

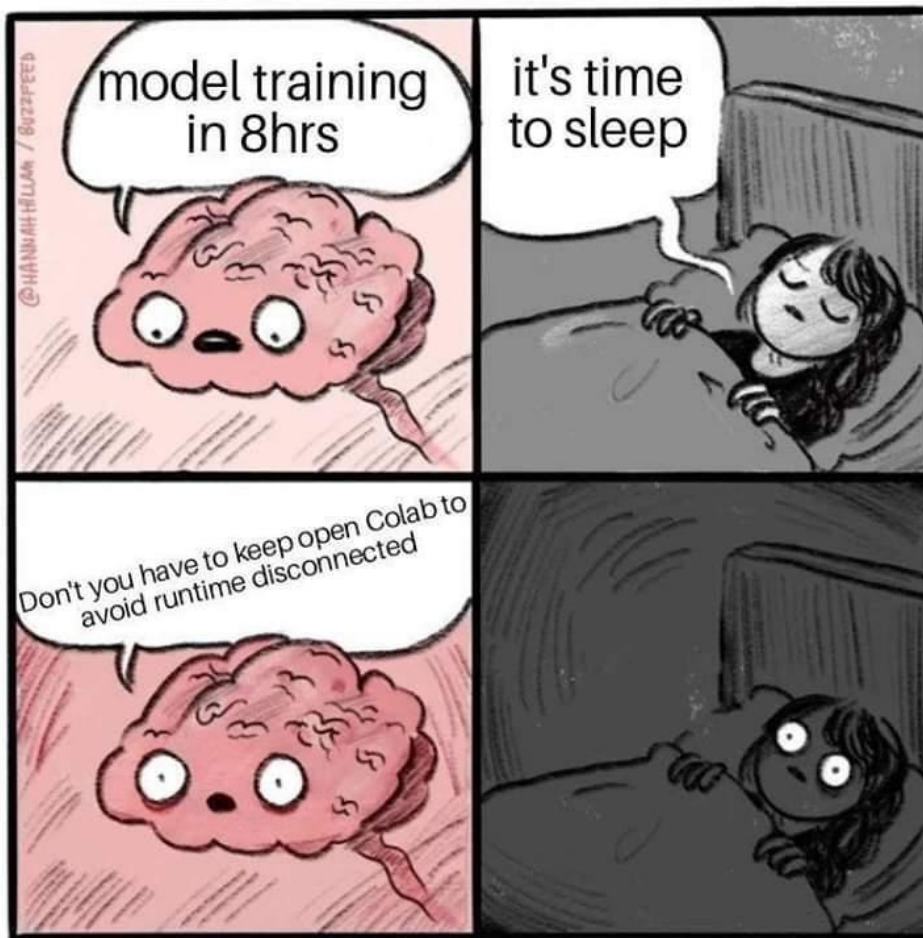
모델 불러오기

TEAMLAB director

최성철

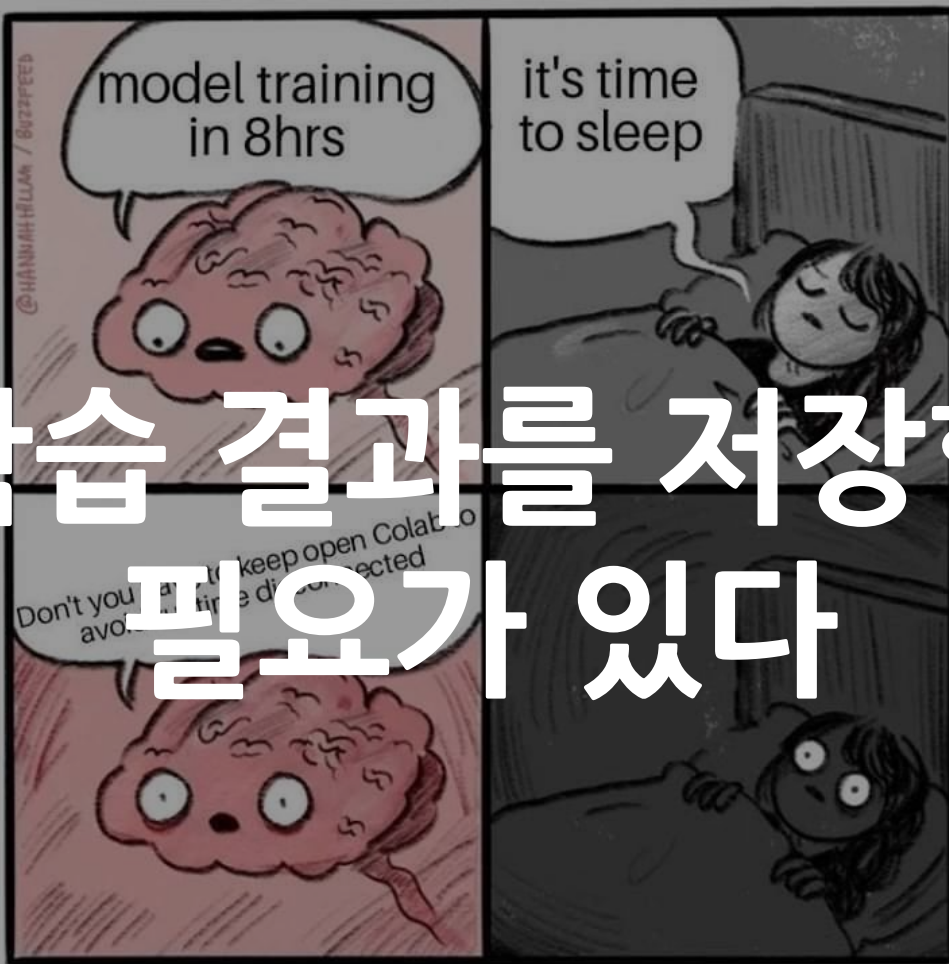
WARNING: 본 교육 콘텐츠의 지식재산권은 재단법인 네이버커넥트에 귀속됩니다. 본 콘텐츠를 어떠한 경로로든 외부로 유출 및 수정하는 행위를 엄격히 금합니다.
다만, 비영리적 교육 및 연구활동에 한정되어 사용할 수 있으나 재단의 허락을 받아야 합니다. 이를 위반하는 경우, 관련 법률에 따라 책임을 질 수 있습니다.

학습 결과를 공유하고 싶다



<https://stackoverflow.com/questions/57113226/how-can-i-prevent-google-colab-from-disconnecting>

학습 결과를 저장할 필요가 있다



`model.save()`

- 학습의 결과를 저장하기 위한 함수
- 모델 형태(**architecture**)와 파라미터를 저장
- 모델 학습 중간 과정의 저장을 통해 최선의 결과모델을 선택
- 만들어진 모델을 외부 연구자와 공유하여 학습 재연성 향상

model.save()

모델 불러오기

```
# Print model's state_dict
print("Model's state_dict:")
for param_tensor in model.state_dict():
    print(param_tensor, "\t", model.state_dict()[param_tensor].size())
```

state_dict : 모델의 파라미터를 표시

```
torch.save(model.state_dict(),
           os.path.join(MODEL_PATH, "model.pt"))
```

모델의 파라미터를 저장

```
new_model = TheModelClass()
```

같은 모델의 형태에서 파라미터만 load

```
new_model.load_state_dict(torch.load(os.path.join(
    MODEL_PATH, "model.pt")))
```

모델의 architecture와 함께 저장

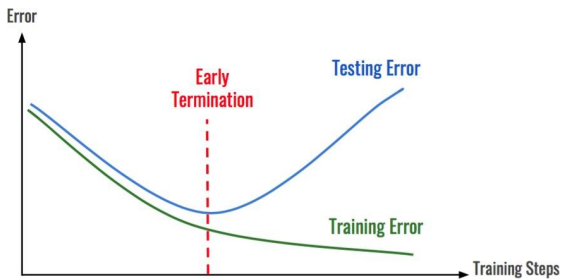
```
torch.save(model, os.path.join(MODEL_PATH, "model.pt"))
```

```
model = torch.load(os.path.join(MODEL_PATH, "model.pt"))
```

모델의 architecture와 함께 load

checkpoints

- 학습의 중간 결과를 저장하여 최선의 결과를 선택
- earlystopping 기법 사용시 이전 학습의 결과물을 저장
- loss와 metric 값을 지속적으로 확인 저장
- 일반적으로 epoch, loss, metric을 함께 저장하여 확인
- colab에서 지속적인 학습을 위해 필요



<https://medium.com/analytics-vidhya/early-stopping-with-pytorch-to-restrain-your-model-from-overfitting-dce6de4081c5>

모델의 정보를 epoch과 함께 저장

```
torch.save({  
    'epoch': e,  
    'model_state_dict': model.state_dict(),  
    'optimizer_state_dict': optimizer.state_dict(),  
    'loss': epoch_loss  
},  
f"saved/checkpoint_model_{e}_{epoch_loss/len(dataloader)}_{epoch_acc/len(dataloader)}.pt")
```

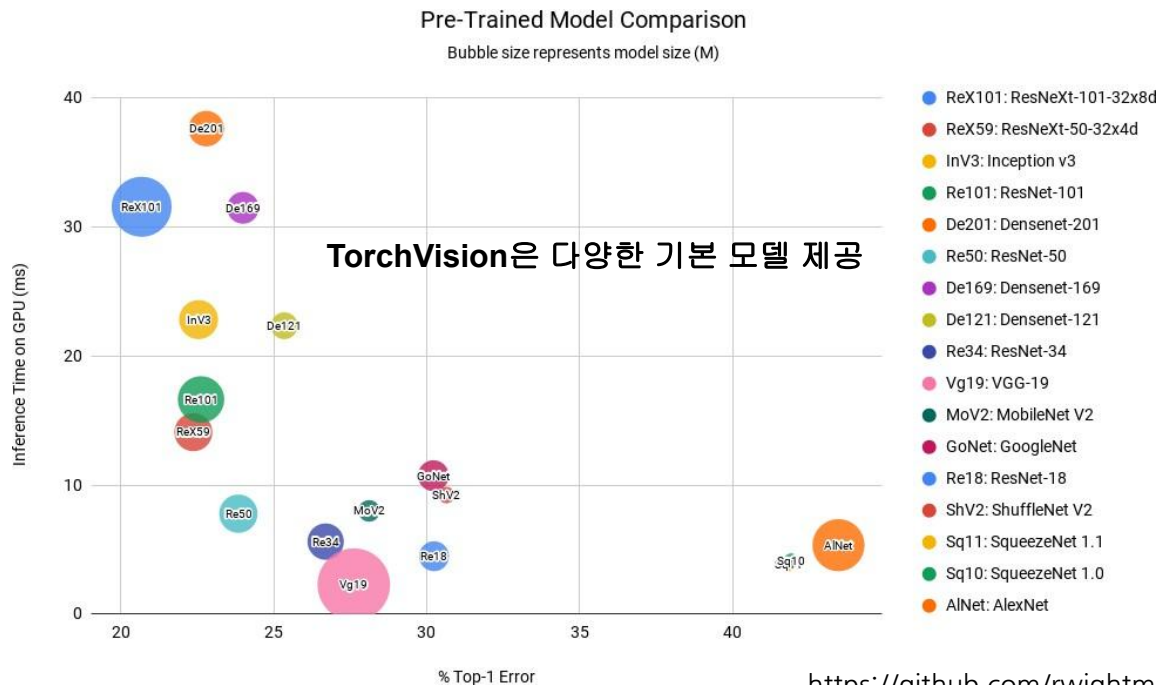
```
checkpoint = torch.load(PATH)  
model.load_state_dict(checkpoint['model_state_dict'])  
optimizer.load_state_dict(checkpoint['optimizer_state_dict'])  
epoch = checkpoint['epoch']  
loss = checkpoint['loss']
```

pretrained model

Transfer learning

남이 만든 모델을 쓰고 싶다

- 다른 데이터셋으로 만든 모델을 현재 데이터에 적용
- 일반적으로 대용량 데이터셋으로 만들어진 모델의 성능↑
- 현재의 DL에서는 가장 일반적인 학습 기법
- **backbone architecture**가 잘 학습된 모델에서 일부분만 변경하여 학습을 수행함



<https://github.com/rwightman/pytorch-image-models#introduction>

Transfer learning

모델 불러오기

The screenshot shows the Hugging Face website's 'Models' section. On the left, there are filters for Tasks, Libraries, Datasets, Languages, and Licenses. The main area displays a grid of model cards, each showing the model name, task, update date, and download count. The models are sorted by 'Most Downloads'.

Tasks: Fill-Mask, Question Answering, Summarization, Table Question Answering, Text Classification, Text Generation, Text2Text Generation, Token Classification, Translation, Zero-Shot Classification, Sentence Similarity (+11)

Libraries: PyTorch, TensorFlow, JAX (+19)

Datasets: wikipedia, conll2003, common_voice, deep_europarl_jrc-acquis, squad, oscar, bookcorpus, CLUECorpusSmall (+433)

Languages: en, es, fr, de, sv, fi, zh, ru (+157)

Licenses: apache-2.0, mit, cc-by-4.0 (+28)

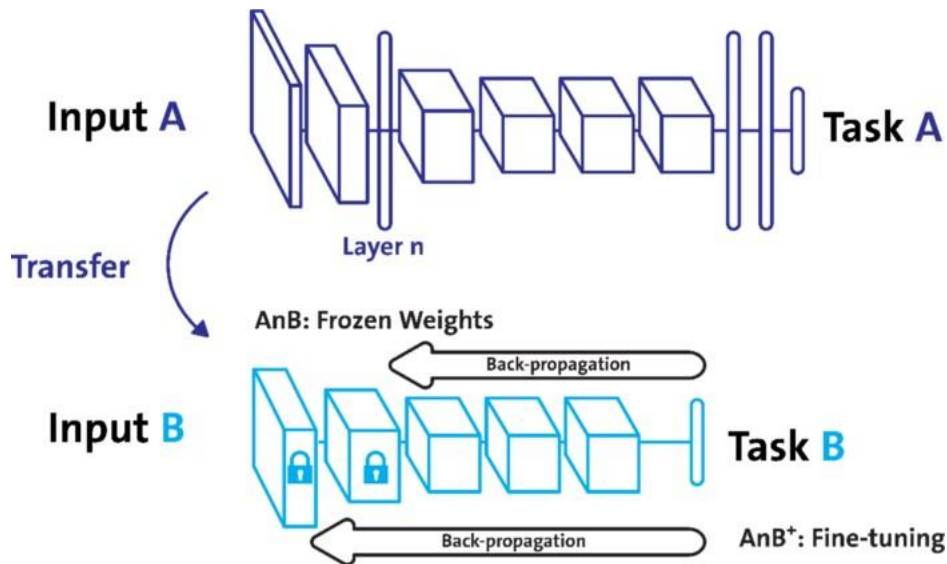
Models (13,303):

- bert-base-uncased**: Fill-Mask · Updated May 18 · 131M · ♥ 17
- roberta-large**: Fill-Mask · Updated May 21 · 8M · ♥ 6
- bert-base-cased**: Fill-Mask · Updated May 18 · 3.7M · ♥ 3
- sentence-transformers/paraphrase-xlm-r-multiling...**: Sentence Similarity · Updated 2 days ago · 2.57M · ♥ 2
- roberta-base**: Fill-Mask · Updated Jul 6 · 2.48M
- xlm-roberta-base**: Fill-Mask · Updated Dec 11, 2020 · 1.48M
- t5-base**: Translation · Updated Jun 23 · 1.02M · ♥ 2
- hfl/chinese-roberta-wwm-ext**: Fill-Mask · Updated May 19 · 628k · ♥ 3
- roberta-large-mnli**: Text Classification · Updated May 20 · 514k · ♥ 1
- bert-base-chinese**: Fill-Mask · Updated May 18 · 420k · ♥ 2
- bert-base-multilingual-uncased**: Fill-Mask · Updated May 18 · 404k · ♥ 2
- bert-large-cased**: Fill-Mask · Updated May 18 · 398k · ♥ 1
- bert-large-uncased-whole-word-masking-finetuned-sq...**: Question Answering · Updated May 18 · 11.4M · ♥ 3
- distilbert-base-uncased**: Fill-Mask · Updated Dec 11, 2020 · 5.8M · ♥ 13
- distilbert-base-uncased-finetuned-sst-2-english**: Text Classification · Updated Feb 9 · 3.15M · ♥ 4
- google/bert_uncased_L-12_H-512_A-8**: Updated May 18 · 2.51M
- finiteautomata/beto-sentiment-analysis**: Text Classification · Updated Jul 22 · 1.5M · ♥ 2
- gpt2**: Text Generation · Updated May 19 · 1.35M · ♥ 5
- facebook/m2m100_418M**: Text2Text Generation · Updated Jun 12 · 964k · ♥ 1
- bert-base-multilingual-cased**: Fill-Mask · Updated May 18 · 584k · ♥ 1
- bert-large-uncased**: Fill-Mask · Updated May 18 · 510k · ♥ 3
- albert-base-v2**: Fill-Mask · Updated Jan 13 · 405k
- deepset/roberta-base-squad2**: Question Answering · Updated 5 days ago · 402k · ♥ 1
- t5-small**: Translation · Updated Jun 23 · 389k

NLP는 HuggingFace가 사실상 표준

<https://huggingface.co/models>

- pretrained model을 활용시 모델의 일부분을 frozen 시킴



<https://purnasaigudikandula.medium.com/deep-view-on-transfer-learning-with-image-classification-pytorch-5cf963939575>


```
vgg = models.vgg16(pretrained=True).to(device)
```

vgg16 모델을 vgg에 할당하기

```
class MyNewNet(nn.Module):  
    def __init__(self):  
        super(MyNewNet, self).__init__()  
        self.vgg19 = models.vgg19(pretrained=True)  
        self.linear_layers = nn.Linear(1000, 1)
```

모델에 마지막 Linear Layer 추가

```
# Defining the forward pass  
def forward(self, x):  
    x = self.vgg19(x)  
    return self.linear_layers(x)
```

```
for param in my_model.parameters():  
    param.requires_grad = False  
for param in my_model.linear_layers.parameters():  
    param.requires_grad = True
```

마지막 레이어를 제외하고 frozen

End of Document
Thank You.