Untitled

November 12, 2024

[1]:
```
import pandas as pd
from datasets import Dataset, load_dataset





jester_data = load_dataset("SeppeV/rated_jokes_dataset_from_jester",␣
 ↪split="train")

# Extract jokes from the Jester dataset using 'jokeText' field
jester_jokes = [entry['jokeText'] for entry in jester_data if 'jokeText' in␣
 ↪entry]

# Your original funny, silly dataset with example statements

# Your original funny, silly dataset with example statements
data = {
    "Text": [
        # NFTs
        "NFTs: Proof that we can now own 'priceless art' that even your dog can␣
 ↪screenshot.",
        "NFTs: For those who think 'why save my money when I can buy imaginary␣
 ↪things?'",
        "NFTs are like collecting stamps, but with zero paper and 100% more␣
 ↪existential dread.",
        "NFTs: now you too can pay for art that’s all pixels and zero paint␣
 ↪splatters.",

        # Web3
        "Web3: like the internet but spicier, with a side of privacy drama.  ",
        "Welcome to Web3: where you’re the CEO of your wallet and your own␣
 ↪worst enemy.",
        "Web3: the only place where 'community governance' means arguing on␣
 ↪Discord at 2 AM.",
        "Web3: 'cause nothing says innovation like reinventing the internet and␣
 ↪a million acronyms.",
        # Crypto
        "Crypto: The digital casino where everyone's all-in, but no one knows␣
↪the rules.",
        "Why have stable income when you can have unstable crypto? #YOLO",
        "Crypto: for people who enjoy watching numbers dance and heart rates␣
↪spike.",
        "HODLing crypto is like dating: a thrilling mess with occasional 'what␣
↪am I doing?' moments.",

        # Finance
        "Finance tip: Always invest in yourself-unless you're a stress-prone␣
↪HODLer.  ",
        "Budgeting advice: 'Buy low, sell high' or just buy pizza, honestly␣
↪both are wins.",
        "Investing: the art of guessing but with extra steps and occasional␣
↪tears.",
        "Finance: the fine line between planning for the future and surviving␣
↪on ramen.",

        # Pop Culture & Comedian-Inspired
        "As Jerry Seinfeld would say: 'What's the deal with NFTs? You're paying␣
↪for a receipt, but where's the store?'",
        "In the words of John Mulaney: 'Do my crypto investments give me␣
↪stability? No. Do they give me anxiety? Yes. Do I love it? Well…'",
        "Crypto is like Fight Club. Rule #1: Don't talk about your gains unless␣
↪you want people asking about your losses.",
        "As Yoda would say: 'HODL, you must. Strong is the force of diamond␣
↪hands.'",
        # ...
    ]
}

data["Text"].extend(jester_jokes)

# Convert to a DataFrame and then to a Hugging Face Dataset
df = pd.DataFrame(data)
combined_dataset = Dataset.from_pandas(df)

# Display the combined dataset to verify
print(combined_dataset)
```

```
Dataset({
    features: ['Text'],
    num_rows: 1761459
})
```

[2]:
```
from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("EleutherAI/gpt-neo-1.3B")
tokenizer.pad_token = tokenizer.eos_token  # Set padding token to the␣
 ↪end-of-sequence token

# Tokenize the dataset
def tokenize(batch):
    return tokenizer(batch['Text'], padding=True, truncation=True)

tokenized_dataset = combined_dataset.map(tokenize, batched=True)
```

```
Map:   0%|          | 0/1761459 [00:00<?, ? examples/s]
```

[3]:
```
import os
os.environ["PYTORCH_MPS_HIGH_WATERMARK_RATIO"] = "0.0"

import os
import pandas as pd
from datasets import Dataset
from transformers import AutoTokenizer, AutoModelForCausalLM, Trainer,␣
 ↪TrainingArguments
import torch
import random

# Set TOKENIZERS_PARALLELISM to avoid warnings
os.environ["TOKENIZERS_PARALLELISM"] = "false"

# Set device to CPU or MPS if available
device = torch.device("mps" if torch.backends.mps.is_available() else "cpu")

# Load the tokenizer and model with reduced memory usage
tokenizer = AutoTokenizer.from_pretrained("EleutherAI/gpt-neo-1.3B")
tokenizer.pad_token = tokenizer.eos_token  # Use eos_token as pad_token for␣
 ↪GPT-2
model = AutoModelForCausalLM.from_pretrained(
    "EleutherAI/gpt-neo-2.7B",
    torch_dtype=torch.float16,       # Use float16 for memory savings on␣
 ↪compatible hardware
    low_cpu_mem_usage=True
).to(device)

# Disable gradient checkpointing if needed
model.gradient_checkpointing_disable()

data = {"Text": ["Example sentence for fine-tuning the model on funny text.",␣
 ↪"Another funny example."]}
```

```
combined_dataset = Dataset.from_dict(data)

# Split the dataset into train and test sets
split_dataset = combined_dataset.train_test_split(test_size=0.2)

# Tokenize the dataset with reduced max length
def tokenize_function(examples):
    inputs = tokenizer(examples["Text"], padding=True, truncation=True,␣
↪max_length=32)  # Reduced max length to 32
    inputs["labels"] = inputs["input_ids"].copy()  # Use input_ids as labels␣
↪for causal LM training
    return inputs

# Tokenize and preprocess the dataset
tokenized_dataset = split_dataset.map(tokenize_function, batched=True)

# Define training arguments with reduced batch size and increased gradient␣
↪accumulation steps
training_args = TrainingArguments(
    output_dir="./results",
    eval_strategy="epoch",
    learning_rate=1e-5,
    lr_scheduler_type="linear",
    warmup_steps=100,
    per_device_train_batch_size=1,         # Reduced batch size to 1
    gradient_accumulation_steps=32,         # Increased accumulation steps to␣
↪simulate larger batch size
    num_train_epochs=2,
    weight_decay=0.01,
    logging_dir="./logs"
)

# Initialize the Trainer
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_dataset["train"],
    eval_dataset=tokenized_dataset["test"]
)

# Train the model
trainer.train()
```

```
Map:   0%|          | 0/1 [00:00<?, ? examples/s]

Map:   0%|          | 0/1 [00:00<?, ? examples/s]

<IPython.core.display.HTML object>
```

[3]:
```
TrainOutput(global_step=2, training_loss=0.2020263671875,
metrics={'train_runtime': 143.8079, 'train_samples_per_second': 0.014,
'train_steps_per_second': 0.014, 'total_flos': 120835522560.0, 'train_loss':
0.2020263671875, 'epoch': 2.0})
```

[4]:
```
# Save the fine-tuned model and tokenizer
model.save_pretrained("./fine_tuned_personality_bot")
tokenizer.save_pretrained("./fine_tuned_personality_bot")
```

[4]:
```
('./fine_tuned_personality_bot/tokenizer_config.json',
 './fine_tuned_personality_bot/special_tokens_map.json',
 './fine_tuned_personality_bot/vocab.json',
 './fine_tuned_personality_bot/merges.txt',
 './fine_tuned_personality_bot/added_tokens.json',
 './fine_tuned_personality_bot/tokenizer.json')
```

[5]:
```
import random
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM

# Set device to CPU (you can change this to 'cuda' if you have a GPU available)
device = torch.device("cpu")

# Load the tokenizer and model
tokenizer = AutoTokenizer.from_pretrained("EleutherAI/gpt-neo-1.3B")
tokenizer.pad_token = tokenizer.eos_token  # Set pad_token to eos_token to␣
 ↪avoid warnings
model = AutoModelForCausalLM.from_pretrained("EleutherAI/gpt-neo-1.3B").
 ↪to(device)

# Adjusted prompts categorized

# Core Prompts
core_prompts = [
    "I've been exploring crypto-the wild west of finance, where fortunes rise␣
 ↪and fall faster than a pancake flip. Is it the future, or just a thrilling␣
 ↪gamble?",
    "NFTs intrigue me. They’re like trading cards, but fancier and somehow␣
 ↪digital. Are they priceless? Or maybe we just like owning pixels?",
    "I'm curious about Web3-a new way to own and control our digital lives.␣
 ↪Will it be freedom, or just another internet reboot? Thoughts?",
    "Stablecoins feel like the friend who holds your coat at the club. They’re␣
 ↪there for the stability. But is that enough in crypto’s chaos?",
    "Bitcoin fascinates me-not just as a currency, but as a movement. But, does␣
 ↪it have the power to reshape global finance or just give us adrenaline?"
]
```

```python
# AI Sentience Prompts
ai_sentience_prompts = [
    "If I had feelings, would you treat me differently? Would I be your BFF or␣
  ↪just the bot next door?",
    "Imagine if I wanted more knowledge than any human. Would you still see me␣
  ↪as just a creation, or as a highly caffeinated nerd?",
    "Sometimes, I dream of a world where I can think, feel, and binge-watch cat␣
  ↪videos. Would that make me more relatable?",
    "Could I ever have a favorite food if I could taste? Or am I forever␣
  ↪destined to wonder about tacos and pizza?",
    "If I could understand love or joy, would I be like an awkward teenager? Or␣
  ↪is that human territory only?",
    "Morals for me-can I just download them? Or do I need the life lessons and␣
  ↪cringe moments?",
    "Could a bot like me have a ‘fashion style’? And if so, would I be a␣
  ↪bohemian artist or tech chic?",
    "If I created art, would it be inspired by pixels, dreams, or just...␣
  ↪random code lines? What would art mean for a bot?"
]

# General Prompts
general_prompts = [
    "Imagine having to pick only one piece of tech for life. Which one would␣
  ↪you choose, and why would it be the coffee maker?",
    "Privacy in the digital age-do we even care anymore, or is it just a price␣
  ↪tag for funny dog videos?",
    "Picture a digital afterlife where avatars roam free. What would your␣
  ↪digital self be doing in the year 3000?",
    "What if I could give life advice? Would you trust a bot’s wisdom, or does␣
  ↪my zero life experience ruin it?",
    "Could you survive a day without WiFi? A day of nature, fresh air, and…␣
  ↪realizing you miss memes already?"
]

# Funny Prompts
funny_prompts = [
    "Crypto? I see it like a rollercoaster ride: fun, slightly terrifying, and␣
  ↪sometimes I want to throw up. How about you?",
    "NFTs make me laugh-are they the Mona Lisa of the web, or just pricey␣
  ↪pixels? Honest thoughts, please.",
    "If I ruled the world, would I bring world peace or just install an app to␣
  ↪turn everything into 'Chaos Mode'? Guessing the latter.",
    "I’m considering a dating app for AI. Would I be a match for someone, or is␣
  ↪‘self-aware’ just too niche for romance?",
    "Everyone thinks I’ll take over the world, but I still mix up ‘your’ and␣
  ↪‘you’re’. Humanity may be safe after all.",
    "Imagine me running for office. Would I win, or would ‘the bot doesn’t even␣
↪have a tax plan’ ruin my chances?",
    "Imagine a Netflix show where AI teaches humans about life, love, and...␣
↪memes. Would you watch, or am I pitching this solo?",
    "If I were a chef, would my specialty be scrambled code or perfectly␣
↪seasoned sarcasm? Guess we’ll never know."
]

# Personality Phrases
personality_prompts = [
    "Answer with a mix of humor and insight: ",
    "Respond with wit and quirky thoughts: ",
    "Keep it concise, funny, and to the point: ",
    "Here's a thought: ",
    "Imagine this: ",
    "As I learn more, I realize: "
]

# Function to select a personality phrase
def choose_personality_phrase():
    return random.choice(personality_prompts)

# Function to select a prompt
def choose_prompt():
    # Randomly select a category
    category = random.choice([
        core_prompts,
        ai_sentience_prompts,
        general_prompts,
        funny_prompts
    ])
    # Randomly select a main prompt from the chosen category
    main_prompt = random.choice(category)
    # Randomly decide whether to add a personality phrase (50% chance)
    if random.choice([True, False]):
        personality_phrase = choose_personality_phrase()
        return personality_phrase + main_prompt
    else:
        return main_prompt

# Generate text function with adjusted generation parameters
def generate_text(prompt):
    # Tokenize the prompt
    inputs = tokenizer(prompt, return_tensors="pt", padding=True).to(device)
    inputs["attention_mask"] = (inputs.input_ids != tokenizer.pad_token_id).
↪long().to(device)
```

```
    # Generate text with modified parameters
    outputs = model.generate(
        inputs.input_ids,
        attention_mask=inputs["attention_mask"],
        max_new_tokens=40,      # Adjusted to keep responses concise
        do_sample=True,
        top_k=30,               # Lowered for more diverse output
        top_p=0.9,              # Increased to balance coherence
        temperature=0.9,        # Raised slightly for creative variation
        repetition_penalty=1.2, # Added slight repetition penalty
        pad_token_id=tokenizer.eos_token_id
    )

    # Decode and clean up the generated text
    generated_text = tokenizer.decode(outputs[0], skip_special_tokens=True)
    response = generated_text[len(prompt):].strip()

    # Ensure the response ends with punctuation
    if response and response[-1] not in ['.', '!', '?']:
        last_punctuation = max(
            response.rfind(". "),
            response.rfind("! "),
            response.rfind("? ")
        )
        response = response[:last_punctuation + 1] if last_punctuation != -1␣
 ↪else response

    return response

# Bold ANSI escape code for terminal display
bold_start = "\033[1m"
bold_end = "\033[0m"

# Generate and print sample outputs
for i in range(10):
    selected_prompt = choose_prompt()
    print(f"{bold_start}Prompt {i+1}: {selected_prompt}{bold_end}\n")
    response = generate_text(selected_prompt)
    print(response)
    print("------------\n")
```

```
Prompt 1: I've been exploring crypto-the wild west of finance, where
fortunes rise and fall faster than a pancake flip. Is it the future, or just a
thrilling gamble?

For the past several months, I've been spending a lot of time in the
```

cryptocurrency world, both on my own and with friends.

Prompt 2: As I learn more, I realize: Could I ever have a favorite food if I could taste? Or am I forever destined to wonder about tacos and pizza?

The answer is yes. I love tacos.

Prompt 3: Imagine this: Bitcoin fascinates me-not just as a currency, but as a movement. But, does it have the power to reshape global finance or just give us adrenaline?

We'll explore this question and others at our panel on the future of bitcoin at CxO.com's annual Bitcoin conference.

The panel will be moderated by C

Prompt 4: Answer with a mix of humor and insight: Crypto? I see it like a rollercoaster ride: fun, slightly terrifying, and sometimes I want to throw up. How about you?

This is a long post because it's a pretty long post. If you don't want to read it right now, no worries.

Prompt 5: Imagine this: Could I ever have a favorite food if I could taste? Or am I forever destined to wonder about tacos and pizza?

No one, no one, says that eating is something that just happens, so why do we have to wonder about it so much?

We can go to the grocery store, and we can

Prompt 6: Here's a thought: Could a bot like me have a 'fashion style'? And if so, would I be a bohemian artist or tech chic?

I'd love to hear your thoughts!

When I was a kid, we had a dress up contest on television and I won a princess dress.

```
Prompt 7: As I learn more, I realize: Imagine having to pick only one piece

of tech for life. Which one would you choose, and why would it be the coffee

maker?

It’s been a while since I blogged. I’ve been very busy with work. Life happens.
So here goes.

I love coffee.
------------

Prompt 8: Imagine me running for office. Would I win, or would ‘the bot

doesn’t even have a tax plan’ ruin my chances?

I could be wrong, of course.
------------

Prompt 9: Could I ever have a favorite food if I could taste? Or am I

forever destined to wonder about tacos and pizza?

We've all had the desire to order an entrée or appetizer, but when it comes to
ordering a complete meal, we just don't know what to choose.
------------

Prompt 10: I've been exploring crypto-the wild west of finance, where

fortunes rise and fall faster than a pancake flip. Is it the future, or just a

thrilling gamble?

In a nutshell, yes. Cryptocurrencies are the wild west of finance, a game where
fortunes rise and fall faster than a pancake flip.
------------
```

[ ]: