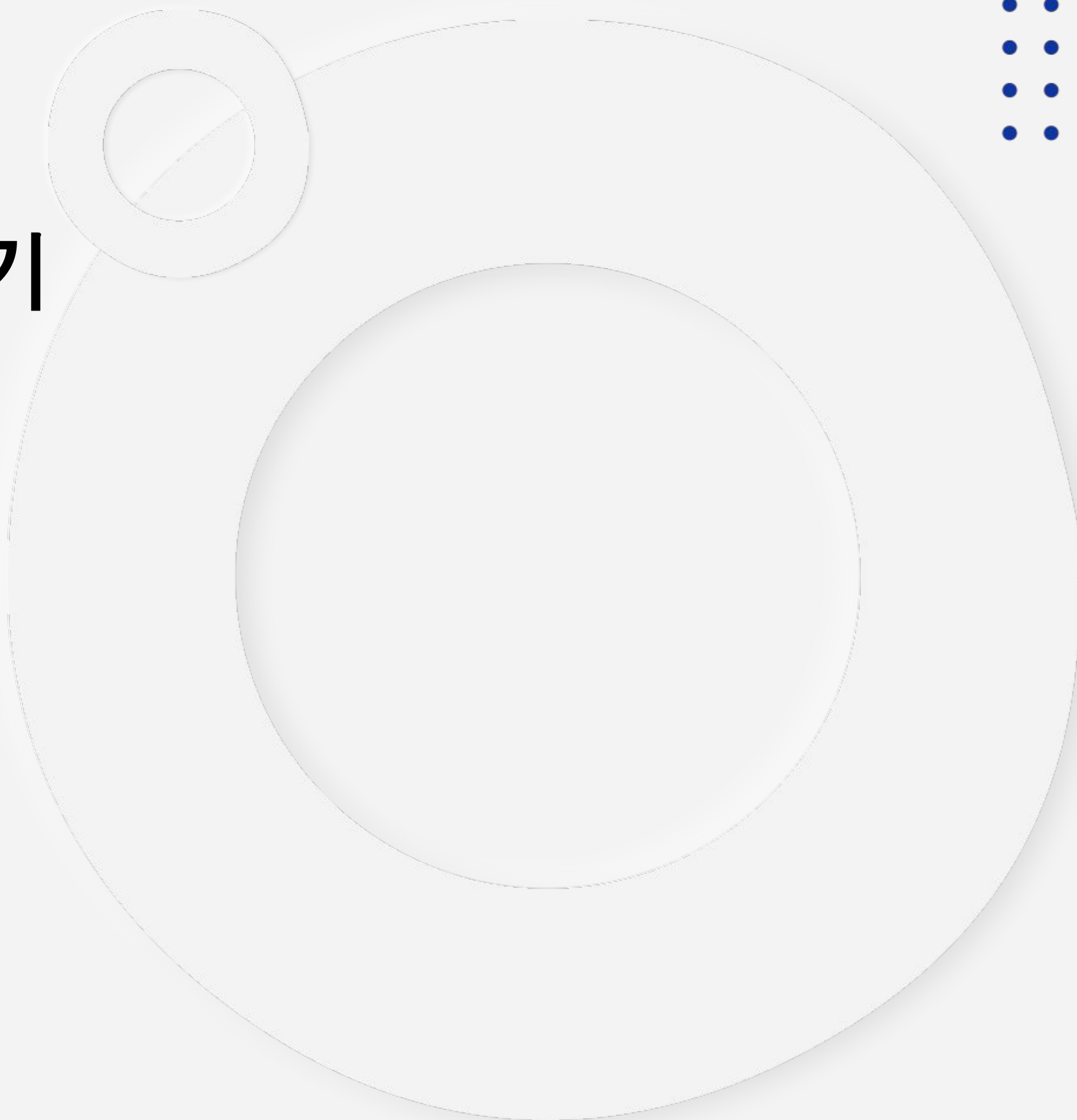


Chapter 9

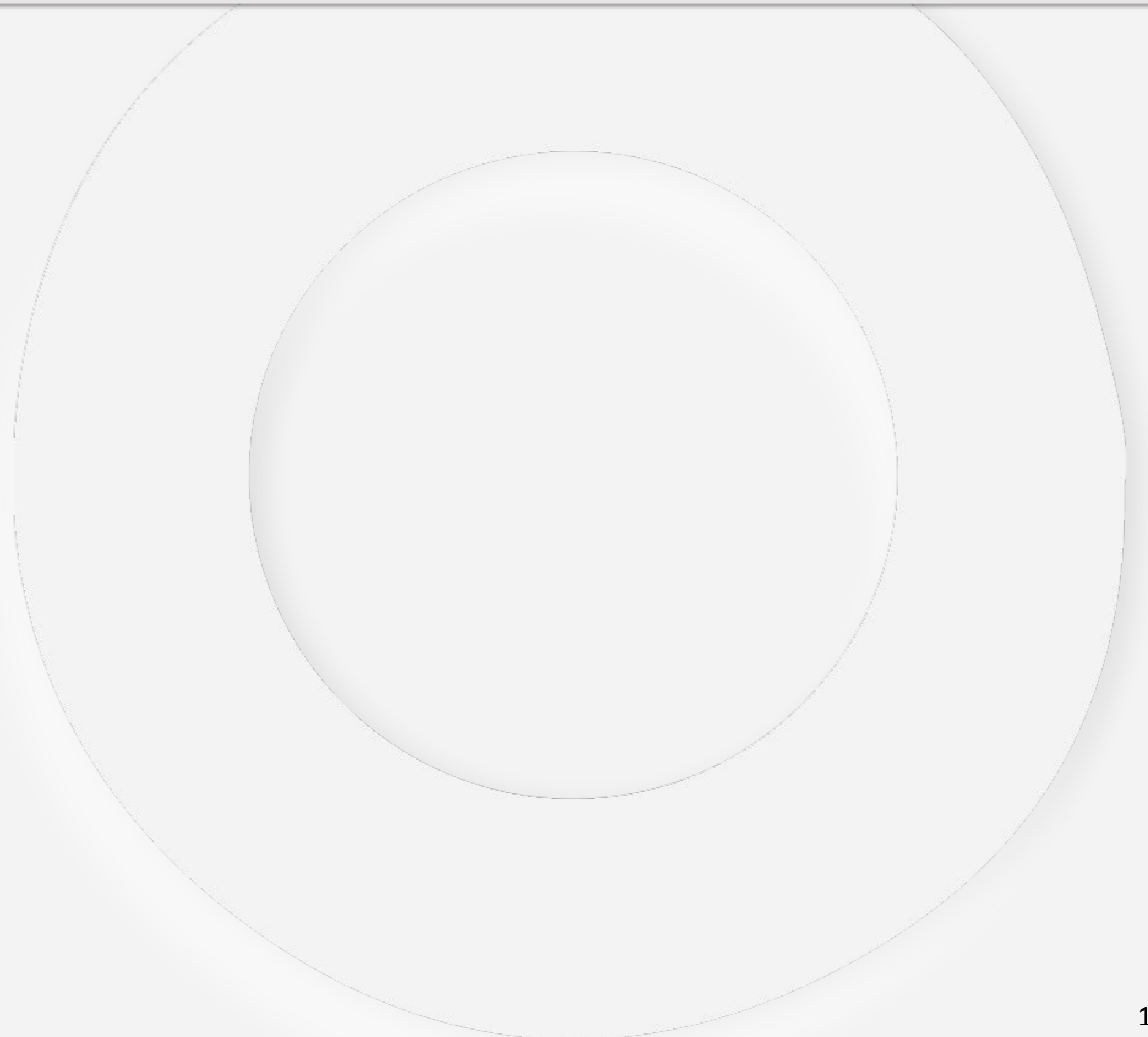
레이블 부족 문제 다루기

2024.01.30 이상민




목차

1. 깃허브 이슈태거
2. 레이블링된 데이터가 없는 경우
3. 레이블링된 데이터가 적은 경우
4. 레이블링되지 않은 데이터 활용



1. 깃허브 이슈 태거

 [huggingface](#) / [transformers](#)

Watch 745

Star 45.5k

Fork 10.8k

[Code](#) [Issues 361](#) [Pull requests 74](#) [Actions](#) [Projects 23](#) [Wiki](#) [Security](#) [Insights](#)


[2D Parallelism] Tracking feasibility #9931

Title

New issue

Open

stas00 opened this issue on 1 Feb · 2 comments

 stas00 commented on 1 Feb · edited · Description

Collaborator

Background

ZeRO-DP (ZeRO Data Parallel) and PP (Pipeline Parallelism) provide each a great memory saving over multiple GPUs. Each 1D allows for a much more efficient utilization of the gpu memory, but it's still not enough for very big models - sometimes not even feasible with any of the existing hardware. e.g. a model that's 45GB-big with just model params (t5-11b) can't fit even on a 40GB GPU.

The next stage in Model Parallelism that can enable loading bigger models onto smaller hardware is 2D Parallelism. That's combining Pipeline Parallelism (PP) with ZeRO-DP.

3D Parallelism is possible too and it requires adding a horizontal MP (ala [Megatron-LM](#), but we don't quite have any way to implement that yet. Need to study Megatron-LM first. So starting with a relatively low hanging fruit of 2D.

Tracking

We have 3 implementations that provide the required components to build 2D Parallelism:

1. DeepSpeed (DS)
2. FairScale (FS)
3. PyTorch (native) (PT)

and the purpose of this issue is to track the feasibility/status/inter-operability in each one of them. And also which parts have been back-ported to PyTorch core.

Plus it tracks the status of where transformers models are at with regards to the above 3 implementations.

The 2 main questions are:

1. native 2D: how do we integrate a native PP with native ZeRO-DP (sharded) (e.g. can fairscale PP work with fairscale ZeRO-DP)
2. inter-operability 2D: is there a chance one implementation of PP/ZeRO-DP could work with one or both others ZeRO-DP/PP (e.g. can fairscale PP work with DeepSpeed ZeRO-DP).

Assignees

stas00

Tags

Labels

DeepSpeed Model Parallel Pipeline Parallel WIP

Projects

None yet

Milestone

No milestone

Linked pull requests

Successfully merging a pull request may close this issue.

None yet



Notifications

Customize

Subscribe

You're not receiving notifications from this thread.

2 participants

1. 깃허브 이슈 태거

1.1 데이터 준비

데이터 다운로드

```
import pandas as pd

dataset_url = "https://git.io/nlp-with-transformers"
df_issues = pd.read_json(dataset_url, lines=True)
print(f"데이터프레임 크기: {df_issues.shape}")
```

데이터프레임 크기: (9930, 26)

불필요한 데이터 제거

```
df_issues["labels"] = (df_issues["labels"]
                       .apply(lambda x: [meta["name"] for meta in x]))
```

1. 깃허브 이슈 태거

1.2 EDA

이슈별 태그 개수 확인

```
df_issues["labels"].apply(lambda x: len(x)).value_counts().to_frame().T
```

index	0	1	2	3	4	5
labels	6440	3057	305	100	25	3

전체 태그 개수 확인

```
df_counts = df_issues["labels"].explode().value_counts().to_frame()
print(f"레이블 개수: {len(df_counts)}")
df_counts.head(8).T
```

레이블 개수: 65

index	wontfix	model card	Core: Tokenization	New model	Core: Modeling	Help wanted	Good First Issue	Usage
labels	2284	649	106	98	64	52	50	46

1. 깃허브 이슈 태거

1.3 데이터 필터링

- 필터링 코드

```
label_map = {"Core: Tokenization": "tokenization",
             "New model": "new model",
             "Core: Modeling": "model training",
             "Usage": "usage",
             "Core: Pipeline": "pipeline",
             "TensorFlow": "tensorflow or tf",
             "PyTorch": "pytorch",
             "Examples": "examples",
             "Documentation": "documentation"}

def filter_labels(x):
    return [label_map[label] for label in x if label in label_map]

df_issues["labels"] = df_issues["labels"].apply(filter_labels)
all_labels = list(label_map.values())
```

index	tokenization	new model	model training	usage	pipeline	tensorflow or tf	pytorch	documentation	examples
labels	106	98	64	46	42	41	37	28	24

- 필터링 결과

split	
unlabeled	9489
labeled	441

2. 레이블링된 데이터가 없는 경우

2.1 zero-shot classification

사전학습 된 모델이 이전에 볼 수 없었던 클래스에서 분류할 수 있는 작업

2.2 자연어 추론

전제조건과 가설이 주어질 때 조건에 대해 가설이 일치하는지를 분류하는 작업

전제	가설	레이블
His favourite color is blue	He is into heavy metal music	neutral
She finds the joke hilarious	She thinks the joke is not funny at all	contradiction
The house was recently	The house is new	entailment

2.3 적용방법

이슈의 내용과 제목을 전제로 아래와 같은 예시를 가설로 정의해 모델에 입력

"This example is about {label}"

2. 레이블링된 데이터가 없는 경우

2.4 코드 구현

추론할때 다중 레이블 분류를 위해 `multi_label = True`로 설정

```
#모델 불러오기
from transformers import pipeline
pipe = pipeline("zero-shot-classification", device = 0)

#추론
sample = ds["train"][0]
print(f"레이블: {sample['labels']}")
output = pipe(sample["text"], all_labels, multi_label = True)#다중레이블 분류를 위해 multi_label=True
print(output["sequence"][:400])
print("\n 예측:")

#결과 출력
for label, score in zip(output["labels"],output["scores"]):
    print(f"{label}, {score:.2f}")
```

2.5 출력 결과

```
레이블: ['new model']
Add new CANINE model

# 🌟 New model addition

## Model description

Google recently proposed a new **C**haracter **A**rchitecture with **N**o
tokenization **I**n **N**eural **E**ncoders architecture (CANINE). Not only the
title is exciting:

> Pipelined NLP systems have largely been superseded by end-to-end neural
modeling, yet nearly all commonly-used models still require an explicit tokeni

예측:
new model, 0.98
tensorflow or tf, 0.37
examples, 0.34
usage, 0.30
pytorch, 0.25
documentation, 0.25
model training, 0.24
tokenization, 0.17
pipeline, 0.16
```

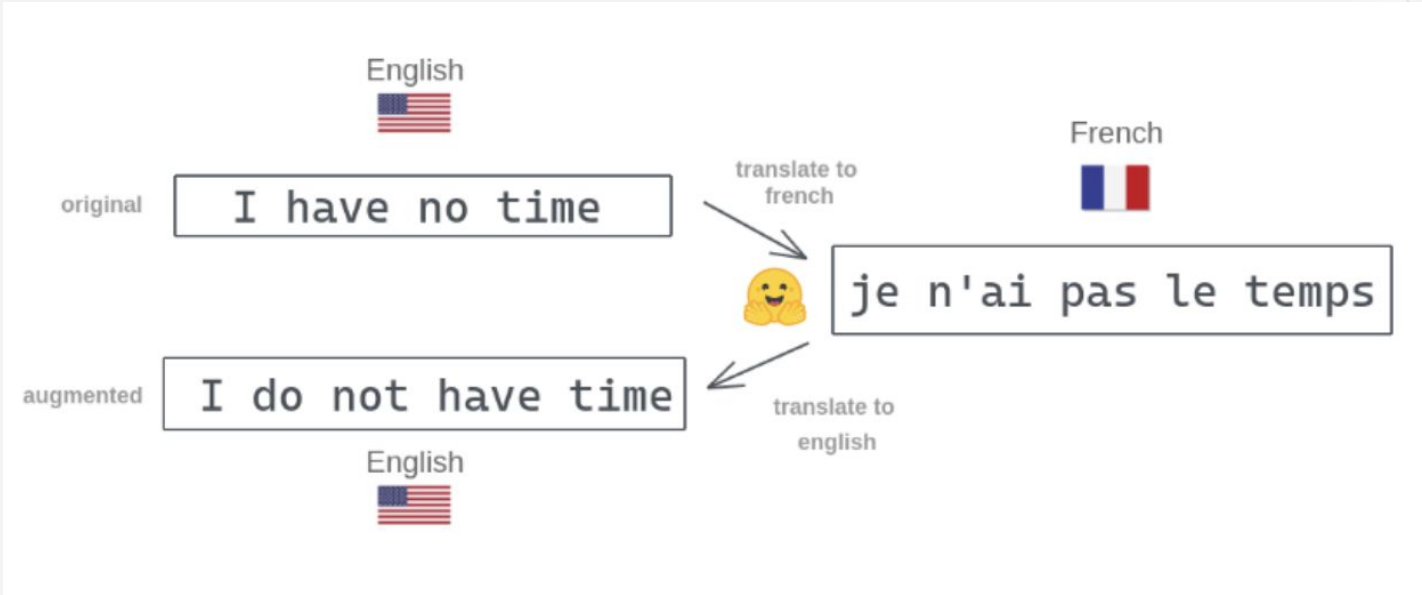

3. 레이블링된 데이터가 적은 경우

3.1 Data Augmentation

데이터 증식(Data augmentation)은 갖고 있는 데이터 셋을 여러가지 방법으로 늘리는 기법

- 역 번역

원본 언어로 된 텍스트를 기계 번역을 사용해 다른 언어로 번역한다. 그 다음 번역된 언어를 다시 원본 언어로 번역



- 토큰 섞기

훈련 세트의 한 텍스트에서 동의어 교체, 단어추가, 교환, 삭제 같은 간단한 변환을 임의로 선택해 수행

3. 레이블링된 데이터가 적은 경우

3.2 Data Augmentation 코드

NlpAug의 wordembedding을 이용한 동의어 교체 코드

```
from transformers import set_seed
import nlpaug.augmenter.word as naw

set_seed(3)
aug = naw.ContextualWordEmbsAug(model_path="distilbert-base-uncased",
                                device="cpu", action="substitute")

text = "Transformers are the most popular toys"
print(f"원본 텍스트: {text}")
print(f"증식된 텍스트: {aug.augment(text)}")
```

3.3 Data Augmentation 결과

원본 텍스트: Transformers are the most popular toys
증식된 텍스트: ['transformers – the most coveted toys']

4. 레이블링되지 않은 데이터 활용

4.1 도메인 적응

하나의 도메인의 데이터에 대해 훈련된 모델을 다른 관련 도메인의 데이터에 대해 잘 수행하기 위해 적응시키는 과정

- 토큰화

```
def tokenize(batch):  
    return tokenizer(batch["text"], truncation=True,  
                      max_length=128, return_special_tokens_mask=True)  
  
ds_mlm = ds.map(tokenize, batched=True)  
ds_mlm = ds_mlm.remove_columns(["labels", "text", "label_ids"])
```

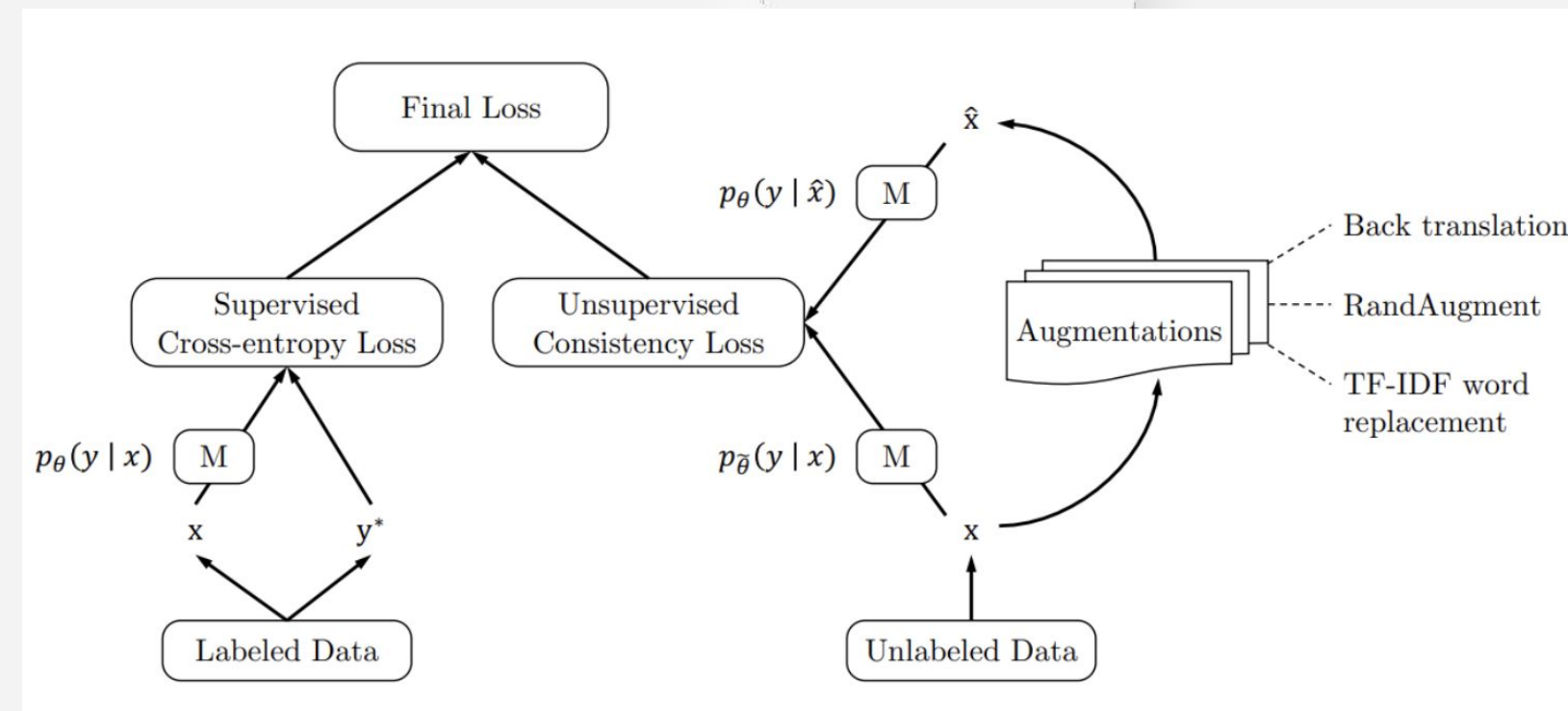
- fine tuning

```
from transformers import AutoModelForMaskedLM  
  
training_args = TrainingArguments(  
    output_dir = f"{model_ckpt}-issues-128", per_device_train_batch_size=32,  
    logging_strategy="epoch", evaluation_strategy="epoch", save_strategy="no",  
    num_train_epochs=16, push_to_hub=True, log_level="error", report_to="none")  
  
trainer = Trainer(  
    model=AutoModelForMaskedLM.from_pretrained("bert-base-uncased"),  
    tokenizer=tokenizer, args=training_args, data_collator=data_collator,  
    train_dataset=ds_mlm["unsup"], #ds_mlm["unsup"]: 레이블(토큰)이 없는 데이터  
    eval_dataset=ds_mlm["train"])  
  
trainer.train()
```

4. 레이블링되지 않은 데이터 활용

4.2 UDA (비지도 데이터 증식)

UDA는 Labeled 데이터와 Unlabeled 데이터를 함께 학습에 활용하는 Semi-supervised Learning 방법



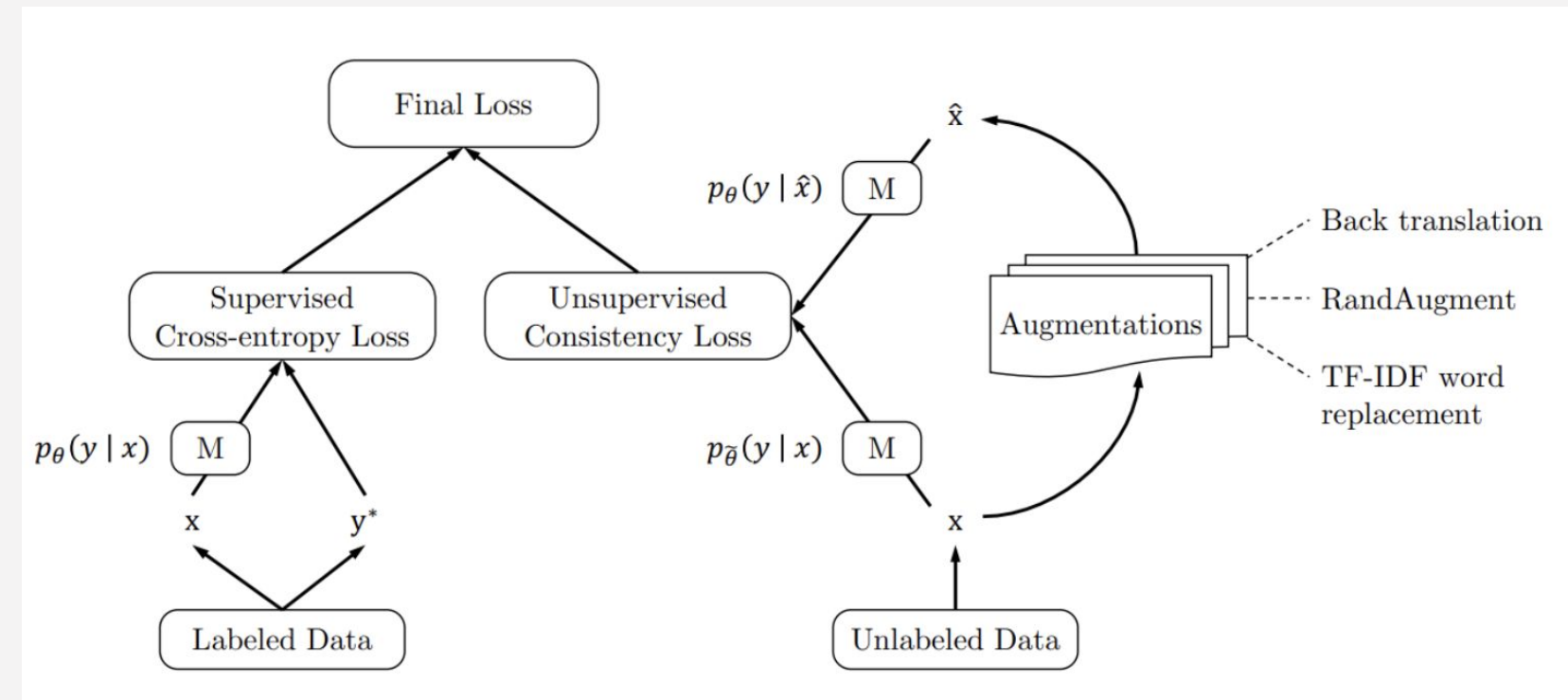
4. 레이블링되지 않은 데이터 활용

4.2 UDA (비지도 데이터 증식)

- Supervised Loss

일반적인 분류학습에 사용하는 cross entropy

- Consistency Loss

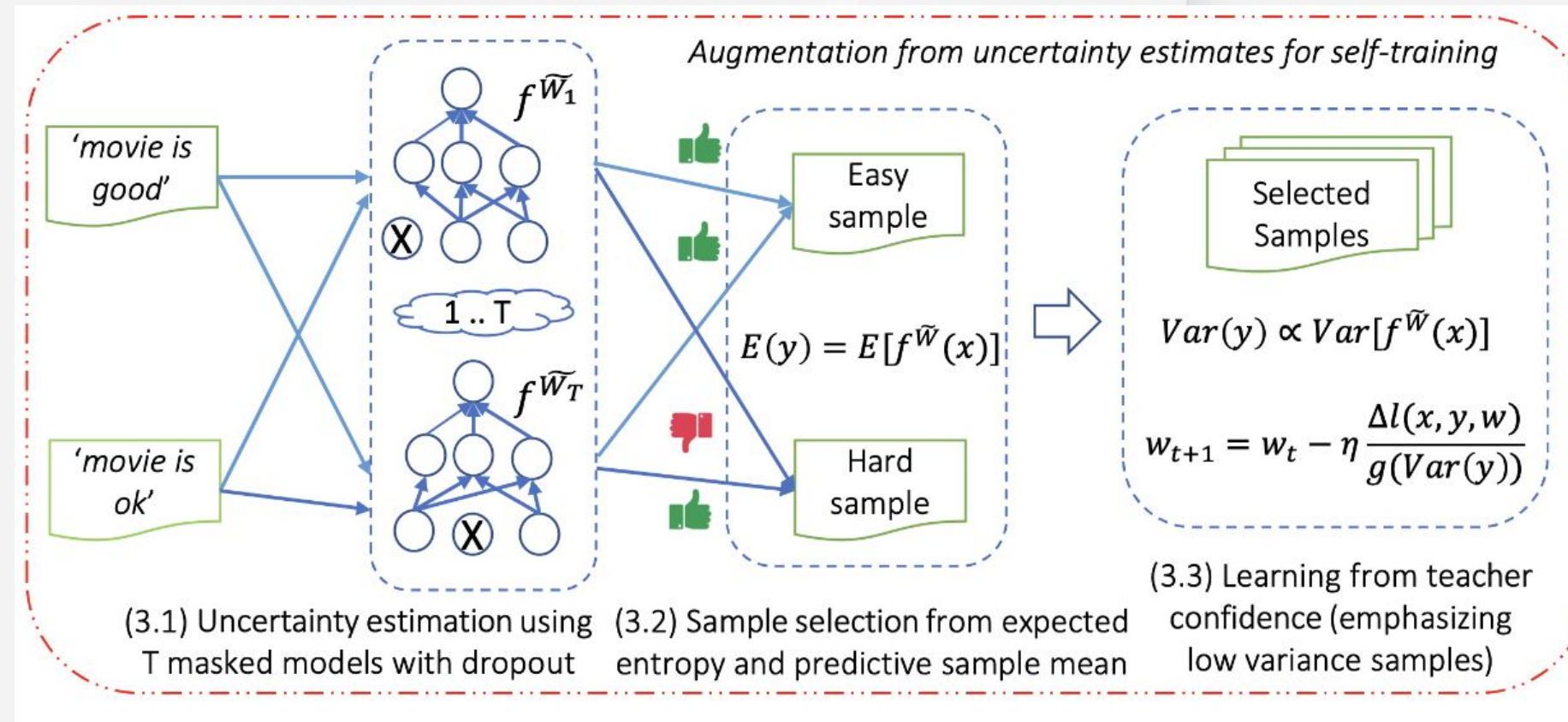


unlabeled 데이터의 문장 x 와 x 에 **augmentation**을 적용한 x^* 을 분류 모델에 넣어 두 개의 **확률 분포**를 추출한 뒤 두 확률분포의 차이인 **KL-Divergence**를 계산하여 **consistency loss**로 활용한다.

4. 레이블링되지 않은 데이터 활용

4.3 UST (불확실성 인지 자기훈련)

- Labeled 데이터와 Unlabeled 데이터를 함께 학습에 활용하는 Semi-supervised Learning 방법
- labeled된 데이터에서 teacher모델을 학습
- teacher모델을 사용해 레이블링 되지 않은 데이터에서 pseudo-label을 만들고 pseudo-label을 이용해 student 학습



Questions

- UDA에서 원본 데이터와 증강 데이터의 두 확률분포 차이를 최소화 하는 것이 왜 학습에 도움이 될까?
- 최근 현업에서 사용하는 semi supervised learning 기법이 뭘까?

