```
!pip install transformers datasets --quiet

import os
os.environ["WANDB_DISABLED"] = "true"
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 480.6/480.6 kB 19.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 116.3/116.3 kB 9.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 179.3/179.3 kB 16.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 134.8/134.8 kB 11.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.1/194.1 kB 17.1 MB/s eta 0:00:00
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the sou
gcsfs 2024.10.0 requires fsspec==2024.10.0, but you have fsspec 2024.9.0 which is incompatible.
```

```
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM, Trainer, TrainingArguments
from datasets import load_dataset
import pandas as pd
import torch


if torch.cuda.is_available():
    print(f"Using GPU: {torch.cuda.get_device_name(0)}")
else:
    print("CUDA not available. Using CPU.")
```

```
Using GPU: Tesla T4
```

```
dataset_path = "tamil_slang_large_dataset.tsv"
data = pd.read_csv(dataset_path, sep="\t", header=0, names=["Text", "Normalized Text"])

train_data = data.sample(frac=0.8, random_state=42)
val_data = data.drop(train_data.index)
train_data.to_json("train.json", orient="records", lines=True)
val_data.to_json("valid.json", orient="records", lines=True)


dataset = load_dataset("json", data_files={"train": "train.json", "validation": "valid.json"})
```

```
Generating train split:        8000/0 [00:00<00:00, 120461.51 examples/s]

Generating validation split:        2000/0 [00:00<00:00, 66159.87 examples/s]
```

```
tokenizer = AutoTokenizer.from_pretrained("facebook/mbart-large-50")

def preprocess_function(examples):
    inputs = tokenizer(
        examples["Text"], max_length=32, truncation=True, padding="max_length"
    )
    targets = tokenizer(
        examples["Normalized Text"], max_length=32, truncation=True, padding="max_length"
    )
    inputs["labels"] = targets["input_ids"]
    return inputs

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

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as s
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
tokenizer_config.json: 100%                                  531/531 [00:00<00:00, 9.14kB/s]

config.json: 100%                                  1.42k/1.42k [00:00<00:00, 38.2kB/s]

sentencepiece.bpe.model: 100%                                  5.07M/5.07M [00:00<00:00, 31.2MB/s]

special_tokens_map.json: 100%                                  649/649 [00:00<00:00, 12.9kB/s]

Map: 100%                                  8000/8000 [00:01<00:00, 5381.31 examples/s]

Map: 100%                                  2000/2000 [00:00<00:00, 5692.16 examples/s]
```

```
model = AutoModelForSeq2SeqLM.from_pretrained("facebook/mbart-large-50").to("cuda")
```

```
pytorch_model.bin: 100%  2.44G/2.44G [00:17<00:00, 252MB/s]
generation_config.json: 100%  261/261 [00:00<00:00, 5.42kB/s]
```

```
training_args = TrainingArguments(
    output_dir="./results",
    evaluation_strategy="epoch",
    learning_rate=5e-5,
    per_device_train_batch_size=8,
    gradient_accumulation_steps=4,
    num_train_epochs=3,
    weight_decay=0.01,
    fp16=True,
    save_total_limit=2,
    save_steps=500,
    logging_dir="./logs",
    report_to="none",
)
```

```
/usr/local/lib/python3.10/dist-packages/transformers/training_args.py:1575: FutureWarning: `evaluation_strategy` is deprecated and v
  warnings.warn(
```

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

```
<ipython-input-9-8a438e11e626>:1: FutureWarning: `tokenizer` is deprecated and will be removed in version 5.0.0 for `Trainer.__init_
  trainer = Trainer(
```

```
trainer.train()
```

[750/750 15:18, Epoch 3/3]

Epoch	Training Loss	Validation Loss
1	No log	0.000030
2	3.640100	0.000013
3	3.640100	0.000011

```
/usr/local/lib/python3.10/dist-packages/transformers/modeling_utils.py:2817: UserWarning: Moving the following attributes in the cor
  warnings.warn(
TrainOutput(global_step=750, training_loss=2.4267450989658634, metrics={'train_runtime': 922.177, 'train_samples_per_second':
26.025, 'train_steps_per_second': 0.813, 'total_flos': 1625347325952000.0, 'train_loss': 2.4267450989658634, 'epoch': 3.0})
```

```
model.save_pretrained("./tamil_slang_mbart")
tokenizer.save_pretrained("./tamil_slang_mbart")
```

```
('./tamil_slang_mbart/tokenizer_config.json',
 './tamil_slang_mbart/special_tokens_map.json',
 './tamil_slang_mbart/sentencepiece.bpe.model',
 './tamil_slang_mbart/added_tokens.json',
 './tamil_slang_mbart/tokenizer.json')
```

```
def normalize_text(input_text, model, tokenizer):
    inputs = tokenizer(input_text, return_tensors="pt", max_length=32, truncation=True).to("cuda")
    outputs = model.generate(inputs["input_ids"])
    return tokenizer.decode(outputs[0], skip_special_tokens=True)
```

```
test_text = "சரி செம்ம"
normalized_output = normalize_text(test_text, model, tokenizer)
print(f"Normalized Output: {normalized_output}")
```

```
Normalized Output: மிகவும் நல்லது
```

```
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM
model_dir = "./tamil_slang_mbart"
model = AutoModelForSeq2SeqLM.from_pretrained(model_dir).to("cuda")
tokenizer = AutoTokenizer.from_pretrained(model_dir)

def normalize_text(input_text, model, tokenizer):
    inputs = tokenizer(input_text, return_tensors="pt", max_length=32, truncation=True).to("cuda")
```

```
    outputs = model.generate(inputs["input_ids"])
    return tokenizer.decode(outputs[0], skip_special_tokens=True)

sample_texts = [
    "சரி செம்ம",
    "அப்போ செம்ம பாஸ்",
    "ஆமா கண்டிப்பா சரியா",
    "ஆமா அய்யோ டா",
    "சந்தோஷம் பண்ணிடுவோம்",
    "கேவலமாக சரியா"
]

for slang in sample_texts:
    normalized = normalize_text(slang, model, tokenizer)
    print(f"Input: {slang} -> Normalized: {normalized}")
```

```
Input: சரி செம்ம -> Normalized: மிகவும் நல்லது
Input: அப்போ செம்ம பாஸ் -> Normalized: மிகவும் நல்லது
Input: ஆமா கண்டிப்பா சரியா -> Normalized: மிகவும் உறுதியாக
Input: ஆமா அய்யோ டா -> Normalized: விசமம்
Input: சந்தோஷம் பண்ணிடுவோம் -> Normalized: மகிழ்ச்சி நாங்கள் செய்வோம்
Input: கேவலமாக சரியா -> Normalized: மிகவும் வலமாக
```

Start coding or generate with AI.