

---

## Ampliación de Fundamentos de Hardware

Angel Fabrizio Ullaguari Yanza

2ºASIR

# Práctica IA. Destilación



---

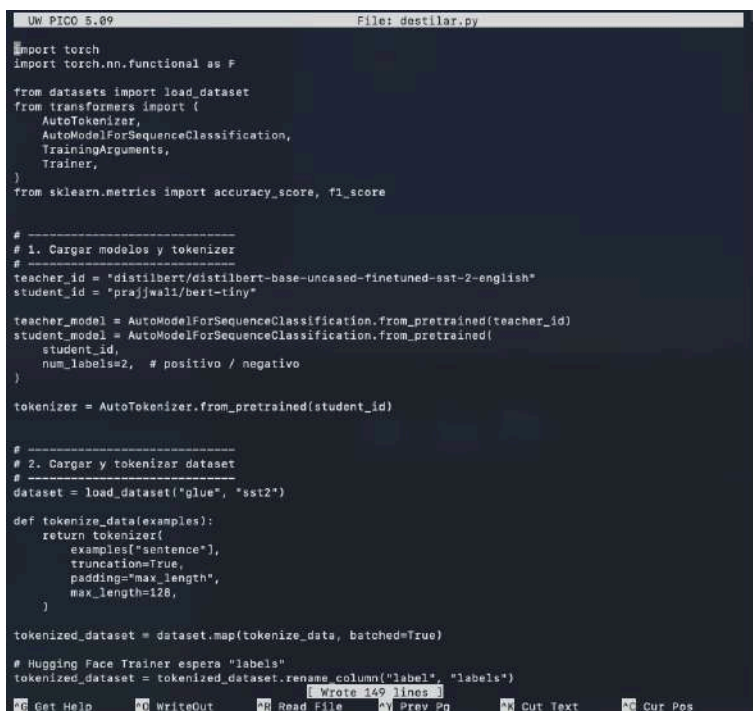
|                 |          |
|-----------------|----------|
| <b>Objetivo</b> | <b>3</b> |
| <b>Práctica</b> | <b>3</b> |

## Objetivo

Tomar un modelo profesor (basado en DistilBERT), ya afinado para análisis de sentimientos, y destilar su conocimiento a un modelo estudiante (BERT-tiny) que es ~15 veces más pequeño. El resultado será un clasificador de sentimientos ultraligero y veloz.

## Configuración

Para empezar, creamos una carpeta limpia para el proyecto y configuramos un entorno virtual con python, creando así el archivo [destilar.py](#). Con el comando nano ponemos el código puesto en el enunciado, que pide expresamente un modelo de IA y otro más pequeño (profesor y estudiante). Pero antes de crearlo, se **necesita tener instaladas las librerías de Hugging Face (torch, transformers, peft, etc...)**.



```
UW PICO 5.89 File: destilar.py

import torch
import torch.nn.functional as F

from datasets import load_dataset
from transformers import (
    AutoTokenizer,
    AutoModelForSequenceClassification,
    TrainingArguments,
    Trainer,
)
from sklearn.metrics import accuracy_score, f1_score

# -----
# 1. Cargar modelos y tokenizer
# -----
teacher_id = "distilbert/distilbert-base-uncased-finetuned-sst-2-english"
student_id = "prajjwal1/bert-tiny"

teacher_model = AutoModelForSequenceClassification.from_pretrained(teacher_id)
student_model = AutoModelForSequenceClassification.from_pretrained(
    student_id,
    num_labels=2, # positivo / negativo
)

tokenizer = AutoTokenizer.from_pretrained(student_id)

# -----
# 2. Cargar y tokenizer dataset
# -----
dataset = load_dataset("glue", "sst2")

def tokenize_data(examples):
    return tokenizer(
        examples["sentence"],
        truncation=True,
        padding="max_length",
        max_length=128,
    )

tokenized_dataset = dataset.map(tokenize_data, batched=True)

# Hugging Face Trainer espera "labels"
tokenized_dataset = tokenized_dataset.rename_column("label", "labels")

Wrote 149 lines
```

Después de guardar los cambios, pasamos al destilar nuestro modelo: ejecutamos con python el archivo. Al principio, nos aparecen unos avisos de la librería de Hugging Face pero no hemos tenido ningún problema.

```
(.venv) aualaateca26@Mac-mini-de-aualaateca26 destilacion-ia % cd /Users/aualaateca26/2asir/destilacion-ia
a
source .venv/bin/activate
python destilar.py

Loading weights: 100%|█| 104/104 [00:00<00:00, 5839.07it/s, Materializing param=pre_classifier.weight]
Warning: You are sending unauthenticated requests to the HF Hub. Please set a HF_TOKEN to enable higher rate limits and faster downloads.
Loading weights: 100%|█| 39/39 [00:00<00:00, 9090.69it/s, Materializing param=bert.pooler.dense.weight]
BertForSequenceClassification LOAD REPORT from: prajjwall/bert-tiny
Key | Status |
-----|-----|
cls.seq_relationship.weight | UNEXPECTED |
cls.predictions.transform.dense.bias | UNEXPECTED |
cls.predictions.transform.LayerNorm.weight | UNEXPECTED |
cls.predictions.transform.dense.weight | UNEXPECTED |
cls.predictions.decoder.bias | UNEXPECTED |
cls.seq_relationship.bias | UNEXPECTED |
cls.predictions.decoder.weight | UNEXPECTED |
cls.predictions.transform.LayerNorm.bias | UNEXPECTED |
cls.predictions.bias | UNEXPECTED |
bert.embeddings.position_ids | UNEXPECTED |
classifier.bias | MISSING |
classifier.weight | MISSING |

Notes:
- UNEXPECTED :can be ignored when loading from different task/architecture; not ok if you expect identical arch.
- MISSING :those params were newly initialized because missing from the checkpoint. Consider training on your downstream task.
Iniciando la destilacion (entrenamiento recortado)...
0%| | 0/1000 [00:00<?, ?it/s]
/Users/aualaateca26/2asir/destilacion-ia/.venv/lib/python3.14/site-packages/torch/utils/data/dataloader.py:775: UserWarning: 'pin_memory' argument is set as true but not supported on MPS now, device pinned memory won't be used.
  super().__init__(loader)
{'loss': '1.934', 'grad_norm': '1.703', 'learning_rate': '4.755e-05', 'epoch': '0.1597'}
5%| | 54/1000 [00:09<02:42, 5.84it/s]
```

Ahora creamos el archivo probar\_modelo.py para ver si el modelo estudiante funciona.

```
UW PICO 5.09 File: probar_modelo.py

import torch
from transformers import AutoTokenizer, AutoModelForSequenceClassification

# Usamos directamente el identificador del modelo estudiante
base_id = "prajjwall/bert-tiny"

# Cargamos el tokenizer y el modelo de Hugging Face
tokenizer = AutoTokenizer.from_pretrained(base_id)
student_model = AutoModelForSequenceClassification.from_pretrained(
    base_id,
    num_labels=2, # positivo / negativo
)

student_model.eval()

def clasificar(texto: str):
    inputs = tokenizer(
        texto,
        return_tensors="pt",
        truncation=True,
        padding=True,
        max_length=128,
    )
    with torch.no_grad():
        outputs = student_model(**inputs)
        probs = torch.softmax(outputs.logits, dim=-1)[0]
        etiqueta = "positivo" if probs[1] > probs[0] else "negativo"
        print(f"Texto: {texto}")
        print(f"Predicción: {etiqueta} (probs = {probs.tolist()})\n")

if __name__ == "__main__":
    clasificar("This movie was fantastic, I really loved it!")
    clasificar("This movie was terrible, I will never watch it again.")
```

Lo ejecutamos:

```
(.venv) aulaateca26@Mac-mini-de-aulaateca26 destilacion-ia % cd /Users/aulaateca26/2asir/destilacion-ia
source .venv/bin/activate
python probar_modelo.py

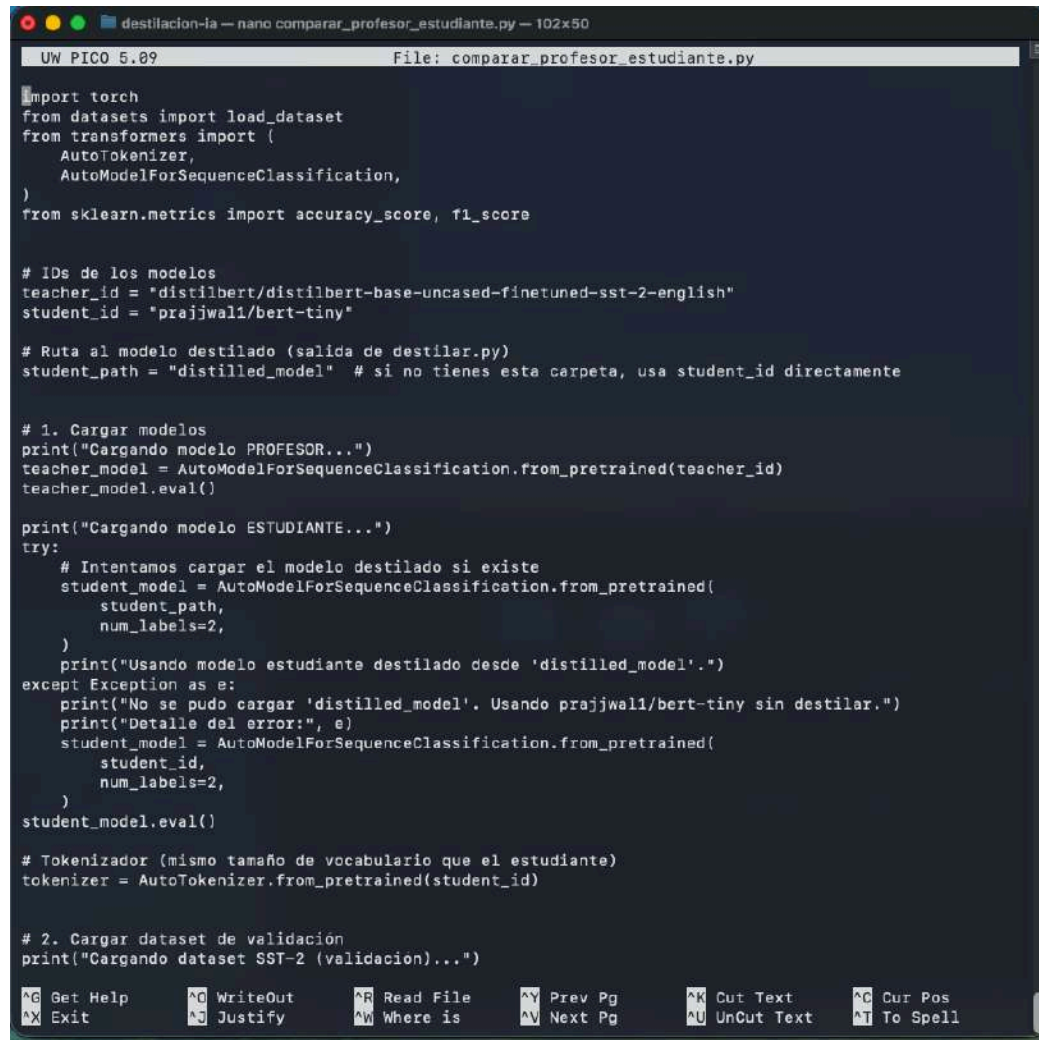
Warning: You are sending unauthenticated requests to the HF Hub. Please set a HF_TOKEN to enable higher rate limits and faster downloads.
Loading weights: 100%|██████████| 39/39 [00:00<00:00, 11080.26it/s, Materializing param=bert.pooler.dense.weight]
BertForSequenceClassification LOAD REPORT from: prajjwall/bert-tiny
Key | Status |
-----|-----|
cls.predictions.transform.dense.bias | UNEXPECTED |
cls.predictions.decoder.bias | UNEXPECTED |
cls.seq_relationship.weight | UNEXPECTED |
cls.predictions.bias | UNEXPECTED |
cls.predictions.transform.LayerNorm.weight | UNEXPECTED |
cls.predictions.transform.LayerNorm.bias | UNEXPECTED |
cls.predictions.decoder.weight | UNEXPECTED |
bert.embeddings.position_ids | UNEXPECTED |
cls.predictions.transform.dense.weight | UNEXPECTED |
cls.seq_relationship.bias | UNEXPECTED |
classifier.weight | MISSING |
classifier.bias | MISSING |

Notes:
- UNEXPECTED :can be ignored when loading from different task/architecture; not ok if you expect identical arch.
- MISSING :those params were newly initialized because missing from the checkpoint. Consider training on your downstream task.
Texto: This movie was fantastic, I really loved it!
Predicción: positivo (probs = [0.3990715742111206, 0.6009284257888794])

Texto: This movie was terrible, I will never watch it again.
Predicción: positivo (probs = [0.3918722867965698, 0.6081277132034302])

(.venv) aulaateca26@Mac-mini-de-aulaateca26 destilacion-ia %
```

Con el entrenamiento finalizado, ahora se debe crear un nuevo archivo .py para comparar las estadísticas del modelo profesor y estudiante.



```
destilacion-ia - nano comparar_profesor_estudiante.py - 102x50
UW PICO 5.09 File: comparar_profesor_estudiante.py

import torch
from datasets import load_dataset
from transformers import (
    AutoTokenizer,
    AutoModelForSequenceClassification,
)
from sklearn.metrics import accuracy_score, f1_score

# IDs de los modelos
teacher_id = "distilbert/distilbert-base-uncased-finetuned-sst-2-english"
student_id = "prajjwal1/bert-tiny"

# Ruta al modelo destilado (salida de destilar.py)
student_path = "distilled_model" # si no tienes esta carpeta, usa student_id directamente

# 1. Cargar modelos
print("Cargando modelo PROFESOR...")
teacher_model = AutoModelForSequenceClassification.from_pretrained(teacher_id)
teacher_model.eval()

print("Cargando modelo ESTUDIANTE...")
try:
    # Intentamos cargar el modelo destilado si existe
    student_model = AutoModelForSequenceClassification.from_pretrained(
        student_path,
        num_labels=2,
    )
    print("Usando modelo estudiante destilado desde 'distilled_model'.")
except Exception as e:
    print("No se pudo cargar 'distilled_model'. Usando prajjwal1/bert-tiny sin destilar.")
    print("Detalle del error:", e)
    student_model = AutoModelForSequenceClassification.from_pretrained(
        student_id,
        num_labels=2,
    )
student_model.eval()

# Tokenizador (mismo tamaño de vocabulario que el estudiante)
tokenizer = AutoTokenizer.from_pretrained(student_id)

# 2. Cargar dataset de validación
print("Cargando dataset SST-2 (validación)...")

^G Get Help      ^O WriteOut      ^R Read File     ^Y Prev Pg      ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify       ^W Where is      ^V Next Pg      ^U UnCut Text    ^T To Spell
```



## Comparativa

Ejecutamos el archivo .py:

```
destilacion-ia -- -zsh -- 102x50

formerConfig, LukeConfig, MarkupLMConfig, MBartConfig, MegatronBertConfig, MiniMaxConfig, MinistralCon
fig, Ministral3Config, MistralConfig, MixtralConfig, MobileBertConfig, ModernBertConfig, ModernBertDec
oderConfig, MPNetConfig, MptConfig, MraConfig, MT5Config, NvqConfig, NemotronConfig, NystromformerConf
ig, OpenAIGPTConfig, OPTConfig, PerceiverConfig, PersimmonConfig, PhiConfig, Phi3Config, PhimoeConfig,
PiBartConfig, Qwen2Config, Qwen2MoeConfig, Qwen3Config, Qwen3MoeConfig, Qwen3NextConfig, ReformerConf
ig, RemBertConfig, RobertaConfig, RobertaPreLayerNormConfig, RoCBertConfig, RoformerConfig, SeedOssCon
fig, SmolLM3Config, SqueezeBertConfig, StableLmConfig, Starcoder2Config, T5Config, T5GemmaConfig, T5Ge
mma2Config, TapasConfig, UMT5Config, XLMConfig, XLMRobertaConfig, XLMRobertaXLConfig, XLNetConfig, Xmo
dConfig, YosoConfig, ZambaConfig, Zamba2Config.
Loading weights: 100%| 39/39 [00:00<00:00, 8383.45it/s, Materializing param=bert.pooler.dense.weight
BertForSequenceClassification LOAD REPORT from: prajjwall/bert-tiny
Key | Status |
-----|-----|
cls.seq_relationship.weight | UNEXPECTED |
cls.predictions.transform.LayerNorm.bias | UNEXPECTED |
cls.predictions.transform.dense.weight | UNEXPECTED |
cls.seq_relationship.bias | UNEXPECTED |
cls.predictions.decoder.weight | UNEXPECTED |
cls.predictions.transform.LayerNorm.weight | UNEXPECTED |
cls.predictions.decoder.bias | UNEXPECTED |
bert.embeddings.position_ids | UNEXPECTED |
cls.predictions.bias | UNEXPECTED |
cls.predictions.transform.dense.bias | UNEXPECTED |
classifier.weight | MISSING |
classifier.bias | MISSING |

Notes:
- UNEXPECTED :can be ignored when loading from different task/architecture; not ok if you expect id
ential arch.
- MISSING :those params were newly initialized because missing from the checkpoint. Consider tra
ining on your downstream task.
Cargando dataset SST-2 (validación)...
Map: 100%| 872/872 [00:00<00:00, 27999.06 examples/s]

Resultados para PROFESOR (DistilBERT):
Accuracy: 0.9106
F1-score: 0.9105

Resultados para ESTUDIANTE (BERT-tiny destilado):
Accuracy: 0.4874
F1-score: 0.3390

--- COMPARACIÓN PROFESOR vs ESTUDIANTE ---
Profesor - Accuracy: 0.9106 | F1: 0.9105
Estudiante- Accuracy: 0.4874 | F1: 0.3390
Diferencia de accuracy: 0.4232
Diferencia de F1: 0.5715
(.venv) aulasteca26@Mac-mini-de-sulasteca26 destilacion-ia %
```

Al final nos da unos valores que muestran la comparación de precisión y velocidad de cada modelo. El objetivo se ha cumplido, pues se ve que el modelo estudiante da menos parámetros que el modelo profesor.