```python
from google.colab import drive
drive.mount('/content/drive')
```

```python
!pip install transformers datasets peft accelerate
```

```python
from transformers import AutoTokenizer, AutoModelForCausalLM
import torch

base_model = "HuggingFaceTB/SmolLM2-360M-Instruct"

tokenizer = AutoTokenizer.from_pretrained(base_model)
tokenizer.pad_token = tokenizer.eos_token

model = AutoModelForCausalLM.from_pretrained(
    base_model,
    torch_dtype=torch.float16,
    device_map="auto"
)
```

```python
import json
from datasets import Dataset

file_path = "/content/drive/MyDrive/Datasets/sorachio-identity-dataset.jsonl"

data = []
with open(file_path, "r") as f:
    for idx, line in enumerate(f, start=1):
        try:
            data.append(json.loads(line))
        except json.JSONDecodeError as e:
            print(f"❌ JSON decode error on line {idx}: {e}")
            print("📄 Line content:")
            print(line)
            break

dataset = Dataset.from_list(data)

print(dataset[:2])

```

```python
from torch.nn import Linear

for name, module in model.named_modules():
    if isinstance(module, Linear):
      print(name)
```

```python
from peft import LoraConfig, get_peft_model, prepare_model_for_kbit_training

lora_config = LoraConfig(
    r=16,
    lora_alpha=32,
    target_modules=["q_proj", "v_proj", "k_proj", "o_proj", "gate_proj", "up_proj", "down_proj"],
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM"
)

model = get_peft_model(model, lora_config)
model.print_trainable_parameters()
```

```python
def format_prompt(example):
    # Memisahkan berdasarkan <|im_start|> dan <|im_end|>
    text = example["text"]
    messages = []

    # Memisahkan berdasarkan token yang ada
    parts = text.split("<|im_end|>")
    for part in parts:
        if "<|im_start|>system" in part:
            messages.append({"role": "system", "content": part.replace("<|im_start|>system\n", "").strip()})
        elif "<|im_start|>user" in part:
```

```
            messages.append({"role": "user", "content": part.replace("<|im_start|>user\n", "").strip()})
        elif "<|im_start|>assistant" in part:
            messages.append({"role": "assistant", "content": part.replace("<|im_start|>assistant\n", "").strip()})

    return tokenizer.apply_chat_template(messages, tokenize=False)

def tokenize(example):
    return tokenizer(format_prompt(example), truncation=True, padding="max_length", max_length=256)

# Tokenisasi dataset
tokenized_dataset = dataset.map(tokenize)
```

```
from transformers import TrainingArguments, Trainer, DataCollatorForLanguageModeling

training_args = TrainingArguments(
    output_dir="./sorachio-lora",
    per_device_train_batch_size=4,
    gradient_accumulation_steps=2,
    num_train_epochs=3,
    learning_rate=2e-4,
    warmup_ratio=0.03,
    weight_decay=0.01,
    fp16=True,
    lr_scheduler_type="cosine",
    save_strategy="epoch",
    logging_steps=10,
    report_to="none"
)

data_collator = DataCollatorForLanguageModeling(tokenizer, mlm=False)

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_dataset,
    data_collator=data_collator
)

trainer.train()
```

```
peft_model = model
model = model.merge_and_unload()
output_path = "/content/drive/MyDrive/Sorachio-360M-Chat/models/"

model.save_pretrained(output_path)
tokenizer.save_pretrained(output_path)
```