	0.020412
1601	i feel so ungrateful when thinking saying thes...	sadness	sadness	0.020654
697	i was missing him desperately and feeling idio...	sadness	sadness	0.020685
1466	i feel so ungrateful to be wishing this pregna...	sadness	sadness	0.020722
1310	i feel like an ungrateful asshole	sadness	sadness	0.020734
1502	i feel ungrateful for stupid shit like	sadness	sadness	0.020954
21	i feel try to tell me im ungrateful tell me im...	sadness	sadness	0.021010
133	i and feel quite ungrateful for it but i m loo...	sadness	sadness	0.021021
1663	i feel idiotic calling again though	sadness	sadness	0.021083
1767	i feel jaded about everything	sadness	sadness	0.021092

06 결과 분석


"I saw a movie today and it was really good."
Class probability (%)
100
80
60
40
20
0
sadness
joy
love
anger
fear
surprise

감사합니다.