```
# %%capture
# #!unzip Datasets.zip


# from google.colab import drive

# # Mount the Google Drive
# drive.mount('/content/drive')


# %%capture
# !pip install datasets
# !pip install transformers
# !pip install librosa
# !pip install jiwer
# !pip install evaluate


import os
import datasets
import pandas as pd
from sklearn.model_selection import train_test_split
from datasets import Dataset

# Set paths
csv_path = "/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final.csv"
audio_folder = "/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-w

# Load the CSV
df = pd.read_csv(csv_path)
df = pd.read_csv(csv_path)
# Ensure the column names match
df.columns = ["Filename", "Transcription"]  # Rename columns if needed

# Append '.wav' to the file names
df['Filename'] = df['Filename'].apply(lambda x: f"{x}.wav")

# Add full paths to the audio files
df['file_path'] = df['Filename'].apply(lambda x: os.path.join(audio_folder, x))

# Verify that all audio files exist
missing_files = df[~df['file_path'].apply(os.path.exists)]
if not missing_files.empty:
    print("The following audio files are missing:")
    print(missing_files)
    raise FileNotFoundError("Some audio files listed in the CSV are missing in the folder.")

# Split into train (27) and test (3)
train_df, test_df = train_test_split(df, test_size=0.2, random_state=42)

# Save splits to CSV for reference
train_csv_path = "train_split.csv"
test_csv_path = "test_split.csv"
train_df.to_csv(train_csv_path, index=False)
test_df.to_csv(test_csv_path, index=False)

# Convert to HuggingFace Dataset format
train_dataset = Dataset.from_pandas(train_df)
test_dataset = Dataset.from_pandas(test_df)

# Save HuggingFace datasets
train_dataset_path = "train_dataset"
test_dataset_path = "test_dataset"
train_dataset.save_to_disk(train_dataset_path)
test_dataset.save_to_disk(test_dataset_path)

# Output
print(f"Train set saved to: {train_csv_path} and {train_dataset_path}")
print(f"Test set saved to: {test_csv_path} and {test_dataset_path}")
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/tqdm/auto.py:21: TqdmWarning: IProgress not found. Pleas
  from .autonotebook import tqdm as notebook_tqdm
Saving the dataset (1/1 shards): 100%|██████████| 2600/2600 [00:00<00:00, 719149.99 examples/s]
Saving the dataset (1/1 shards): 100%|██████████| 650/650 [00:00<00:00, 371582.06 examples/s]Train set saved to: train_s
Test set saved to: test_split.csv and test_dataset
```

```
from datasets import load_from_disk

train_dataset = load_from_disk("train_dataset")
test_dataset = load_from_disk("test_dataset")
```

```
print(train_dataset)
print(test_dataset)
```

```
Dataset({
    features: ['Filename', 'Transcription', 'file_path', '__index_level_0__'],
    num_rows: 2600
})
Dataset({
    features: ['Filename', 'Transcription', 'file_path', '__index_level_0__'],
    num_rows: 650
})
```

```
from datasets import ClassLabel
import random
import pandas as pd
from IPython.display import display, HTML

def show_random_elements(dataset, num_examples=10):
    assert num_examples <= len(dataset), "Can't pick more elements than there are in the dataset."
    picks = []
    for _ in range(num_examples):
        pick = random.randint(0, len(dataset)-1)
        while pick in picks:
            pick = random.randint(0, len(dataset)-1)
        picks.append(pick)

    df = pd.DataFrame(dataset[picks])
    display(HTML(df.to_html()))

show_random_elements(train_dataset)
```

	Filename	Transcription	file_path	__index_level_0__
0	jhon-01.1_126.wav	علمدارِ پتہٕ با ستہٕ پرہیزگار توأریخن پس خطِ ارشادِ مَن حضرتِ میر محمد ہمدانی	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/jhon-01.1_126.wav	1956
1	jhon-03_80.wav	یتھ باغِ سُلیمان نادٕرُچھ چھٕ اَمہٕ لیچھ ہِژ مغازیٔ النبی رسالہِ گل و بلبل تہٕ	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/jhon-03_80.wav	2901
2	ishrat1-15_10.wav	امہٕ کنۍ آمت توے چُھ نافہ ژھٹاں از مہٕ قسم وجاں شامن	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/ishrat1-15_10.wav	1480
3	jhon-01.2_30.wav	سپیدیٕ نہٕ تس کار گر آتھی دوران وُچھ زیٲ سُکھ ماجہٕ خابس ۍ	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/jhon-01.2_30.wav	2533
4	8140192.wav	گژھان یہٕ چھ اَسی مأنتھ زِ عروضچ بنیاد	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/8140192.wav	191
5	rafiya-04_16.wav	تہٕ مثنویٔ آسہ تس وہرأژ روٚدٕکی پأٹھی قلمہٕ ۍ شرانہٕ وسان	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/rafiya-04_16.wav	3090
6	jhon-01.1_87.wav	اپس واتان توھٕزَن چھ سأرِی کَشیر اَکھ زیٹھ زیوٹھ سرینگر	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/jhon-01.1_87.wav	2496
7	ishrat1-34_5.wav	مُشتھ گوے کیاز وتم گام اَخر	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/ishrat1-34_5.wav	1710
8	jhon-03_43.wav	مِن وَجہہ شمس الضحیٰ مِن خدہ بَدرُ الدجیٰ	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/jhon-03_43.wav	2860
9	rafiya-05_23.wav	دَلیلہٕ کردار حالات بلکہ مُلک تہٕ مأدانِ جنگِ پریتھ کیۍ ہہٕ چھ	/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-waves/rafiya-05_23.wav	3168

If there are any unwanted special characters in the dataset, we can remove them here, since there are none, I am keeping that as it is.

```
def extract_all_chars(batch):
  all_text = " ".join(batch["Transcription"])
  vocab = list(set(all_text))
  return {"vocab": [vocab], "all_text": [all_text]}

vocab_train = train_dataset.map(extract_all_chars, batched=True, batch_size=-1, keep_in_memory=True, remove_columns=train_da
vocab_test = test_dataset.map(extract_all_chars, batched=True, batch_size=-1, keep_in_memory=True, remove_columns=test_datas
```

```
Map: 100%|██████████| 2600/2600 [00:00<00:00, 214961.08 examples/s]
Map: 100%|██████████| 650/650 [00:00<00:00, 251805.45 examples/s]
```

```
vocab_list = list(set(vocab_train["vocab"][0]) | set(vocab_test["vocab"][0]))

vocab_dict = {v: k for k, v in enumerate(sorted(vocab_list))}
vocab_dict
```

Show hidden output

```
vocab_dict["|"] = vocab_dict[" "]
del vocab_dict[" "]


vocab_dict["[UNK]"] = len(vocab_dict)
vocab_dict["[PAD]"] = len(vocab_dict)
len(vocab_dict)
```

```
60
```

```
import json
with open('vocab.json', 'w') as vocab_file:
    json.dump(vocab_dict, vocab_file)
```

```
from transformers import Wav2Vec2CTCTokenizer

tokenizer = Wav2Vec2CTCTokenizer.from_pretrained("./", unk_token="[UNK]", pad_token="[PAD]", word_delimiter_token="|", clear
```

```
from transformers import Wav2Vec2FeatureExtractor

feature_extractor = Wav2Vec2FeatureExtractor(feature_size=1, sampling_rate=16000, padding_value=0.0, do_normalize=True, retu
```

```
from transformers import Wav2Vec2Processor

processor = Wav2Vec2Processor(feature_extractor=feature_extractor, tokenizer=tokenizer)
```

```
train_dataset[0]["file_path"]
```

```
'/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/final-
waves/farhat-03_21.wav'
```

Replacing the File Path with Actual Audio.

```
from datasets import load_from_disk, Audio

# Load datasets
train_dataset = load_from_disk("train_dataset")  # Adjust to your actual path
test_dataset = load_from_disk("test_dataset")

# Rename 'file_path' to 'audio'
train_dataset = train_dataset.rename_column("file_path", "audio")
test_dataset = test_dataset.rename_column("file_path", "audio")

# # Cast the 'audio' column to use the Audio feature
train_dataset = train_dataset.cast_column("audio", Audio(sampling_rate=16_000))
test_dataset = test_dataset.cast_column("audio", Audio(sampling_rate=16_000))

# # Drop unnecessary columns if needed
train_dataset = train_dataset.remove_columns(["__index_level_0__"])
test_dataset = test_dataset.remove_columns(["__index_level_0__"])

# # Verify the dataset structure
print(train_dataset)
print(test_dataset)

# # Inspect the first example
print(train_dataset[0])
```

```
Dataset({
    features: ['Filename', 'Transcription', 'audio'],
    num_rows: 2600
})
Dataset({
    features: ['Filename', 'Transcription', 'audio'],
    num_rows: 650
})
{'Filename': 'farhat-03_21.wav', 'Transcription': 'زأنہٕ شاہ صأبن بٮ۫شابسمرا قبہٕ پتہٕ تہٕ یٖیٖلہٕ\xa0', 'audio': {'path': '/ho
        0.0256958 ,  0.02392578]), 'sampling_rate': 16000}}
```

```
#print(test_dataset[0]['audio'])
```

```
rand_int = random.randint(0, len(train_dataset))

print("Target text:", train_dataset[rand_int]["Transcription"])
print("Input array shape:", train_dataset[rand_int]["audio"]["array"].shape)
print("Sampling rate:", train_dataset[rand_int]["audio"]["sampling_rate"])
```

```
Target text: ژۆر مأری ئندی کأشِری چھا شعر کأشِر زبأنی توش تہٕ بوش عطا کران
Input array shape: (139205,)
Sampling rate: 16000
```

```
def prepare_dataset(batch):
    audio = batch["audio"]

    # batched output is "un-batched"
    batch["input_values"] = processor(audio["array"], sampling_rate=audio["sampling_rate"]).input_values[0]
    batch["input_length"] = len(batch["input_values"])

    batch["labels"] = processor(text=batch["Transcription"]).input_ids

    return batch


train_dataset = train_dataset.map(prepare_dataset, remove_columns=train_dataset.column_names)
test_dataset = test_dataset.map(prepare_dataset, remove_columns=test_dataset.column_names)
```

```
import torch

from dataclasses import dataclass, field
from typing import Any, Dict, List, Optional, Union

@dataclass
class DataCollatorCTCWithPadding:
    """
    Data collator that will dynamically pad the inputs received.
    Args:
        processor (:class:`~transformers.Wav2Vec2Processor`)
            The processor used for proccessing the data.
        padding (:obj:`bool`, :obj:`str` or :class:`~transformers.tokenization_utils_base.PaddingStrategy`, `optional`, defa
            Select a strategy to pad the returned sequences (according to the model's padding side and padding index)
            among:
            * :obj:`True` or :obj:`'longest'`: Pad to the longest sequence in the batch (or no padding if only a single
              sequence if provided).
            * :obj:`'max_length'`: Pad to a maximum length specified with the argument :obj:`max_length` or to the
              maximum acceptable input length for the model if that argument is not provided.
            * :obj:`False` or :obj:`'do_not_pad'` (default): No padding (i.e., can output a batch with sequences of
              different lengths).
    """

    processor: Wav2Vec2Processor
    padding: Union[bool, str] = True

    def __call__(self, features: List[Dict[str, Union[List[int], torch.Tensor]]]) -> Dict[str, torch.Tensor]:
        # split inputs and labels since they have to be of different lenghts and need
        # different padding methods
        input_features = [{"input_values": feature["input_values"]} for feature in features]
        label_features = [{"input_ids": feature["labels"]} for feature in features]

        batch = self.processor.pad(
            input_features,
            padding=self.padding,
            return_tensors="pt",
        )
        with self.processor.as_target_processor():
            labels_batch = self.processor.pad(
                label_features,
                padding=self.padding,
                return_tensors="pt",
            )

        # replace padding with -100 to ignore loss correctly
        labels = labels_batch["input_ids"].masked_fill(labels_batch.attention_mask.ne(1), -100)

        batch["labels"] = labels

        return batch

data_collator = DataCollatorCTCWithPadding(processor=processor, padding=True)
```

```
import evaluate

wer_metric = evaluate.load("wer")
```

```
2025-04-06 07:08:16.786995: E external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:477] Unable to register cuFFT fact
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
E0000 00:00:1743903496.837447    2721 cuda_dnn.cc:8310] Unable to register cuDNN factory: Attempting to register factory
E0000 00:00:1743903496.852595    2721 cuda_blas.cc:1418] Unable to register cuBLAS factory: Attempting to register facto
```

```
2025-04-06 07:08:16.968900: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler fl
Using the latest cached version of the module from /home/muzaffar/.cache/huggingface/modules/evaluate_modules/metrics/ev
```

```
from evaluate import load

cer_metric = load("cer")
```

```
Using the latest cached version of the module from /home/muzaffar/.cache/huggingface/modules/evaluate_modules/metrics/ev
```

includes both WER and CER

```
def compute_metrics(pred):
    pred_logits = pred.predictions
    pred_ids = np.argmax(pred_logits, axis=-1)

    # Replace padding token (-100) with pad_token_id
    pred.label_ids[pred.label_ids == -100] = processor.tokenizer.pad_token_id

    # Decode predictions and labels to strings
    pred_str = processor.batch_decode(pred_ids)
    label_str = processor.batch_decode(pred.label_ids, group_tokens=False)

    if isinstance(label_str, list):
        if isinstance(pred_str, list) and len(pred_str) == len(label_str):
            for index in random.sample(range(len(label_str)), 3):
                print(f'reference: "{label_str[index]}"')
                print(f'predicted: "{pred_str[index]}"')

        else:
            for index in random.sample(range(len(label_str)), 3):
                print(f'reference: "{label_str[index]}"')
                print(f'predicted: "{pred_str}"')

    # Compute WER
    wer = wer_metric.compute(predictions=pred_str, references=label_str)

    # Compute CER
    cer = cer_metric.compute(predictions=pred_str, references=label_str)

    return {"wer": wer, "cer": cer}
```

```
#mmmmiiiiiinnnneeeee
```

```
# from transformers import Wav2Vec2ForCTC

# model = Wav2Vec2ForCTC.from_pretrained(
#     # "facebook/wav2vec2-xls-r-300m",
#     'facebook/wav2vec2-large-xlsr-53',
#     attention_dropout=0.05,
#     hidden_dropout=0.1,
#     feat_proj_dropout=0.1,
#     mask_time_prob=0.05,
#     layerdrop=0.01377,
#     gradient_checkpointing=True,
#     ctc_loss_reduction="mean",
#     ctc_zero_infinity=True,
#     pad_token_id=processor.tokenizer.pad_token_id,
#     vocab_size=len(processor.tokenizer),


# )
```

```
Some weights of Wav2Vec2ForCTC were not initialized from the model checkpoint at facebook/wav2vec2-large-xlsr-53 and are
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
```

```
#pppeeeerrrrrseeaain-------iran

from transformers import Wav2Vec2ForCTC

model = Wav2Vec2ForCTC.from_pretrained(
    'facebook/wav2vec2-large-xlsr-53',
    attention_dropout=0.05,
```

```
        activation_dropout=0.1,
        hidden_dropout=0.1,
        feat_proj_dropout=0.01249,
        final_dropout=0.0,
        mask_time_prob=0.05,
        mask_time_length=10,
        mask_feature_prob=0,
        mask_feature_length=10,
        layerdrop=0.01377,
        gradient_checkpointing=True,
        ctc_loss_reduction="mean",
        ctc_zero_infinity=True,
        bos_token_id=processor.tokenizer.bos_token_id,
        eos_token_id=processor.tokenizer.eos_token_id,
        pad_token_id=processor.tokenizer.pad_token_id,
        vocab_size=len(processor.tokenizer.get_vocab())
    )
```

```
Some weights of Wav2Vec2ForCTC were not initialized from the model checkpoint at facebook/wav2vec2-large-xlsr-53 and are
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
```

```
    model.freeze_feature_encoder()


    # import huggingface_hub

    # huggingface_hub.login()


    #repo_name = "wav2vec2-kashmiri-jhon-data-one"


    save_dir = "/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/2training_


    #MMMMMMMIIIINNNEEEEE



    from transformers import TrainingArguments

    training_args = TrainingArguments(
        output_dir=save_dir,
        group_by_length=True,
        per_device_train_batch_size=8,
        per_device_eval_batch_size=8,
        gradient_accumulation_steps=2,
        evaluation_strategy="steps",
        num_train_epochs=30,
        fp16=True,
        save_steps=500,
        eval_steps=500,
        logging_steps=10,
        learning_rate=4e-4,
        warmup_steps=250,
        save_total_limit=2,
        dataloader_num_workers=24
    )
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/training_args.py:1594: FutureWarning: `eval
  warnings.warn(
```

```
# #PERSIAN

# from transformers import TrainingArguments

# training_args = TrainingArguments(
#   output_dir=repo_name,
#   group_by_length=True,
#   per_device_train_batch_size=2,
#   gradient_accumulation_steps=2,
#   eval_strategy="steps",
#   num_train_epochs=20,
#   gradient_checkpointing=True,
#   fp16=True,
#   save_steps=20,
#   eval_steps=20,
#   logging_steps=40,
#   learning_rate=3e-4,
#   warmup_steps=50,
#   save_total_limit=2,
#   push_to_hub=True,
# )


# import numpy as np
# from transformers import Trainer
# trainer = Trainer(
#     model=model,
#     data_collator=data_collator,
#     args=training_args,
#     compute_metrics=compute_metrics,
#     train_dataset=train_dataset,
#     eval_dataset=test_dataset,
#     tokenizer=processor.feature_extractor,
# )


import numpy as np
from transformers import Trainer

# Assuming processor is an instance of Wav2Vec2Processor (or similar for your model)
trainer = Trainer(
    model=model,
    data_collator=data_collator,
    args=training_args,
    compute_metrics=compute_metrics,
    train_dataset=train_dataset,
    eval_dataset=test_dataset,
    processing_class=processor,  # Use the processor directly for feature extraction
)




print("step1")
train_result = trainer.train()
print("step2")

metrics = train_result.metrics
print("step3")
max_train_samples = len(train_dataset)
metrics["train_samples"] = min(max_train_samples, len(train_dataset))
print("step4")
trainer.save_model()
print("model created!")
trainer.log_metrics("train", metrics)
trainer.save_metrics("train", metrics)
trainer.save_state()
```

```
step1
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/torch/utils/data/dataloader.py:624: UserWarning: This Da
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
```

[4860/4860 7:43:53, Epoch 29/30]

Step	Training Loss	Validation Loss	Wer	Cer
500	2.490800	2.142393	1.000000	0.708172
1000	0.573100	0.633072	0.576649	0.167976
2000	0.310000	0.647443	0.498264	0.141202
2500	0.187500	0.704146	0.487667	0.139980
3000	0.109900	0.731170	0.457336	0.131951
3500	0.090700	0.786579	0.445277	0.127134
4000	0.089800	0.802601	0.439795	0.123538
4500	0.065800	0.806295	0.438699	0.123294

```
trainer.evaluate()  # Evaluate the model on the test dataset
```

```
Trainer.tokenizer is now deprecated. You should use Trainer.processing_class instead.
Trainer.tokenizer is now deprecated. You should use Trainer.processing_class instead.
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/torch/utils/data/dataloader.py:624: UserWarning: This Da
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
```

```
# trainer.push_to_hub()
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
```

TESTING WITH OUT KENLM

```
import torch
import torchaudio
import librosa
import numpy
from transformers import Wav2Vec2ForCTC, Wav2Vec2Processor
from transformers import Wav2Vec2Processor
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
```

```
# model_name_or_path = "/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment
# device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
# print(model_name_or_path, device)

# processor = Wav2Vec2Processor.from_pretrained(model_name_or_path)
# model = Wav2Vec2ForCTC.from_pretrained(model_name_or_path).to(device)


# def speech_file_to_array_fn(batch):
#     speech_array, sampling_rate = torchaudio.load(batch["file_path"])
#     speech_array = speech_array.squeeze().numpy()
#     #speech_array = librosa.resample(np.asarray(speech_array), sampling_rate, processor.feature_extractor.sampling_rate)
#     speech_array = librosa.resample(y=np.asarray(speech_array), orig_sr=sampling_rate, target_sr=processor.feature_extract


#     batch["speech"] = speech_array
#     return batch


# def predict(batch):
#     features = processor(
#         batch["speech"],
#         sampling_rate=processor.feature_extractor.sampling_rate,
#         return_tensors="pt",
#         padding=True
#     )

#     input_values = features.input_values.to(device)
#     #attention_mask = features.attention_mask.to(device)
#     attention_mask = features.attention_mask.to(device) if "attention_mask" in features else None


#     with torch.no_grad():
#         logits = model(input_values, attention_mask=attention_mask).logits

#     pred_ids = torch.argmax(logits, dim=-1)

#     batch["predicted_N_LM"] = processor.batch_decode(pred_ids)
#     return batch
```

```
# import torchaudio
# import librosa
# from datasets import load_dataset
# import numpy as np

# dataset = load_dataset("csv", data_files={"/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Pape

# dataset = dataset.map(speech_file_to_array_fn)
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
```

```
def speech_file_to_array_fn(batch):
    speech_array, sampling_rate = torchaudio.load(batch["file_path"])
    speech_array = speech_array.squeeze().numpy()  # Convert to numpy array

    # Ensure the audio is always a 1D NumPy array (sometimes it's multi-channel)
    if len(speech_array.shape) > 1:
        speech_array = np.mean(speech_array, axis=0)  # Convert stereo to mono

    # Resample to match the processor's expected sample rate
    speech_array = librosa.resample(
        y=np.asarray(speech_array),
        orig_sr=sampling_rate,
        target_sr=processor.feature_extractor.sampling_rate
    )

    batch["speech"] = speech_array.tolist()  # Convert to Python list (ensures consistency)
    return batch
```

```
    warnings.warn(
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
```

```
model_name_or_path = "/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print(model_name_or_path, device)

processor = Wav2Vec2Processor.from_pretrained(model_name_or_path)
model = Wav2Vec2ForCTC.from_pretrained(model_name_or_path).to(device)


def predict(batch):
    features = processor(
        batch["speech"],
        sampling_rate=processor.feature_extractor.sampling_rate,
        return_tensors="pt",
        padding=True
    )

    input_values = features.input_values.to(device)
    #attention_mask = features.attention_mask.to(device)
    attention_mask = features.attention_mask.to(device) if "attention_mask" in features else None


    with torch.no_grad():
        logits = model(input_values, attention_mask=attention_mask).logits

    pred_ids = torch.argmax(logits, dim=-1)

    batch["predicted_N_LM"] = processor.batch_decode(pred_ids)
    return batch
```

```
import torchaudio
import librosa
from datasets import load_dataset
import numpy as np

dataset = load_dataset("csv", data_files={"/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/

dataset = dataset.map(speech_file_to_array_fn)
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
/home/muzaffar/Desktop/Research/papers/5-paper Wav2Vec/5. Wave2vec Whisper Paper/KASHMIRI/experiment5/2training_all_deta
Map: 100%|[redacted]| 650/650 [00:07<00:00, 88.12 examples/s]
/home/muza[redacted]nda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
```

```
result = dataset.map(predict, batched=True, batch_size=4)
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
Parameter 'function'=<function predict at 0x7f8a77adb380> of the transform datasets.arrow_dataset.Dataset._map_single co
Map: 100%|[redacted]| 650/650 [00:55<00:00, 11.80 examples/s]
/home/muza[redacted]nda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
```

```
from evaluate import load  # Use `evaluate` instead of `datasets`

# Load WER and CER metrics
wer = load("wer")
cer = load("cer")

# Compute WER and CER using the correct split
print("WER: {:.2f}".format(100 * wer.compute(predictions=result["train"]["predicted_N_LM"],
                                             references=result["train"]["Transcription"])))
print("CER: {:.2f}".format(100 * cer.compute(predictions=result["train"]["predicted_N_LM"],
                                             references=result["train"]["Transcription"])))
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
Using the latest cached version of the module from /home/muzaffar/.cache/huggingface/modules/evaluate_modules/metrics/ev
Using the latest cached version of the module from /home/muzaffar/.cache/huggingface/modules/evaluate_modules/metrics/ev
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
WER: 43.49
CER: 12.30
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
```

```
for i in range(len(result["train"])):  # Specify the "train" split
    reference = result["train"]["Transcription"][i]  # Use "Text" as reference
    predicted_N_LM = result["train"]["predicted_N_LM"][i]

    if reference.strip() == predicted_N_LM.strip():
        continue

    print("Reference:", reference)
    print("Predicted:", predicted_N_LM)
    print('---')
```

```
predicted: "رنلچوتکرب"
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/torch/utils/data/dataloader.py:624: UserWarning: This Da
```

```
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/torch/utils/data/dataloader.py:624: UserWarning: This Da
Reference: واتناوان میوِ کُلی زُوان تہٕ گوشہ نشینی اختیار کرن
  warnings.warn(
Predicted: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
Reference: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Predicted: گتھ کٔری نہٕ پۆنپریو تہٕ دودٕے سانہٕ کانہہ شمع
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Reference: اتھ وعدٕ کران وألی چھہ سوال کران زِ تییٚلہ کیازِ کرِہے پانہٕ بٲشہ
  warnings.warn(
Predicted: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
Reference: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Predicted: چھَس بونی سوفریس سُہ کُٹھ تہٕ کُچھ ہٕنز ہرٕ ہِھ
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Reference: شایہِ تہٕ اَنر دُنہ لولہٕ کتھ ہے اوتار بھٹ پننہٕ شأعری ہند
  warnings.warn(
Predicted: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
Reference: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Predicted: رأژن ہاوکھ ناہ دوہ میأنی تعریقی پبٹھ کیاہ چھِھ
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Reference: تِمن سِتی منسوٗب واقعات چھہ کأنسہ لُکہٕ شأعرن موزوٗن کٔرۍمتی
  warnings.warn(
Predicted: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
Reference: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Predicted: چھہ کأبلہ قبوٗل تہٕ اَمہِ خاندان کہ شجرِ نسبُک تلکھ
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Reference: انِچ شوٗب زوٗن أس نہٕ فقط یوسُف شِہنی محلہٕ خانِچ کونگٕ تہٕچ
  warnings.warn(
Predicted: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
Reference: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Predicted: بسان ردیف تہٕ کأفیاہ چُھس شعرٕ پت شعرٕ بنان
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Reference: کینژن ہند خیال چُھ زِ زول چُھ یوان زأنہٕ شاہ صأبنٕ ہِس
  warnings.warn(
Predicted: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
  warnings.warn(
Reference: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Predicted: نہٕ سنزُک پے نہٕ کانہہ پتہھ ہے کتھ
  warnings.warn(
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
Reference: پانہٕ ونی سلاماہ بندیخاہ کٔرِتھ تہٕ کتھ
  warnings.warn(
Predicted: [illegible]
/home/muzaffar/anaconda3/envs/tf14/lib/python3.11/site-packages/transformers/models/wav2vec2/processing_wav2vec2.py:174:
```