COSI 136 Project 2

Timothy Obiso, Anastasiia Tatlubaeva

December 15, 2023

1 Project goal

The goal of this project was to build an Automatic Speech Recognition system for Czech, trained and tested on audio files and paired transcriptions from the Book of Genesis from the Bible. We trained two Whisper models using the first 50 chapters of Genesis. We aimed to create an ASR tool capable of transcribing Czech speech, focusing on formal situations.

While our corpus for this project was relatively small and limited to recordings from a single speaker, we achieved solid results that can be used as a foundation for future improvements. While we recognize that the formulation of our corpus may limit the model's performance on more diverse speech patterns found in everyday language usage, we decided to only use data with accompanying transcriptions. Our model could be improved by adding recordings from different speakers (in age, gender, race, etc.) and a variety of speech styles and discourse situations. This expansion would make our ASR model more versatile, allowing it to accurately process a wider range of Czech audio content.

2 Corpus

For this project, we used the Bible audio and text from the Book of Genesis in Czech. The audio files were downloaded from `https://www.wordproject.org/`. The text files were obtained by scraping the same website using `BeautifulSoup`. In total, there were 50 mp3 files and 50 txt files, corresponding to 50 distinct chapters. The length of each mp3 file varied from 3 to 13 minutes. All 50 audio files were recorded by the same male speaker who seems to be 30-65 years old. Unfortunately, there is no biographical information available on him.

2.1 Preprocessing

All mp3 files were first converted to wav format. Then, these wav files were resampled to 16 kHz and rechanneled to one channel. After that, we divided all audio data into wav files of the same length, resulting in files of around 7 seconds. In total, we had 2,413 audio files. However, some of them inadvertently turned out to be "empty" due to the speaker making long pauses between sentences. Those files were discarded from our project. As a result, we used 2,390 audio files: 1,910 for training and 480 for testing. In terms of duration, we had 3 hours and 54 minutes of training data and 1 hour of testing data.

The text files are stored in `textgrids`. The list of train and test files are found in `train_files.txt` and `test_files.txt` respectively. Since paired files are named identically (except for the extension), these files include the split of the audio and text files. Here is an example of a training instance before converting it to a format used by Whisper. The audio data here is a NumPy array, with each element representing a discrete sample of the audio waveform:

```
'audio': array([ 0.02685547, 0.03381348, 0.03799438, ...,
 -0.00036621, -0.00036621, -0.00085449], dtype=float32),
'sentence': 'jméno jeho Ezau Potom pak vyšel bratr jeho a rukou'
```

At this point, we needed to convert the raw audio files into a format suitable for training our Whisper models. Here is a brief description of each step:

- **Feature extraction**: Using the `WhisperFeatureExtractor`, we converted the raw audio data, represented by NumPy arrays of sample amplitudes, into sets of log-Mel spectrogram features.
- **Tokenization**: The corresponding transcriptions were tokenized using the `WhisperTokenizer`. This tokenizer was specialized for the Czech language and the task of transcription. It converted each sentence into a sequence of numerical tokens. Each token represents a word or subword unit from the sentence, so that the model could map audio features to textual components.

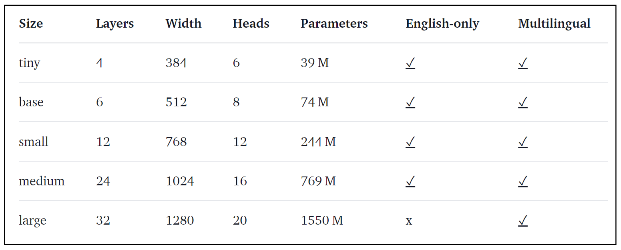
Here is the training instance from the example before but in the format suitable for Whisper:

```
'input_features': array([[ 0.5140126 , 0.0890159 , 0.1645338 , ..., -0.65262187,
 -0.65262187, -0.65262187], [ 0.52374864, -0.05902326, 0.04201525, ..., -0.65262187,
-0.65262187, -0.65262187], [ 0.5416091 , 0.35897875, 0.21340752, ..., -0.65262187,
-0.65262187, -0.65262187], ..., [-0.5880704 , -0.65262187, -0.65262187, ...,
-0.65262187, -0.65262187, -0.65262187], [-0.593704 , -0.65262187, -0.65262187, ...,
-0.65262187, -0.65262187, -0.65262187], [-0.5965189 , -0.65262187, -0.65262187, ...,
-0.65262187, -0.65262187, -0.65262187]], dtype=float32),
 'labels': [50258, 50283, 50359, 50363, 35195, 526, 1771, 1506, 1289, 27211, 1459,
9145, 298, 20843, 44766, 7891, 338, 47869, 81, 1506, 1289, 257, 367, 2034, 263, 50257]
```

Finally, the data is standardized using `WhisperProcessor`. `input_features` are extracted and padded for each instance to ensure that all sequences in a batch have the same length. `labels` are treated in the same way. The padded areas in `labels` are marked with a special token ID (-100) to indicate to the loss function that these areas should be ignored during model training, as they do not correspond to any meaningful data.

3 Model architecture and parameters

For this project, we used two versions of OpenAI's Whisper model: base and small. `whisper-base` served as our baseline which we compared to a larger model, `whisper-small`. `whisper-small` has twice as many layers, is 50% wider and has 12 heads as opposed to 8. It also has 244M parameters compared to `whisper-base` which only has 74M. The specifications of each configuration are summarised in the following table:



Size	Layers	Width	Heads	Parameters	English-only	Multilingual
tiny	4	384	6	39 M	✓	✓
base	6	512	8	74 M	✓	✓
small	12	768	12	244 M	✓	✓
medium	24	1024	16	769 M	✓	✓
large	32	1280	20	1550 M	x	✓

Figure 1: Comparison of Whisper configurations (HuggingFace)

3.1 Base model

This section presents the parameters of the `whisper-base` configuration that achieved the best results based on WER:

- `per_device_train_batch_size`: 8 - number of training examples.
- `gradient_accumulation_steps`: 1 - number of steps over which gradients are accumulated; in our case, gradients are updated after processing every batch.
- `learning_rate`: 1e-6 - learning rate.
- `warmup_steps`: 250 - number of steps for learning rate warmup.
- `max_steps`: 2000 - total number of training steps.
- `fp16`: True - training with mixed precision (FP16) for better speed and memory efficiency.
- `evaluation_strategy`: `steps` - evaluation of the model was performed at regular step intervals.
- `per_device_eval_batch_size`: 8 - number of evaluation examples.
- `predict_with_generate`: True - during inference, the model the output sequence iteratively
- `eval_steps`: 500 - model was evaluated after every 500 steps.
- `metric_for_best_model`: `WER` - model performance was evaluated based on Word Error Rate

3.2 Small model

Here are the best parameters of the `whisper-small` model configuration based on WER:

- `per_device_train_batch_size`: 16
- `gradient_accumulation_steps`: 1

- `learning_rate`: 1e-5
- `warmup_steps`: 500
- `max_steps`: 2000
- `fp16`: True
- `evaluation_strategy`: `steps`
- `per_device_eval_batch_size`: 8
- `predict_with_generate`: True
- `eval_steps`: 500
- `metric_for_best_model`: `WER`

4 Results

The tables below show the training results for two versions of the Whisper model. Table 1 shows the performance of the base model, and Table 2 shows the performance of the small model. Each table captures the progression of training loss, validation loss, and Word Error Rate (WER) at significant training steps

Step	Training Loss	Validation Loss	WER
500	0.014	0.528	30.345
1000	0.012	0.536	30.428
1500	0.016	0.544	30.345
2000	0.009	0.545	30.428

Table 1: Results for whisper-base

Step	Training Loss	Validation Loss	WER
500	0.115	0.390	34.944
1000	0.010	0.417	34.064
1500	0.001	0.446	29.963
2000	0.001	0.456	29.897

Table 2: Results for whisper-small

5 Discussion

As we expected, the larger model was able to perform better. Although the larger model took about 1500 steps to outperform the smaller model, it continues to improve while the smaller model stabilizes with a WER of approximately 30.4. Due to the time it took to train, we were only able to train the larger model once to 2000 steps. If training was continued, we believe the model's performance would also continue to improve. Some changes could be made to lead to improved performance: a more extensive hyperparameter tuning process such as a grid search, continuing training with more steps, using a smaller batch size, and training on an even larger model. The most common error produced by both models involved proper nouns commonly restricted to the Bible such as names and places.

It is crucial to contextualize our results by comparing them to established benchmarks. In particular, we refer to the benchmarks set by the original Whisper models as reported in *Robust Speech Recognition via Large-Scale Weak Supervision*, which can be accessed here. According to this paper, 401 hours of Czech audio data were used for training all configurations of Whisper on the transcription task. The paper presents the WER evaluation of the model on the Common Voice dataset which we compare to the performance of our models in Table 3:

Configuration	Standard Whisper model	Our fine-tuned Whisper model
base	63.1	30.4
small	34.1	29.9

Table 3: Comparison of standard Whisper models and our Whisper models

Our project's focus on fine-tuning the Whisper models with approximately 4 hours of additional training data led to remarkable improvements. Notably, the WER for the whisper-base model was reduced from 63.1 to 30.4. While the improvement for the whisper-small model was less pronounced, there was still a noteworthy reduction in WER from 34.1 to 29.9. These outcomes highlight the potential for improving model performance through the inclusion of more and varied data.

6 Future Work

One of the major strengths of our corpus is the lack of background noise. In addition, the speaker reads with clear and deliberate pronunciation which is very important for training an ASR model. Furthermore, the language used in religious texts like the Bible is often formal and standardized. As a result, it was easier for us to maintain consistency in the dataset.

At the same time, the Bible does not cover the full spectrum of language used in everyday conversations. It includes too many archaic or uncommon words and is completely devoid of slang or modern language. This aspect of our dataset could become a challenge if we tried to apply the model trained on our dataset to some data collected in more informal environments. Our model could also struggle with more casual or colloquial pronunciation and styles found in other, non-religious contexts. Since the only available recordings of the Bible in Czech were recorded by men, our model is likely biased towards male speakers, leading to potential challenges when transcribing female or child speech. The addition of more training data from multiple speakers would improve our system. A more significant improvement could be achieved by ensuring diversity of age, gender, and dialect in the additional data.