Homework for **L02**
name: FINICHIU Eduard - Adelin
group: 1241 EB
FILS CTI-(E)

The teacher asked the following questions:

- ❑ What worked well?
- ❑ Did the model learn the **style**?
- ❑ Any interesting, funny, or weird results?
- ❑ Would you change anything next time?

Responses:

1. The coding part went well: I managed to adjust the parameters to my needs. I have used a smaller database, a .CSV made by me, with hundreds of jokes. Making the database was easy, by using jokes from Huggingface databases. The Google Colab worked well, but I was still wondering if it will stop (due to free trial)

2. He learned the style, he became more funny, but it is not that obvious. I was not able to make it very obvious, because not even ChatGPT 5 can do good jokes.

3. Yes, I had a weird result:

```
Tell me a joke! What you really mean is something like, 'Hey look at this girl wearing pink dress.
```

   These kinds of jokes were common. The jokes are not funny, because… they do not have a good sense of humor :) But, they sound kind of weird.

4. I would like to use a bigger database, I hope that this would make the GPT do better jokes.

I decided to add top_k and top_p for better results:

```
output = model.generate(
    **inputs,
    max_length=100,
    do_sample=True,        # sampling in loc de greedy
    top_k=50,              # ia cele mai probabile 50 tokenuri
    top_p=0.95,            # nucleu sampling
    repetition_penalty=1.3
)
```

I adjusted code for the new database:

```
# write the code here
def tokenize_dataset(text):
  tokenizer.pad_token = tokenizer.eos_token
  return tokenizer(text["Joke"], truncation=True, padding="max_length", max_length=128)

tokenized_dataset = dataset.map(tokenize_dataset, batched=True)
```

I trained the model:

1. Training Arguments

```
# write the code here
training_args = TrainingArguments(
    output_dir="./jokes-finetuned",
    per_device_train_batch_size=8,
    num_train_epochs=3,
    save_steps=250,
    logging_steps=20,
    learning_rate=5e-5,
    warmup_steps=50,
    weight_decay=0.01,
    report_to="none"
)
```

I observed results:

```
# write the code here
from transformers import DataCollatorForLanguageModeling
data_collator = DataCollatorForLanguageModeling(
    tokenizer=tokenizer, mlm=False,
)

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_dataset["train"],
    tokenizer=tokenizer,
    data_collator=data_collator,
)

# generate a joke
prompt = "Tell me a joke!"
inputs = tokenizer(prompt, return_tensors="pt").to(model.device)

output = model.generate(
    **inputs,
    max_length=25,
    do_sample=True,
    top_k=50,
    top_p=0.95,
    repetition_penalty=1.2
)

print(tokenizer.decode(output[0], skip_special_tokens=True))
```

I observed bad jokes:

```
# generate a joke
prompt = "Bananas are"
inputs = tokenizer(prompt, return_tensors="pt").to(model.device)

output = model.generate(
    **inputs,
    max_length=25,
    do_sample=True,
    top_k=50,
    top_p=0.95,
    repetition_penalty=1.2
)

print(tokenizer.decode(output[0], skip_special_tokens=True))
```

```
/tmp/ipython-input-576923811.py:7: FutureWarning: `tokenizer` is deprecated and will be removed in version 5.0.0 for `
  trainer = Trainer(
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
Bananas are so delicious you can't eat the ones they had and it's like making a sandwich with them instead of eating
```

Good old AI jokes:

```
prompt = "The cow said"
inputs = tokenizer(prompt, return_tensors="pt").to(model.device)

output = model.generate(
    **inputs,
    max_length=25,
    do_sample=True,
    top_k=80,
    top_p=0.75,
    repetition_penalty=1.2
)

print(tokenizer.decode(output[0], skip_special_tokens=True))
```

```
/tmp/ipython-input-763339191.py:7: FutureWarning: `tokenizer` is deprecated and will be r
  trainer = Trainer(
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
The cow said, "I don't know what happened but I have no idea where the herd came from."
One of
```

I used a CSV file with 300 jokes as my dataset. First, I cleaned and tokenized the text so the model could understand it. Then, I trained the model using Hugging Face's Trainer. After training, the model can make new jokes if you give it a prompt. This project shows how you can use AI to generate funny text and learn how text models work.

I also learned how to set up the training so the model doesn't overfit on such a small dataset. I used tricks like padding and truncation to make all the jokes the same length. For generating new jokes, I tried different settings to make the outputs fun and not repetitive. It was cool to see the model come up with stuff that actually made sense and sometimes was funny. Overall, it helped me understand how AI can learn patterns from text and create new content on its own.