bgayvrkbn

March 29, 2025

0.0.1 Download Data

0.0.2 Downloading the Test Set

[ ]:
```
!gdown 1ewIzzKVP396I7TU4HxP_z9KhR_wsSF3e
```

```
Downloading…
From: https://drive.google.com/uc?id=1ewIzzKVP396I7TU4HxP_z9KhR_wsSF3e
To: /content/321.zip
100% 123k/123k [00:00<00:00, 130MB/s]
```

[ ]:
```
!gdown 1XpDugI3grTw4eWBxetcYSrDT1wLJN0at
```

```
Downloading…
From (original):
https://drive.google.com/uc?id=1XpDugI3grTw4eWBxetcYSrDT1wLJN0at
From (redirected): https://drive.google.com/uc?id=1XpDugI3grTw4eWBxetcYSrDT1wLJN
0at&confirm=t&uuid=d610672b-9b68-4ab0-b82a-aa965d1f89f8
To: /content/123.zip
100% 73.8M/73.8M [00:00<00:00, 169MB/s]
```

[ ]:
```
!unzip /content/123.zip
```

```
Archive:  /content/123.zip
 extracting: Test sources/Buendia - Instruccion.pdf
 extracting: Test sources/These files for HANDWRITTEN test ONLY/ES-AHPHU -
J-000312-0014    1579.pdf
 extracting: Test sources/These files for HANDWRITTEN test
ONLY/J&#x3a;0017&#x3a;03-J&#x3a;0085&#x3a;11    1799-1845.pdf
 extracting: Test sources/Constituciones sinodales Calahorra 1602.pdf
 extracting: Test sources/Ezcaray - Vozes.pdf
 extracting: Test sources/Mendo - Principe perfecto.pdf
 extracting: Test sources/Paredes - Reglas generales.pdf
 extracting: Test sources/PORCONES.228.35    1636.pdf
```

[ ]:
```
!unzip /content/321.zip
```

```
Archive:  /content/321.zip
 extracting: Test transcriptions/Buendia transcription.docx
```

```
 extracting: Test transcriptions/Constituciones sinodales transcription.docx
 extracting: Test transcriptions/Ezcaray transcription.docx
 extracting: Test transcriptions/Mendo transcription.docx
 extracting: Test transcriptions/Paredes transcription.docx
 extracting: Test transcriptions/PORCONES.228.35 1636 transcription.docx
```

0.0.3 Downloading the Train Set

This Train Set is from IIT_ISM AI-of-God 3.0 ML Challenge Problem Set. I participated in it on kaggle and got a WER of 0.116 and won the challenge in 2024.

[ ]:
```
!gdown 1aBM0-Pt7Bw_D7t3NzoACsm5aD3ws9CzI
```

```
Downloading…
From (original):
https://drive.google.com/uc?id=1aBM0-Pt7Bw_D7t3NzoACsm5aD3ws9CzI
From (redirected): https://drive.google.com/uc?id=1aBM0-Pt7Bw_D7t3NzoACsm5aD3ws9
CzI&confirm=t&uuid=b72e1fe1-5773-4893-a035-392afe6f833d
To: /content/ai-of-god-3.zip
100% 413M/413M [00:06<00:00, 59.2MB/s]
```

[ ]:
```
!unzip ai-of-god-3.zip
```

0.0.4 Load the pre-trained weights(saved on gdrive)

[ ]:
```
# TrOCR
!gdown 1-0zzez0RRwcCF3LZS3SByDMcCPmDpqtm
```

```
Downloading…
From (original):
https://drive.google.com/uc?id=1-0zzez0RRwcCF3LZS3SByDMcCPmDpqtm
From (redirected): https://drive.google.com/uc?id=1-0zzez0RRwcCF3LZS3SByDMcCPmDp
qtm&confirm=t&uuid=3bdf0616-0039-40a0-97e6-9fc3d3b7b29c
To: /content/checkpoint-1502.zip
100% 3.90G/3.90G [00:44<00:00, 86.6MB/s]
```

[ ]:
```
!unzip checkpoint-1502.zip -d checkpoint-1502
```

```
Archive:  checkpoint-1502.zip
  inflating: checkpoint-1502/scheduler.pt
  inflating: checkpoint-1502/scaler.pt
  inflating: checkpoint-1502/preprocessor_config.json
  inflating: checkpoint-1502/training_args.bin
  inflating: checkpoint-1502/model.safetensors
  inflating: checkpoint-1502/config.json
  inflating: checkpoint-1502/optimizer.pt
  inflating: checkpoint-1502/rng_state.pth
```

```
  inflating: checkpoint-1502/trainer_state.json
  inflating: checkpoint-1502/generation_config.json
```

[ ]:
```
#T5
!gdown 1k0W_ENJWBgNlWLiJ0Vck6ttTpa_3zk-G
```

```
Downloading…
From (original):
https://drive.google.com/uc?id=1k0W_ENJWBgNlWLiJ0Vck6ttTpa_3zk-G
From (redirected): https://drive.google.com/uc?id=1k0W_ENJWBgNlWLiJ0Vck6ttTpa_3z
k-G&confirm=t&uuid=db2700a9-2290-415a-ada3-e97c727b691c
To: /content/t5Model.zip
100% 823M/823M [00:10<00:00, 76.8MB/s]
```

[ ]:
```
!unzip t5Model.zip -d t5Model
```

```
Archive:  t5Model.zip
  inflating: t5Model/special_tokens_map.json
  inflating: t5Model/tokenizer_config.json
  inflating: t5Model/spiece.model
  inflating: t5Model/model.safetensors
  inflating: t5Model/tokenizer.json
  inflating: t5Model/config.json
  inflating: t5Model/generation_config.json
```

0.0.5 Import Necessary Libraries and Modules

[ ]:
```
import pandas as pd
import numpy as np

import os

import matplotlib.pyplot as plt
import random
from PIL import Image

import cv2
from heapq import heappush, heappop
from collections import defaultdict, Counter
from matplotlib import cm
from google.colab.patches import cv2_imshow
import io

import torch
from torch.utils.data import Dataset
from PIL import Image

from sklearn.model_selection import train_test_split
```

```
from transformers import default_data_collator
from transformers import Seq2SeqTrainer, Seq2SeqTrainingArguments
from transformers import VisionEncoderDecoderModel
from transformers import TrOCRProcessor

import random
from dataclasses import dataclass, field
from collections import defaultdict, Counter
```

0.0.6 FineTuning the Model TrOCR

[ ]:
```
df_train_pth = 'Public_data/train.csv'
df_train = pd.read_csv(df_train_pth)

df_test_pth = 'Public_data/test.csv'
df_test = pd.read_csv(df_test_pth)
```

The training samples correspond to handwritten samples.

[ ]:
```
def display_train_images(df_train, folder_path='Public_data/train_images',␣
↪num=9, n=3, rand=False):
    valid_extensions = ['.png', '.jpg', '.jpeg']

    if rand:
        indices = random.sample(range(len(df_train)), num)
    else:
        indices = list(range(num))

    fig, axes = plt.subplots(n, n, figsize=(15, 15))

    for i, idx in enumerate(indices):
        row = i // n
        col = i % n

        image_name = df_train['unique Id'][idx]
        transcription = df_train['transcription'][idx]

        if not any(image_name.endswith(ext) for ext in valid_extensions):
            image_name += '.png'

        image_path = os.path.join(folder_path, image_name)

        if os.path.exists(image_path):
            img = Image.open(image_path)
            axes[row, col].imshow(img)
            axes[row, col].set_title(transcription)
```

```
            axes[row, col].axis('off')

            for spine in axes[row, col].spines.values():
                spine.set_edgecolor('black')
                spine.set_linewidth(2)
        else:
            print(f"Image not found: {image_path}")

    plt.tight_layout()
    plt.show()

display_train_images(df_train, num=9, rand=False)
```

[ ]:
```
max_chars = df_train['transcription'].apply(len).max()
print(f"Maximum number of characters in 'transcription' column: {max_chars}")
```

```
Maximum number of characters in 'transcription' column: 74
```

[ ]:
```
train_df, test_df = train_test_split(df_train, test_size=0.2)
# we reset the indices to start from zero
train_df.reset_index(drop=True, inplace=True)
test_df.reset_index(drop=True, inplace=True)
```

[ ]:
```
class SpanishOldWritten(Dataset):
    def __init__(self, root_dir, df, processor, max_target_length=128):
```

```
        self.root_dir = root_dir
        self.df = df
        self.processor = processor
        self.max_target_length = max_target_length

    def __len__(self):
        return len(self.df)

    def __getitem__(self, idx):
        # get file name + text
        file_name = self.df['unique Id'][idx]
        text = self.df['transcription'][idx]
        # prepare image (i.e. resize + normalize)
        image_path = os.path.join(self.root_dir, file_name + ".png")
        image = Image.open(image_path).convert("RGB")
        pixel_values = self.processor(image, return_tensors="pt").pixel_values
        # add labels (input_ids) by encoding the text
        labels = self.processor.tokenizer(text,
                                          padding="max_length",
                                          max_length=self.max_target_length).
↪input_ids
        # important: make sure that PAD tokens are ignored by the loss function
        labels = [label if label != self.processor.tokenizer.pad_token_id else␣
↪-100 for label in labels]

        encoding = {"pixel_values": pixel_values.squeeze(), "labels": torch.
↪tensor(labels)}
        return encoding
```

We finetune TrOCR on the handwritten documents

[ ]:
```
processor = TrOCRProcessor.from_pretrained("microsoft/trocr-base-handwritten")
```

```
/usr/local/lib/python3.11/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 secret in your Google Colab
and restart your session.
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(
Using a slow image processor as `use_fast` is unset and a slow processor was
saved with this model. `use_fast=True` will be the default behavior in v4.50,
even if the model was saved with a slow processor. This will result in minor
differences in outputs. You'll still be able to use a slow processor with
`use_fast=False`.
```

[ ]:
```
train_dataset = SpanishOldWritten(root_dir='Public_data/train_images/',
                                  df=train_df,
                                  processor=processor)
eval_dataset = SpanishOldWritten(root_dir='Public_data/train_images/',
                                  df=test_df,
                                  processor=processor)
```

[ ]:
```
print("Number of training examples:", len(train_dataset))
print("Number of validation examples:", len(eval_dataset))
```

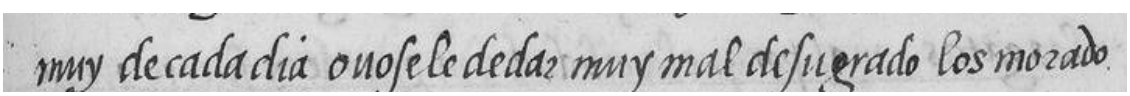
```
Number of training examples: 12008
Number of validation examples: 3002
```

[ ]:
```
encoding = train_dataset[0]
for k,v in encoding.items():
  print(k, v.shape)
```

```
pixel_values torch.Size([3, 384, 384])
labels torch.Size([128])
```

[ ]:
```
image = Image.open(train_dataset.root_dir + train_df['unique Id'][0] + ".png").
 ↪convert("RGB")
image
```

[ ]:



[ ]:
```
labels = encoding['labels']
labels[labels == -100] = processor.tokenizer.pad_token_id
label_str = processor.decode(labels, skip_special_tokens=True)
print(label_str)
```

```
muy de cada dia ouosele de dar muy mal de su grado los morado
```

[ ]:
```
model = VisionEncoderDecoderModel.from_pretrained("microsoft/trocr-base-stage1")
```

```
config.json:   0%|          | 0.00/4.21k [00:00<?, ?B/s]

model.safetensors:   0%|          | 0.00/1.54G [00:00<?, ?B/s]

Config of the encoder: <class 'transformers.models.vit.modeling_vit.ViTModel'>
is overwritten by shared encoder config: ViTConfig {
  "attention_probs_dropout_prob": 0.0,
  "encoder_stride": 16,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.0,
  "hidden_size": 768,
```

```
  "image_size": 384,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "model_type": "vit",
  "num_attention_heads": 12,
  "num_channels": 3,
  "num_hidden_layers": 12,
  "patch_size": 16,
  "pooler_act": "tanh",
  "pooler_output_size": 768,
  "qkv_bias": false,
  "torch_dtype": "float32",
  "transformers_version": "4.50.0"
}

Config of the decoder: <class
'transformers.models.trocr.modeling_trocr.TrOCRForCausalLM'> is overwritten by
shared decoder config: TrOCRConfig {
  "activation_dropout": 0.0,
  "activation_function": "relu",
  "add_cross_attention": true,
  "attention_dropout": 0.0,
  "bos_token_id": 0,
  "classifier_dropout": 0.0,
  "cross_attention_hidden_size": 768,
  "d_model": 1024,
  "decoder_attention_heads": 16,
  "decoder_ffn_dim": 4096,
  "decoder_layerdrop": 0.0,
  "decoder_layers": 12,
  "decoder_start_token_id": 2,
  "dropout": 0.1,
  "eos_token_id": 2,
  "init_std": 0.02,
  "is_decoder": true,
  "layernorm_embedding": false,
  "max_position_embeddings": 1024,
  "model_type": "trocr",
  "pad_token_id": 1,
  "scale_embedding": true,
  "tie_word_embeddings": false,
  "torch_dtype": "float32",
  "transformers_version": "4.50.0",
  "use_cache": false,
  "use_learned_position_embeddings": false,
  "vocab_size": 50265
}
```

```
Some weights of VisionEncoderDecoderModel were not initialized from the model
checkpoint at microsoft/trocr-base-stage1 and are newly initialized:
['encoder.pooler.dense.bias', 'encoder.pooler.dense.weight']
You should probably TRAIN this model on a down-stream task to be able to use it
for predictions and inference.
```

```
generation_config.json:   0%|          | 0.00/190 [00:00<?, ?B/s]
```

[ ]:
```
model.config.decoder_start_token_id = processor.tokenizer.cls_token_id
model.config.pad_token_id = processor.tokenizer.pad_token_id
model.config.vocab_size = model.config.decoder.vocab_size

model.config.eos_token_id = processor.tokenizer.sep_token_id
model.config.max_length = 80 # The max length is 74 (as in the above code block)
model.config.early_stopping = True
model.config.no_repeat_ngram_size = 3
model.config.length_penalty = 2.0
model.config.num_beams = 4
```

[ ]:
```
num_training_steps = 1000
half_steps = num_training_steps // 2

training_args = Seq2SeqTrainingArguments(
    predict_with_generate=True,
    evaluation_strategy="steps",
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    fp16=True,
    output_dir="AI3",
    logging_steps=2,
    save_steps=half_steps,
    save_total_limit=2,
    eval_steps=200,
    num_train_epochs=2
)
```

```
/usr/local/lib/python3.11/dist-packages/transformers/training_args.py:1611:
FutureWarning: `evaluation_strategy` is deprecated and will be removed in
version 4.46 of  Transformers. Use `eval_strategy` instead
  warnings.warn(
```

[ ]:
```
!pip install evaluate jiwer
```

```
Collecting evaluate
  Downloading evaluate-0.4.3-py3-none-any.whl.metadata (9.2 kB)
Collecting jiwer
  Downloading jiwer-3.1.0-py3-none-any.whl.metadata (2.6 kB)
```

```
Collecting datasets>=2.0.0 (from evaluate)
  Downloading datasets-3.4.1-py3-none-any.whl.metadata (19 kB)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-
packages (from evaluate) (2.0.2)
Collecting dill (from evaluate)
  Downloading dill-0.3.9-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages
(from evaluate) (2.2.2)
Requirement already satisfied: requests>=2.19.0 in
/usr/local/lib/python3.11/dist-packages (from evaluate) (2.32.3)
Requirement already satisfied: tqdm>=4.62.1 in /usr/local/lib/python3.11/dist-
packages (from evaluate) (4.67.1)
Collecting xxhash (from evaluate)
  Downloading
xxhash-3.5.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata
(12 kB)
Collecting multiprocess (from evaluate)
  Downloading multiprocess-0.70.17-py311-none-any.whl.metadata (7.2 kB)
Requirement already satisfied: fsspec>=2021.05.0 in
/usr/local/lib/python3.11/dist-packages (from fsspec[http]>=2021.05.0->evaluate)
(2025.3.0)
Requirement already satisfied: huggingface-hub>=0.7.0 in
/usr/local/lib/python3.11/dist-packages (from evaluate) (0.29.3)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-
packages (from evaluate) (24.2)
Requirement already satisfied: click>=8.1.8 in /usr/local/lib/python3.11/dist-
packages (from jiwer) (8.1.8)
Collecting rapidfuzz>=3.9.7 (from jiwer)
  Downloading rapidfuzz-3.12.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x
86_64.whl.metadata (12 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-
packages (from datasets>=2.0.0->evaluate) (3.18.0)
Requirement already satisfied: pyarrow>=15.0.0 in
/usr/local/lib/python3.11/dist-packages (from datasets>=2.0.0->evaluate)
(18.1.0)
Collecting dill (from evaluate)
  Downloading dill-0.3.8-py3-none-any.whl.metadata (10 kB)
Collecting multiprocess (from evaluate)
  Downloading multiprocess-0.70.16-py311-none-any.whl.metadata (7.2 kB)
Collecting fsspec>=2021.05.0 (from fsspec[http]>=2021.05.0->evaluate)
  Downloading fsspec-2024.12.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-
packages (from datasets>=2.0.0->evaluate) (3.11.14)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-
packages (from datasets>=2.0.0->evaluate) (6.0.2)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.7.0->evaluate)
(4.12.2)
```

```
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->evaluate)
(3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-
packages (from requests>=2.19.0->evaluate) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->evaluate)
(2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->evaluate)
(2025.1.31)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas->evaluate) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-
packages (from pandas->evaluate) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-
packages (from pandas->evaluate) (2025.1)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (2.6.1)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (1.3.2)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-
packages (from aiohttp->datasets>=2.0.0->evaluate) (25.3.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (1.5.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (6.2.0)
Requirement already satisfied: propcache>=0.2.0 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (0.3.0)
Requirement already satisfied: yarl<2.0,>=1.17.0 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets>=2.0.0->evaluate) (1.18.3)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-
packages (from python-dateutil>=2.8.2->pandas->evaluate) (1.17.0)
Downloading evaluate-0.4.3-py3-none-any.whl (84 kB)
                         84.0/84.0 kB
7.3 MB/s eta 0:00:00
Downloading jiwer-3.1.0-py3-none-any.whl (22 kB)
Downloading datasets-3.4.1-py3-none-any.whl (487 kB)
                         487.4/487.4 kB
34.5 MB/s eta 0:00:00
Downloading dill-0.3.8-py3-none-any.whl (116 kB)
                         116.3/116.3 kB
```

```
9.7 MB/s eta 0:00:00
Downloading fsspec-2024.12.0-py3-none-any.whl (183 kB)
                         183.9/183.9 kB
14.1 MB/s eta 0:00:00
Downloading multiprocess-0.70.16-py311-none-any.whl (143 kB)
                         143.5/143.5 kB
11.4 MB/s eta 0:00:00
Downloading
rapidfuzz-3.12.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.1
MB)
                         3.1/3.1 MB
89.9 MB/s eta 0:00:00
Downloading
xxhash-3.5.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (194 kB)
                         194.8/194.8 kB
14.4 MB/s eta 0:00:00
Installing collected packages: xxhash, rapidfuzz, fsspec, dill,
multiprocess, jiwer, datasets, evaluate
  Attempting uninstall: fsspec
    Found existing installation: fsspec 2025.3.0
    Uninstalling fsspec-2025.3.0:
      Successfully uninstalled fsspec-2025.3.0
```

```
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
torch 2.6.0+cu124 requires nvidia-cublas-cu12==12.4.5.8; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cublas-cu12
12.5.3.2 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-cupti-cu12==12.4.127; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cuda-cupti-cu12
12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-nvrtc-cu12==12.4.127; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cuda-nvrtc-cu12
12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-runtime-cu12==12.4.127; platform_system
== "Linux" and platform_machine == "x86_64", but you have nvidia-cuda-runtime-
cu12 12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cudnn-cu12==9.1.0.70; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cudnn-cu12
9.3.0.75 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cufft-cu12==11.2.1.3; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cufft-cu12
11.2.3.61 which is incompatible.
torch 2.6.0+cu124 requires nvidia-curand-cu12==10.3.5.147; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-curand-cu12
10.3.6.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cusolver-cu12==11.6.1.9; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cusolver-cu12
11.6.3.83 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cusparse-cu12==12.3.1.170; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-cusparse-cu12
12.5.1.3 which is incompatible.
torch 2.6.0+cu124 requires nvidia-nvjitlink-cu12==12.4.127; platform_system ==
"Linux" and platform_machine == "x86_64", but you have nvidia-nvjitlink-cu12
12.5.82 which is incompatible.
gcsfs 2025.3.0 requires fsspec==2025.3.0, but you have fsspec 2024.12.0 which is
incompatible.
Successfully installed datasets-3.4.1 dill-0.3.8 evaluate-0.4.3
```

```
fsspec-2024.12.0 jiwer-3.1.0 multiprocess-0.70.16 rapidfuzz-3.12.2 xxhash-3.5.0
```

We use the metrices `WER` to determine the accuracy of the model.

[ ]:
```
from jiwer import wer as jiwer_wer

def compute_metrics(pred):
    labels_ids = pred.label_ids
    pred_ids = pred.predictions

    pred_str = processor.batch_decode(pred_ids, skip_special_tokens=True)
    labels_ids[labels_ids == -100] = processor.tokenizer.pad_token_id
    label_str = processor.batch_decode(labels_ids, skip_special_tokens=True)

    wer_scores = [jiwer_wer(ref, hyp) for ref, hyp in zip(label_str, pred_str)]
    avg_wer = np.mean(wer_scores)
    return {"wer": avg_wer}
```

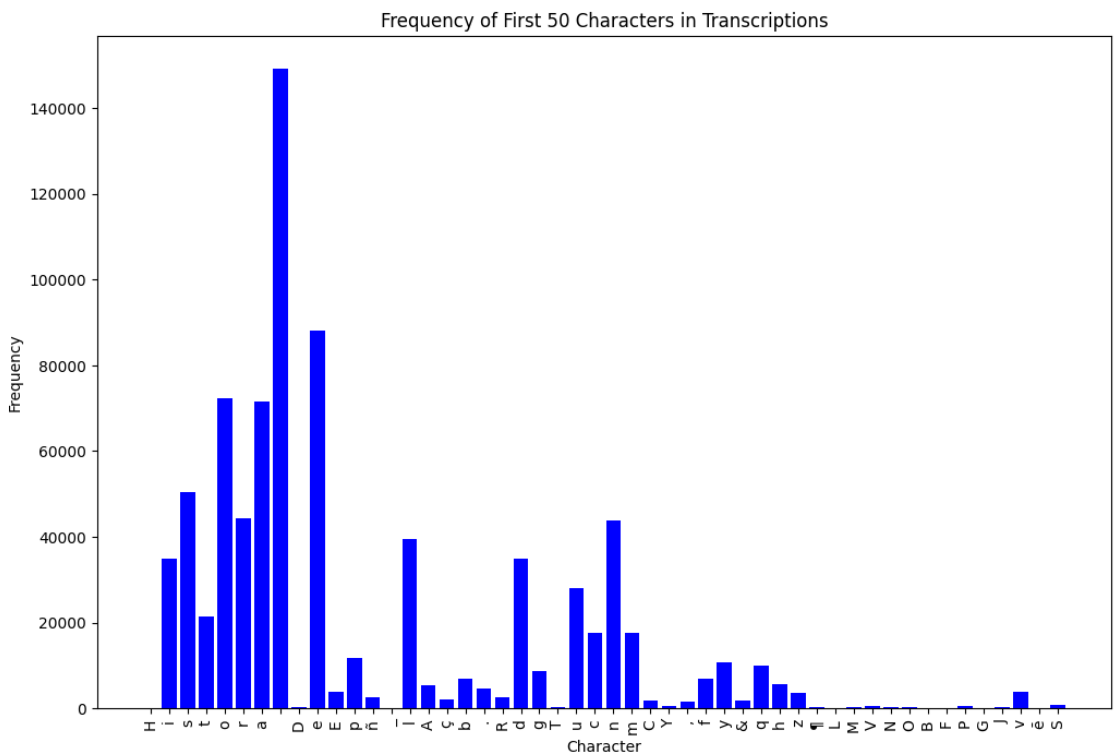
[ ]:
```
# instantiate trainer -> run to start finetuning

# trainer = Seq2SeqTrainer(
#     model=model,
#     tokenizer=processor.feature_extractor,
#     args=training_args,
#     compute_metrics=compute_metrics,
#     train_dataset=train_dataset,
#     eval_dataset=eval_dataset,
#     data_collator=default_data_collator,
# )
# trainer.train()
```

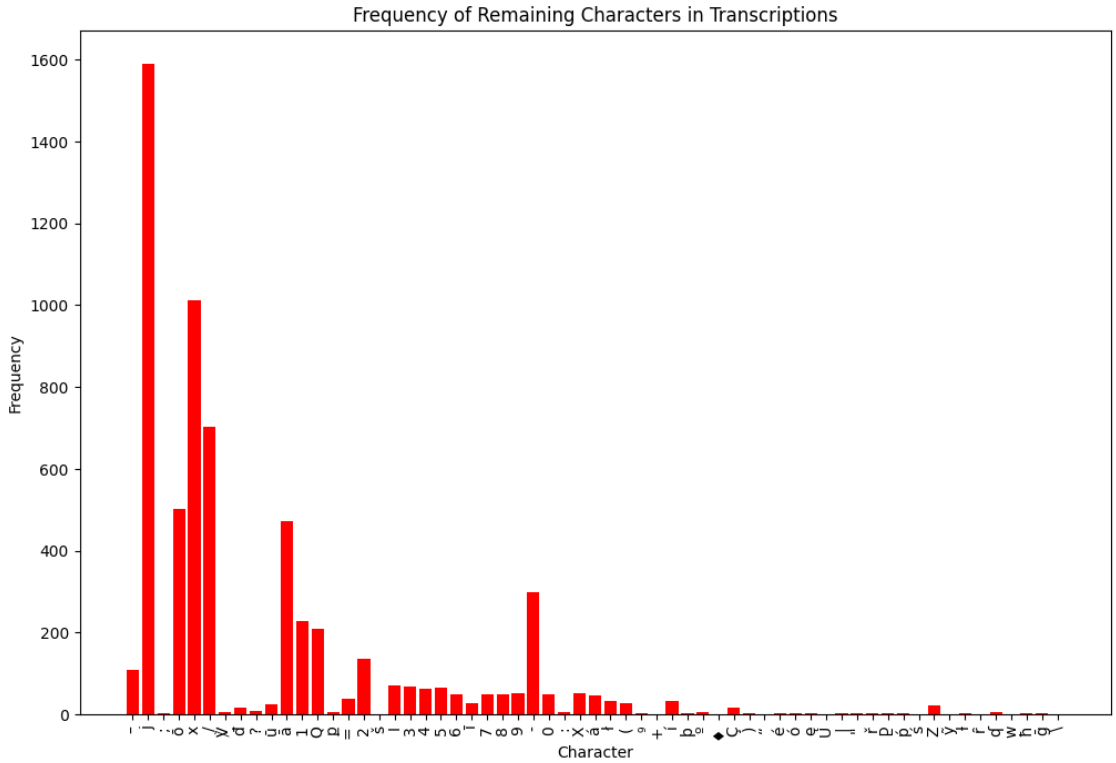
0.0.7 Finetuning the T5 to act as grammar correction

We will check the words occoring frequency in the train set

[ ]:
```
df_train = pd.read_csv('Public_data/train.csv')

df_train['char_count'] = df_train['transcription'].apply(len)

all_text = ''.join(df_train['transcription'])

char_count = Counter(all_text)

first_50_chars = dict(list(char_count.items())[:50])
remaining_chars = dict(list(char_count.items())[50:])

plt.figure(figsize=(12, 8))
plt.bar(first_50_chars.keys(), first_50_chars.values(), color='b')
```

```
plt.title('Frequency of First 50 Characters in Transcriptions')
plt.xlabel('Character')
plt.ylabel('Frequency')
plt.xticks(rotation=90)
plt.show()

plt.figure(figsize=(12, 8))
plt.bar(remaining_chars.keys(), remaining_chars.values(), color='r')
plt.title('Frequency of Remaining Characters in Transcriptions')
plt.xlabel('Character')
plt.ylabel('Frequency')
plt.xticks(rotation=90)
plt.show()
```





we calculated the order of occouring words, which will tell us in what order the characters in the words occour (what possible char can form that word), this will tell what word to remove for getiing a higher prob grammer checker,

From now on, we will train/finetune a T5(text-text) based model for error handling and grammer correction.

For this we need a dataset of bad spanish / good spanish, for this we used the training set text with modifications,

0. The model output test_predict.csv was checked and it detected 'n-accent' as 'A_', wrote a code to replace all 'A_' by 'n'.
1. Removed around 10% of characters, as our model was giving WER around 0.27 at this point of time, so removing 10% of random characters, also remove 15% occourence of 'a', 'e', 'n' in particular.

– This will take into account for the missing words, 'z', 'n-accent', 'q' and others.

2. Interchanging {(u,v), (f,s)}, removing u, removing v, removing both at random, do the same for f and s. This injection into the code is done at random, and it is only done for 50% of the time.

More changes to lower WER, and to get better results

3. (f,s) were interchanged in some of the places they occoured at random and (u, v) at lesser than (f, s).

change: (f,s):20%, (u,v):20%

4. Around 20% of the words at random were split from between with a '_' or ':' or '-'.

[ ]:
```
@dataclass
class Config:
    random_char_removal_percent: float = 0.10  # Remove 10% of random characters
    specific_char_removal: dict = field(default_factory=lambda: {})  # Removal␣
↪percentages for specific characters
    interchange_pairs: list = field(default_factory=lambda: [])      #␣
↪Interchanging character pairs
    interchange_prob: float = 0.20                                   # 20%␣
↪chance for each pair occurrence
    word_split_percent: float = 0.20                                 # Split␣
↪20% of the words at random
    word_split_separators: list = field(default_factory=lambda: [])    #␣
↪Separators for word splitting
    df_span_gram: pd.DataFrame = field(default_factory=pd.DataFrame)   # To␣
↪store the output DataFrame

config = Config(
    specific_char_removal={'a': 0.15, 'e': 0.15, 'n': 0.15},
    interchange_pairs=[('u', 'v'), ('f', 's')],
    word_split_separators=['_', ':', '-']
)

def remove_random_characters(text, percent):
    num_chars_to_remove = int(len(text) * percent)
    if num_chars_to_remove == 0:
        return text
    indices_to_remove = random.sample(range(len(text)), num_chars_to_remove)
    return ''.join([char for i, char in enumerate(text) if i not in␣
↪indices_to_remove])

def remove_specific_characters(text, char_dict):
    new_text = list(text)
    for char, percent in char_dict.items():
        char_indices = [i for i, c in enumerate(new_text) if c == char]
        num_chars_to_remove = int(len(char_indices) * percent)
        if num_chars_to_remove > 0:
            indices_to_remove = random.sample(char_indices, num_chars_to_remove)
            for i in indices_to_remove:
                new_text[i] = ''
    return ''.join(new_text)

def interchange_characters(text, interchange_pairs, probability):
    new_text = list(text)
```

```
    for i, char in enumerate(new_text):
        for pair in interchange_pairs:
            if char in pair and random.random() < probability:
                new_text[i] = pair[1] if char == pair[0] else pair[0]
    return ''.join(new_text)

def random_split_words(text, percent, separators):
    words = text.split()
    num_words_to_split = int(len(words) * percent)
    if num_words_to_split == 0:
        return text
    words_to_split = random.sample(range(len(words)), num_words_to_split)
    for i in words_to_split:
        if len(words[i]) > 1:
            split_index = random.randint(1, len(words[i]) - 1)
            separator = random.choice(separators)
            words[i] = words[i][:split_index] + separator +␣
 ↪words[i][split_index:]
    return ' '.join(words)

def apply_changes(df_train, config):
    corrections = []
    sentences = []

    for idx, transcription in enumerate(df_train['transcription']):
        original_text = transcription

        corrupted_text = remove_random_characters(original_text, config.
 ↪random_char_removal_percent)
        corrupted_text = remove_specific_characters(corrupted_text, config.
 ↪specific_char_removal)
        corrupted_text = interchange_characters(corrupted_text, config.
 ↪interchange_pairs, config.interchange_prob)
        corrupted_text = random_split_words(corrupted_text, config.
 ↪word_split_percent, config.word_split_separators)

        corrections.append(f"[ {corrupted_text} ]")
        sentences.append(original_text)

    config.df_span_gram['corrections'] = corrections
    config.df_span_gram['sentences'] = sentences
    return config.df_span_gram

df_train = pd.read_csv('Public_data/train.csv')
df_span_gram = apply_changes(df_train, config)
```

[ ]:
```
config.df_span_gram.to_csv('trainT5.csv', index=False)
```

[ ]:
```
!pip install happytransformer
```

```
Requirement already satisfied: happytransformer in
/usr/local/lib/python3.11/dist-packages (3.0.0)
Requirement already satisfied: torch>=1.0 in /usr/local/lib/python3.11/dist-
packages (from happytransformer) (2.6.0+cu124)
Requirement already satisfied: tqdm>=4.43 in /usr/local/lib/python3.11/dist-
packages (from happytransformer) (4.67.1)
Requirement already satisfied: transformers<5.0.0,>=4.30.1 in
/usr/local/lib/python3.11/dist-packages (from happytransformer) (4.50.0)
Requirement already satisfied: datasets<3.0.0,>=2.13.1 in
/usr/local/lib/python3.11/dist-packages (from happytransformer) (2.21.0)
Requirement already satisfied: sentencepiece in /usr/local/lib/python3.11/dist-
packages (from happytransformer) (0.2.0)
Requirement already satisfied: protobuf in /usr/local/lib/python3.11/dist-
packages (from happytransformer) (5.29.4)
Requirement already satisfied: accelerate<1.0.0,>=0.20.1 in
/usr/local/lib/python3.11/dist-packages (from happytransformer) (0.34.2)
Requirement already satisfied: tokenizers<1.0.0,>=0.13.3 in
/usr/local/lib/python3.11/dist-packages (from happytransformer) (0.21.1)
Requirement already satisfied: wandb in /usr/local/lib/python3.11/dist-packages
(from happytransformer) (0.19.8)
Requirement already satisfied: numpy<3.0.0,>=1.17 in
/usr/local/lib/python3.11/dist-packages (from
accelerate<1.0.0,>=0.20.1->happytransformer) (2.0.2)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.11/dist-packages (from
accelerate<1.0.0,>=0.20.1->happytransformer) (24.2)
Requirement already satisfied: psutil in /usr/local/lib/python3.11/dist-packages
(from accelerate<1.0.0,>=0.20.1->happytransformer) (5.9.5)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.11/dist-packages
(from accelerate<1.0.0,>=0.20.1->happytransformer) (6.0.2)
Requirement already satisfied: huggingface-hub>=0.21.0 in
/usr/local/lib/python3.11/dist-packages (from
accelerate<1.0.0,>=0.20.1->happytransformer) (0.29.3)
Requirement already satisfied: safetensors>=0.4.3 in
/usr/local/lib/python3.11/dist-packages (from
accelerate<1.0.0,>=0.20.1->happytransformer) (0.5.3)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-
packages (from datasets<3.0.0,>=2.13.1->happytransformer) (3.18.0)
Requirement already satisfied: pyarrow>=15.0.0 in
/usr/local/lib/python3.11/dist-packages (from
datasets<3.0.0,>=2.13.1->happytransformer) (18.1.0)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in
/usr/local/lib/python3.11/dist-packages (from
datasets<3.0.0,>=2.13.1->happytransformer) (0.3.8)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages
```

```
(from datasets<3.0.0,>=2.13.1->happytransformer) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in
/usr/local/lib/python3.11/dist-packages (from
datasets<3.0.0,>=2.13.1->happytransformer) (2.32.3)
Requirement already satisfied: xxhash in /usr/local/lib/python3.11/dist-packages
(from datasets<3.0.0,>=2.13.1->happytransformer) (3.5.0)
Requirement already satisfied: multiprocess in /usr/local/lib/python3.11/dist-
packages (from datasets<3.0.0,>=2.13.1->happytransformer) (0.70.16)
Requirement already satisfied: fsspec<=2024.6.1,>=2023.1.0 in
/usr/local/lib/python3.11/dist-packages (from
fsspec[http]<=2024.6.1,>=2023.1.0->datasets<3.0.0,>=2.13.1->happytransformer)
(2024.6.1)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-
packages (from datasets<3.0.0,>=2.13.1->happytransformer) (3.11.14)
Requirement already satisfied: typing-extensions>=4.10.0 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(4.12.2)
Requirement already satisfied: networkx in /usr/local/lib/python3.11/dist-
packages (from torch>=1.0->happytransformer) (3.4.2)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages
(from torch>=1.0->happytransformer) (3.1.6)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.4.127 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(12.4.127)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.4.127 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(12.4.127)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.4.127 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(12.4.127)
Requirement already satisfied: nvidia-cudnn-cu12==9.1.0.70 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(9.1.0.70)
Requirement already satisfied: nvidia-cublas-cu12==12.4.5.8 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(12.4.5.8)
Requirement already satisfied: nvidia-cufft-cu12==11.2.1.3 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(11.2.1.3)
Requirement already satisfied: nvidia-curand-cu12==10.3.5.147 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(10.3.5.147)
Requirement already satisfied: nvidia-cusolver-cu12==11.6.1.9 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(11.6.1.9)
Requirement already satisfied: nvidia-cusparse-cu12==12.3.1.170 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(12.3.1.170)
```

```
Requirement already satisfied: nvidia-cusparselt-cu12==0.6.2 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(0.6.2)
Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(2.21.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(12.4.127)
Requirement already satisfied: nvidia-nvjitlink-cu12==12.4.127 in
/usr/local/lib/python3.11/dist-packages (from torch>=1.0->happytransformer)
(12.4.127)
Requirement already satisfied: triton==3.2.0 in /usr/local/lib/python3.11/dist-
packages (from torch>=1.0->happytransformer) (3.2.0)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.11/dist-
packages (from torch>=1.0->happytransformer) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/usr/local/lib/python3.11/dist-packages (from
sympy==1.13.1->torch>=1.0->happytransformer) (1.3.0)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.11/dist-packages (from
transformers<5.0.0,>=4.30.1->happytransformer) (2024.11.6)
Requirement already satisfied: click!=8.0.0,>=7.1 in
/usr/local/lib/python3.11/dist-packages (from wandb->happytransformer) (8.1.8)
Requirement already satisfied: docker-pycreds>=0.4.0 in
/usr/local/lib/python3.11/dist-packages (from wandb->happytransformer) (0.4.0)
Requirement already satisfied: gitpython!=3.1.29,>=1.0.0 in
/usr/local/lib/python3.11/dist-packages (from wandb->happytransformer) (3.1.44)
Requirement already satisfied: platformdirs in /usr/local/lib/python3.11/dist-
packages (from wandb->happytransformer) (4.3.7)
Requirement already satisfied: pydantic<3,>=2.6 in
/usr/local/lib/python3.11/dist-packages (from wandb->happytransformer) (2.10.6)
Requirement already satisfied: sentry-sdk>=2.0.0 in
/usr/local/lib/python3.11/dist-packages (from wandb->happytransformer) (2.24.0)
Requirement already satisfied: setproctitle in /usr/local/lib/python3.11/dist-
packages (from wandb->happytransformer) (1.3.5)
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-
packages (from wandb->happytransformer) (75.1.0)
Requirement already satisfied: six>=1.4.0 in /usr/local/lib/python3.11/dist-
packages (from docker-pycreds>=0.4.0->wandb->happytransformer) (1.17.0)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets<3.0.0,>=2.13.1->happytransformer) (2.6.1)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets<3.0.0,>=2.13.1->happytransformer) (1.3.2)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-
packages (from aiohttp->datasets<3.0.0,>=2.13.1->happytransformer) (25.3.0)
```

```
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets<3.0.0,>=2.13.1->happytransformer) (1.5.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets<3.0.0,>=2.13.1->happytransformer) (6.2.0)
Requirement already satisfied: propcache>=0.2.0 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets<3.0.0,>=2.13.1->happytransformer) (0.3.0)
Requirement already satisfied: yarl<2.0,>=1.17.0 in
/usr/local/lib/python3.11/dist-packages (from
aiohttp->datasets<3.0.0,>=2.13.1->happytransformer) (1.18.3)
Requirement already satisfied: gitdb<5,>=4.0.1 in
/usr/local/lib/python3.11/dist-packages (from
gitpython!=3.1.29,>=1.0.0->wandb->happytransformer) (4.0.12)
Requirement already satisfied: annotated-types>=0.6.0 in
/usr/local/lib/python3.11/dist-packages (from
pydantic<3,>=2.6->wandb->happytransformer) (0.7.0)
Requirement already satisfied: pydantic-core==2.27.2 in
/usr/local/lib/python3.11/dist-packages (from
pydantic<3,>=2.6->wandb->happytransformer) (2.27.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from
requests>=2.32.2->datasets<3.0.0,>=2.13.1->happytransformer) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-
packages (from requests>=2.32.2->datasets<3.0.0,>=2.13.1->happytransformer)
(3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from
requests>=2.32.2->datasets<3.0.0,>=2.13.1->happytransformer) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.11/dist-packages (from
requests>=2.32.2->datasets<3.0.0,>=2.13.1->happytransformer) (2025.1.31)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.11/dist-packages (from
jinja2->torch>=1.0->happytransformer) (3.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from
pandas->datasets<3.0.0,>=2.13.1->happytransformer) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-
packages (from pandas->datasets<3.0.0,>=2.13.1->happytransformer) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-
packages (from pandas->datasets<3.0.0,>=2.13.1->happytransformer) (2025.1)
Requirement already satisfied: smmap<6,>=3.0.1 in
/usr/local/lib/python3.11/dist-packages (from
gitdb<5,>=4.0.1->gitpython!=3.1.29,>=1.0.0->wandb->happytransformer) (5.0.2)
```

[ ]:
```python
from happytransformer import HappyTextToText, TTSettings
```

[ ]:
```python
happy_tt = HappyTextToText("T5", "vennify/t5-base-grammar-correction")
```

[ ]:
```python
args = TTSettings(num_beams=4, min_length=1)
```

[ ]:
```python
import csv
```

[ ]:
```python
def generate_csv(csv_path, df_span_gram):
    with open(csv_path, 'w', newline='') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(["input", "target"])
        # Iterate over each row in df_span_gram
        for idx, row in df_span_gram.iterrows():
            # Add prefix "grammar: " to input (sentence)
            input_text = "grammar: " + row["sentences"]
            # Correction column is already in the desired format
            correction_text = row["corrections"]
            # Write to CSV if input and correction are not blank
            if input_text and correction_text:
                writer.writerow([input_text, correction_text])

generate_csv("train_val_gram.csv", df_span_gram)
```

[ ]:
```python
from happytransformer import TTTrainArgs

# run -> train your own grammar correction model

# args = TTTrainArgs(batch_size=8)
# happy_tt.train("train_val_gram.csv", args=args)
```

0.0.8 Using the A* Line Segmentation

[ ]:
```python
class AStar:
    def __init__(self, binary_image, start, end, mask=None):
        self.binary = binary_image
        self.start = start
        self.end = end
        self.mask = mask if mask is not None else np.ones_like(binary_image)
        self.height, self.width = binary_image.shape

    def heuristic(self, a, b):
        return abs(a[0] - b[0]) + abs(a[1] - b[1])

    def get_neighbors(self, node):
        i, j = node
        neighbors = []
```

```
        directions = [(0, -1), (0, 1), (-1, 0), (1, 0), (-1, -1), (-1, 1), (1,␣
↪-1), (1, 1)]
        weights = [1, 1, 3, 3, 4, 4, 4, 4]

        for idx, (di, dj) in enumerate(directions):
            ni, nj = i + di, j + dj
            if 0 <= ni < self.height and 0 <= nj < self.width and self.mask[ni,␣
↪nj]:
                cost = weights[idx] * (1 if self.binary[ni, nj] > 0 else 10)
                neighbors.append(((ni, nj), cost))

        return neighbors

    def find_path(self):
        open_set = []
        heappush(open_set, (0, self.start))

        came_from = {}
        g_score = defaultdict(lambda: float('inf'))
        g_score[self.start] = 0

        f_score = defaultdict(lambda: float('inf'))
        f_score[self.start] = self.heuristic(self.start, self.end)

        open_set_hash = {self.start}

        while open_set:
            _, current = heappop(open_set)
            open_set_hash.remove(current)

            if current == self.end:
                path = []
                while current in came_from:
                    path.append(current)
                    current = came_from[current]
                path.append(self.start)
                return path[::-1]

            for neighbor, cost in self.get_neighbors(current):
                tentative_g_score = g_score[current] + cost

                if tentative_g_score < g_score[neighbor]:
                    came_from[neighbor] = current
                    g_score[neighbor] = tentative_g_score
                    f_score[neighbor] = tentative_g_score + self.
↪heuristic(neighbor, self.end)
```

```
                    if neighbor not in open_set_hash:
                        heappush(open_set, (f_score[neighbor], neighbor))
                        open_set_hash.add(neighbor)

        return None

def preprocess_image(image):
    if len(image.shape) == 3:
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    else:
        gray = image

    binary = cv2.adaptiveThreshold(gray, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
                                   cv2.THRESH_BINARY_INV, 15, 5)

    kernel = np.ones((2, 2), np.uint8)
    binary = cv2.morphologyEx(binary, cv2.MORPH_OPEN, kernel)

    return gray, binary

def compute_projection_profile(binary):
    return np.sum(binary, axis=1)

def detect_text_lines(binary, projection, min_height=5, min_gap=10):
    height = binary.shape[0]
    threshold = np.mean(projection[projection > 0]) * 0.3

    line_regions = []
    in_line = False
    start = 0

    for i in range(height):
        if not in_line and projection[i] > threshold:
            in_line = True
            start = i
        elif in_line and (projection[i] <= threshold or i == height - 1):
            if i - start >= min_height:
                line_regions.append((start, i))
            in_line = False

    merged_regions = []
    if not line_regions:
        return []

    current = line_regions[0]
```

```
    for next_region in line_regions[1:]:
        if next_region[0] - current[1] < min_gap:
            current = (current[0], next_region[1])
        else:
            merged_regions.append(current)
            current = next_region

    merged_regions.append(current)

    return merged_regions

def extract_line_paths(binary, line_regions):
    height, width = binary.shape
    line_paths = []

    for y_start, y_end in line_regions:
        line_mask = np.zeros_like(binary)
        margin = int((y_end - y_start) * 0.2) + 2
        y_min = max(0, y_start - margin)
        y_max = min(height, y_end + margin)
        line_mask[y_min:y_max, :] = 1

        line_img = binary * line_mask

        col_projection = np.sum(line_img, axis=0)
        text_cols = np.where(col_projection > 0)[0]

        if len(text_cols) < 2:
            continue

        mid_y = (y_start + y_end) // 2
        left_x = text_cols[0]
        right_x = text_cols[-1]

        left_region = line_img[y_start:y_end, left_x:left_x+20]
        right_region = line_img[y_start:y_end, max(0, right_x-20):right_x+1]

        left_y = y_start + np.argmax(np.sum(left_region, axis=1))
        right_y = y_start + np.argmax(np.sum(right_region, axis=1))

        start = (left_y, left_x)
        end = (right_y, right_x)

        astar = AStar(binary, start, end, line_mask)
        path = astar.find_path()

        if path:
```

```
            line_paths.append((path, (y_start, y_end)))

    return line_paths

def extract_line_images(original, binary, line_paths, padding=5):
    line_images = []

    for path, (y_start, y_end) in line_paths:
        path_points = np.array(path)
        min_y = max(0, min(y_start, np.min(path_points[:, 0])) - padding)
        max_y = min(binary.shape[0], max(y_end, np.max(path_points[:, 0])) +␣
 ↪padding)
        min_x = max(0, np.min(path_points[:, 1]) - padding)
        max_x = min(binary.shape[1], np.max(path_points[:, 1]) + padding)

        line_img = original[min_y:max_y, min_x:max_x].copy()

        line_images.append(line_img)

    return line_images

def visualize_extraction(image, binary, line_regions, line_paths):
    if len(image.shape) == 2:
        vis_img = cv2.cvtColor(image, cv2.COLOR_GRAY2BGR)
    else:
        vis_img = image.copy()

    for y_start, y_end in line_regions:
        cv2.line(vis_img, (0, y_start), (vis_img.shape[1], y_start), (0, 255,␣
 ↪0), 1)
        cv2.line(vis_img, (0, y_end), (vis_img.shape[1], y_end), (0, 255, 0), 1)

    for path, _ in line_paths:
        for i in range(len(path) - 1):
            cv2.line(vis_img, (path[i][1], path[i][0]), (path[i+1][1],␣
 ↪path[i+1][0]), (0, 0, 255), 1)

    plt.figure(figsize=(15, 12))
    plt.imshow(cv2.cvtColor(vis_img, cv2.COLOR_BGR2RGB))
    plt.title("Text Line Extraction Visualization")
    plt.axis('off')
    plt.tight_layout()
    plt.show()

def extract_text_lines_from_image(image, visualize=True):
    gray, binary = preprocess_image(image)
    projection = compute_projection_profile(binary)
```

```
    line_regions = detect_text_lines(binary, projection)
    line_paths = extract_line_paths(binary, line_regions)

    if visualize:
        visualize_extraction(image, binary, line_regions, line_paths)

    line_images = extract_line_images(image, binary, line_paths)
    return line_images
```

[ ]:
```
input_image_path = "1.png"
original_image = cv2.imread(input_image_path)

if original_image is None:
    print("Error: Image not found or invalid image format.")
else:
    lines = extract_text_lines_from_image(original_image, visualize=True)

    output_folder = "output_folder"
    if not os.path.exists(output_folder):
        os.makedirs(output_folder)

    for idx, line_img in enumerate(lines, start=1):
        output_path = os.path.join(output_folder, f"line_{idx}.png")
        cv2.imwrite(output_path, line_img)
        print(f"Saved line image {idx} at {output_path}")
```

0.0.9 Final inference

[ ]:
```
processor = TrOCRProcessor.from_pretrained('qantev/trocr-large-spanish')
model = VisionEncoderDecoderModel.from_pretrained('qantev/trocr-large-spanish')
```

```
preprocessor_config.json:   0%|          | 0.00/364 [00:00<?, ?B/s]

tokenizer_config.json:   0%|          | 0.00/1.38k [00:00<?, ?B/s]

vocab.json:   0%|          | 0.00/798k [00:00<?, ?B/s]

merges.txt:   0%|          | 0.00/456k [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/2.11M [00:00<?, ?B/s]

special_tokens_map.json:   0%|          | 0.00/957 [00:00<?, ?B/s]

config.json:   0%|          | 0.00/4.97k [00:00<?, ?B/s]

pytorch_model.bin:   0%|          | 0.00/2.44G [00:00<?, ?B/s]

model.safetensors:   0%|          | 0.00/2.44G [00:00<?, ?B/s]

Config of the encoder: <class 'transformers.models.vit.modeling_vit.ViTModel'>
is overwritten by shared encoder config: ViTConfig {
  "attention_probs_dropout_prob": 0.0,
```

```
  "encoder_stride": 16,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.0,
  "hidden_size": 1024,
  "image_size": 384,
  "initializer_range": 0.02,
  "intermediate_size": 4096,
  "layer_norm_eps": 1e-12,
  "model_type": "vit",
  "num_attention_heads": 16,
  "num_channels": 3,
  "num_hidden_layers": 24,
  "patch_size": 16,
  "pooler_act": "tanh",
  "pooler_output_size": 1024,
  "qkv_bias": false,
  "torch_dtype": "float32",
  "transformers_version": "4.50.0"
}

Config of the decoder: <class
'transformers.models.trocr.modeling_trocr.TrOCRForCausalLM'> is overwritten by
shared decoder config: TrOCRConfig {
  "activation_dropout": 0.0,
  "activation_function": "relu",
  "add_cross_attention": true,
  "attention_dropout": 0.0,
  "bos_token_id": 0,
  "classifier_dropout": 0.0,
  "d_model": 1024,
  "decoder_attention_heads": 16,
  "decoder_ffn_dim": 4096,
  "decoder_layerdrop": 0.0,
  "decoder_layers": 12,
  "decoder_start_token_id": 2,
  "dropout": 0.1,
  "encoder_hidden_size": 1024,
  "eos_token_id": 2,
  "init_std": 0.02,
  "is_decoder": true,
  "layernorm_embedding": false,
  "max_position_embeddings": 1024,
  "model_type": "trocr",
  "pad_token_id": 1,
  "scale_embedding": true,
  "tie_word_embeddings": false,
  "torch_dtype": "float32",
  "transformers_version": "4.50.0",
```

```
  "use_cache": false,
  "use_learned_position_embeddings": false,
  "vocab_size": 50265
}

generation_config.json:   0%|          | 0.00/420 [00:00<?, ?B/s]
```

[ ]:
```python
# load the model
model_train = VisionEncoderDecoderModel.from_pretrained("/content/
 ↪checkpoint-1502")
processor_train = TrOCRProcessor.from_pretrained("microsoft/
 ↪trocr-base-handwritten")
```

```
Config of the encoder: <class 'transformers.models.vit.modeling_vit.ViTModel'>
is overwritten by shared encoder config: ViTConfig {
  "attention_probs_dropout_prob": 0.0,
  "encoder_stride": 16,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.0,
  "hidden_size": 768,
  "image_size": 384,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "model_type": "vit",
  "num_attention_heads": 12,
  "num_channels": 3,
  "num_hidden_layers": 12,
  "patch_size": 16,
  "pooler_act": "tanh",
  "pooler_output_size": 768,
  "qkv_bias": false,
  "torch_dtype": "float32",
  "transformers_version": "4.50.0"
}

Config of the decoder: <class
'transformers.models.trocr.modeling_trocr.TrOCRForCausalLM'> is overwritten by
shared decoder config: TrOCRConfig {
  "activation_dropout": 0.0,
  "activation_function": "relu",
  "add_cross_attention": true,
  "attention_dropout": 0.0,
  "bos_token_id": 0,
  "classifier_dropout": 0.0,
  "cross_attention_hidden_size": 768,
  "d_model": 1024,
```

```
  "decoder_attention_heads": 16,
  "decoder_ffn_dim": 4096,
  "decoder_layerdrop": 0.0,
  "decoder_layers": 12,
  "decoder_start_token_id": 2,
  "dropout": 0.1,
  "eos_token_id": 2,
  "init_std": 0.02,
  "is_decoder": true,
  "layernorm_embedding": false,
  "max_position_embeddings": 1024,
  "model_type": "trocr",
  "pad_token_id": 1,
  "scale_embedding": true,
  "tie_word_embeddings": false,
  "torch_dtype": "float32",
  "transformers_version": "4.50.0",
  "use_cache": false,
  "use_learned_position_embeddings": false,
  "vocab_size": 50265
}
```


[ ]:
```
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
```

[ ]:
```
#grammar correction
happy_tt = HappyTextToText("T5", "t5Model")
```

[ ]:
```
pth = '/content/test1.png'
image = Image.open(pth).convert("RGB")
```

[ ]:
```
#pre trained on HF
model.to(device)

pixel_values = processor(images=image, return_tensors="pt").pixel_values.
 ↪to(device)
model.eval()

with torch.no_grad():
    generated_ids = model.generate(pixel_values, max_length=128, num_beams=5)
    print("Generated IDs:", generated_ids)

generated_text = processor.batch_decode(generated_ids,␣
 ↪skip_special_tokens=True)[0]
print("Generated Text:", generated_text)
```

```
Generated IDs: tensor([[    2,     0,   417,   710,  5272,  2156,   385, 11950,
```

```
117, 16749,
           203,   102,   181,  2407,    12,     2,     1]], device='cuda:0')
Generated Text: duría, donde no ay mucha pure-
```

[ ]:
```python
result = happy_tt.generate_text(generated_text)
print(result.text)
```

```
[ dura, donde no ay mucha pur-- ]
```

[ ]:
```python
# finetuned model on the dataset
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model_train.to(device)

pixel_values = processor_train(images=image, return_tensors="pt").pixel_values.
 ↪to(device)
model_train.eval()

with torch.no_grad():
    generated_ids = model_train.generate(pixel_values, max_length=128,␣
 ↪num_beams=5)
    print("Generated IDs:", generated_ids)

generated_text = processor_train.batch_decode(generated_ids,␣
 ↪skip_special_tokens=True)[0]
print("Generated Text:", generated_text)
```

```
Generated IDs: tensor([[    0, 31695,   385,   117,   117,   203,   181,     4,
2]],
       device='cuda:0')
Generated Text: uria d no no much p.
```

[ ]:
```python
result = happy_tt.generate_text(generated_text)
print(result.text)
```

```
[ uria d no no mvch p. ]
```

[ ]:
```python

```