University of Sheffield

Computer Says 'Know': ASR Confidence and Transcription

Adam Spencer

Supervisor: Professor JP Barker

A report submitted in fulfilment of the requirements
for the degree of BSc in Artificial Intelligence and Computer Science

in the

Department of Computer Science

May 16, 2023

Declaration

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Adam Spencer

Signature:

Date: May 16, 2023

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum. Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Aims and Objectives

1.2 Overview of the Report

Chapter 2

Literature Survey

2.1 Automatic Speech Recognition

2.1.1 What is ASR?

Automatic Speech Recognition (ASR) is a technology which allows computers to recognise and produce a text transcription of spoken language. The research and development of technology involving speech has been a part of computer science since the late 1930s[1, 2], with rudimentary ASR systems being constructed as early as the 1950s[3]. These early attempts at recognising human speech treated it as a 'pattern matching' problem, the theory being that words could be constructed by matching the pattern created in a speech signal to corresponding spoken phonemes[1]. This paradigm falls apart when the system must be re-tuned for each individual, even for simple tasks such as recognising spoken digits[3], due to the fact that individual speakers don't produce exactly the same signal for each phoneme[4].

Since the 1970s, finding the solution to the problem of pattern matching for speech recognition has been considered unviable through the precise matching of patterns but instead finding the most probable pattern using statistical modelling[1]. The method which became most widely adopted and is still at the heart of modern ASR is Hidden Markovian Modelling, which was first applied to ASR in the '70s[5] and continued through the '90s[6] until today[7].

2.1.2 Hidden Markov Models

In his 1960 work[8], Dynkin describes a Markov process using the example of a randomly-moving particle in space;

" If the position of the particle is known at the instant t, supplementary information regarding the phenomena observed up till the instant t (and in particular, regarding the nature of the motion until t) has no effect on prognosis of the motion after the instant t (for a known "present", the "future" and the "past" are independent of eachother). "

From his description, we can draw the following assumptions for modelling a system as a Markov process;

- The system consists of states.
- The system is *in motion*, i.e. moving between states.
- This motion is random.
- The motion observed prior to t (say, $t-1$) does not influence $t+k$ where $k \geq 1$.
- Because the particle is constantly moving between states, the state at time t depends only on the state immediately prior, $t-1$.
- There is some probability, p, that the system moves from one state to another.

In a *Hidden* Markov Model (HMM), the states and transition probabilities between them are known, but for some ouput sequence the order and selection of states used to produce the output is not known. Knowing both the states and transition probabilities, it is therefore possible to calculate the most probable set of inputs used to produce the output.

To apply this model to speech, treat speech as a continuous sequence of discrete states, where each state is a feature vector representing an acoustic signal (either whole words, phonemes or even sub-phonetic features[6]). Assuming that each state is generated from a probabilistic distribution correlated with other states in the model[9] (i.e. probability that one state follows another) and having trained these distributions on known data, the output signal (i.e. the speech signal) can be used to determine the most probable sequence of tokens spoken. These tokens may then be decoded by a language model to construct a transcription[6].

Despite making up much of the research foundational to modern ASR, Markov models have a crucial flaw when applied to speech; parts of speech are dependent on more than just the part immediately before (i.e. $t-1$). For instance, in a presentation discussing *hats* it is unlikely that the word *cat* would be used, despite the phonetics of the word being largely the same.

2.1.3 How Does Modern ASR Work?

Skipping ahead from the mid-1980s to the current day, ASR has moved towards what is known as the 'end-to-end' or 'encoder-decoder' model; at a high level, the input audio is encoded into features, these features are aligned with language and then decoded to produce an output transcript[10]. The key difference between modern approaches and the classical HMM-based approach is the use of widely researched 'machine-learning' techniques, including various forms of neural network[11, 12, 13, 14].

Recent research has proposed a new network architecture called the *Transformer*[15], aiming to reduce the computational complexity of encoder-decoder models by forgoing convolutional or recurrent neural networks (CNNs and RNNs) and instead relying on 'self-attention'. The motivation for the *Transformer* can be understood as follows;

- CNNs (e.g. [16]) and RNNs (e.g. [17]), while popular, have greater per-layer computational complexity than self-attention[15].
- Recurrent neural networks must perform $O(n)$ sequential operations for a sequence length n, whereas self-attention has a constant (i.e. $O(1)$) maximum number of sequential operations, enabling parallel computation[15].
- By allowing each layer in the encoder and decoder to attend to the whole output of the previous layer,

In a multilayer network, self-attention layers are used to build relations between seperate parts of an input sequence by allowing each node (or 'attention head'[18]) to attend to all outputs from the previous layer.

According to the work that introduced it[15], attention in the Transformer architecture is calculated as;

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Where;

- Q is the Query
- ...

2.1.4 Evaluating ASR Systems

2.1.5 Problems in ASR

Despite their ubiquity, modern ASR systems aren't without fault. Cutting edge systems like *(*wav2vec) are touted as being capable of achieving 'greater-than-human' scores on specific datasets[19, 20, 21] such as *LibriSpeech*[22], achieving as low as 1.4% error[23].

A major problem with comes when the data is not 'clean', for example, background noise is present, microphones are far away, the speaker has an atypical speech pattern, etc. In this setting, *wav2vec* achieves much poorer scores with word error rates as high as 65%[24] on the *CHiME6* corpus[25].

2.2 Whisper

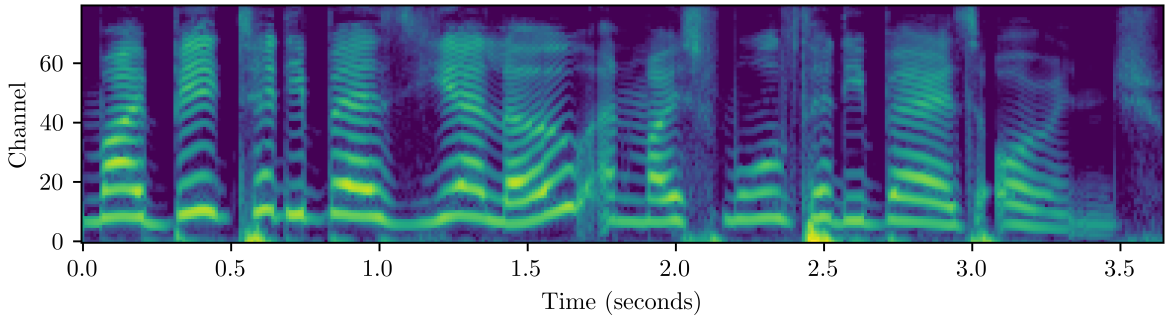
In late September 2022, the OpenAI research laboratory (known for such projects as GPT-3/4 and ChatGPT) released a new open-source ASR system known as 'Whisper' [24]. Whisper is unique in being very large (trained on 680,000 hours of speech data), open-source, and fully supervised; all the training data used to create the model has been accurately labeled and quality-checked by humans, unlike much larger unsupervised (or semi-supervised) models such as 'BigSSL' (1,000,000+ hours of data) [20].

Unsupervised training is appealing for training speech recognisers because there is a wealth of unlabeled recordings, and labeled recordings are uncommon for less widely-spoken languages[26]. Unsupervised systems have a clear disadvantage, however, when compared to supervised; they lack clear decoder mappings[24], meaning that even for a succesfully encoded input there may not be a clear mapping from that input into a speech token. To solve this, fine-tuning to map encodings to decodings is done on the part of the model's developers, though this is a precarious route to overfitting; if the model is too fine-tuned to its training data, performance will suffer when faced with data which isn't well represented in the training set.

For example, if an unsupervised model were trained using the voices of young people, it may perform with considerably poorer accuracy when used to transcribe elderly speakers due to differences inherent to their speech[4].

2.2.1 Operation

Whisper is an encoder-decoder transformer, though it uses 'learned positional encodings' to



Whisper uses a natural language model to perform next-token prediction (in layperson's terms, there is a secondary system trying to ensure the intelligibility of sentences produced from transcription). In a practical setting this means that conversational speech (i.e. speech which flows as sentences rather than semantically-disjoint terms) should be transcribed with a higher degree of accuracy.

Whisper is implemented in Python using the *PyTorch*[27] library which allows computation to take place on GPUs which support CUDA, meaning transcripts can be generated very quickly on a high-performance computer (HPC) system.

2.3 Confidence

2.4 Speech Corpora

2.5 Transcription

2.5.1 Manual Transcription

2.5.2 Fully-Automatic Transcription

2.5.3 Semi-Automatic Transcription

2.6 Summary

Chapter 3

Requirements and Analysis

The objective of this work is not to produce a fully-working, infallible system which aims to receive actual use by transcribers, rather, the aim of this work is to explore the current state of the field of ASR and to understand the extent that current ASR technology could provide aid to a human transcriber. With the purpose of facilitating an extensive evaluation, this chapter shall list the requirements for this work to meet its objective and provide a detailed analysis of each requirement.

3.1 Understand the motivations of computer-aided transcription

Before exploring how a computer system may aid a human transcriber, it is important to understand;

- *why* a human transcriber may require aid;
- to *whom* a computer-aided transcription system would provide benefit; and
- what the *extent* of such a benefit would be.

If this report does not properly motivate computer-aided transcription to a reader, it will have partially failed in its purpose

3.2 Generate transcripts using ASR

Evaluating the quality of ASR transcription requires a key set of data; ASR-generated transcripts. Rather than comparing different ASR systems, Whisper[24] has been chosen as the only system to use for generating transcripts because;

- it is new (made available in September 2022);
- it is entirely free and open-source, meaning it is easily modifiable and available to be used without licence; and

- it reportedly achieves very good results across different speech corpora.

As mentioned in the literature review, Whisper is implemented using *PyTorch*, meaning this work would benefit greatly from access to high-performance GPUs. This would enable fast turnaround times when transcribing large speech corpora and thus enable rapid evaluation and tweaking of settings to minimise erronious results.

The key to generating useful transcripts is some high-quality speech recordings from a speech corpus. While preliminary testing of Whisper may use data from any available corpora, it would be very useful to obtain some data which is;

- not present in Whisper's training data, to prevent the model from regurgitating labels for data it has already seen;
- is well-suited to Whisper's particularities, aiming to maximise the usefulness of results; and
- is representative of real data which would benefit from computer-aided transcription, as to enable more practical evaluation.

Two preliminary aims, therefore, are to understand what kind of data is suited to Whisper and then what kind of data would benefit from computer-aided transription. Once these aims are understood, a suitable dataset may be gathered and used for evaluation, however it is also useful to understand the caveats related to using well-suited data! The naïve assumption that all data seen by *any* computer system is 'perfect' would misrepresent the usefulness of the system in question. To combat this, this work must properly acknowledge the limited extent to which a computer-aided transcription system using Whisper is viable, and evaluate how the viability could be increased to be more applicable to real-world tasks.

3.3 Implement various confidence measures

Neural network confidence is widely discussed in the literature. Considering the aim of this work, it shall focus on re-creating and applying existing measures of confidence to Whisper, whether through modification to the model itself or through inference of the model's output.

If some such modifications fall out of scope of the project, this work would still benefit from a discussion of how those approaches may be applied to a future system and what advantages they may bring.

3.4 Understand the effectiveness of selected confidence measures

Evaluation of the extent to which Whisper may aid human transcription may be done by comparing the accuracy of Whisper's predictions against a reference and the reported model confidence.

This type of evaluation requires taking the following steps to complete;

1. Selection of suitable measure(s) of system accuracy
2. Extensive normalisation of text to facilitate accurate measurements
3. Visualisation of results

The standard across surveyed literature is *word error rate* (WER), despite potential limitations. For the sake of evaluation, other measurements such as *phone error rate* (PER) and *character error rate* (CER) should be calculated alongside WER.

3.5 Explore designs for computer-aided transcription

It would be of great utility to understand how system confidence may aid a human transcriber as this would further refine the 'lens' through which the system may be evaluated, and as such is vital to the completion of this work. There's limited use in a purely theoretical exploration of a computer system such as this, which is designed to be interfaced with by a human. Instead, demonstrating the benefits of a computer-aided transcription system would be easily facilitated using a graphical program.

A number of considerations are required for the design of this system specifically due to its intended nature to serve as an example rather than a final implementation, including;

- several design iterations should be produced to demonstrate different extents of computer aid;
- each design decision should be discussed thoroughly to demonstrate the intended effect and any noteable caveats; and
- a suitable number of screenshots are required in the appendix to demonstrate use of the system.

3.6 Evaluation

3.7 Ethical, Professional and Legal Issues

Chapter 4

Design

4.1 Speech Data

According to its authors, Whisper's robustness is due likely in part to its use of a language model in its decoder[24]. Though likely beneficial for keeping track of sentence context, this poses a potential threat to the models accuracy in a number of circumstances, including;

- Misspoken words or sentences with improper syntax (e.g. 'then' instead of 'than'), for these errors may be corrected by the model, despite being innacurate to the original recording.
- Disjoint terms (e.g. 'book purple dishsoap'), as such terms are highly unlikely to occur in sequence and thus the language model will not consider them a probable output.

To combat these drawbacks, this work will use a conversational speech corpus rather than one made of spoken disjoint terms. While not representative of all speech, an argument can be made that the majority of speech which must be transcribed (e.g. conference recordings, courtroom hearings, lectures, etc.) has a maintained context throughout and is not significantly formed of disjoint terms, though the potential for disjoint terms to be present in a recording should be acknowledged as a potential area of weakness for Whisper.

While introducing the corpus selected for this work that the original objective was to explore the impact of age-related changes to speech on ASR performance, and for this goal the *LifeLUCID* corpus[28] was determined to be the best match. Despite the scope of the project having since changed, the data gathered fits the new objective very well, as it is formed of 52 recordings of conversations between 104 discrete speakers aged between 8 and 85 years old. They are solving a 'spot-the-difference' task, and the data selected for this work was recorded in normal conditions (that is, they can hear and communicate with eachother normally).

The corpus' authors mention that the reference transcripts were generated by an ASR system and only one channel's audio was human-corrected. The ability to compare the quality of ASR output to a reference transcript is required to evaluate this work, thus, only

the human-corrected transcript and corresponding audio channel were used to ensure the references are reliable. This leaves 52 10-minute recordings of individual speakers with gaps where the other participant is speaking.

4.1.1 TextGrid Format

The reference transcripts are supplied in *Praat TextGrid* format which is produced by the Praat software suite[29]. The TextGrid format consists of each individual part of speech (words, hesitations, mid-sentence silences, silences when the other participant is speaking etc.) being present in consecutive entries with the time in the recording which they start and finish.

```
intervals [14]:
  xmin = 21.05
  xmax = 21.47
  text = "BUSH"
```

Figure 4.1: Example of an entry in TextGrid format

Figure 4.1 is an example of a single 'interval' in the TextGrid format. A larger example is available in Appendix 7 You may observe that `xmin` and `xmax` denote the points in the recording at which the section starts and ends, and `text` denotes the content of the section. Considering that each entry appears consecutively and that there are over 1,000 in each file, the format is not easily human-readable.

4.1.2 Data Preparation

There are a number of issues with using the data in its original format, including;

1. Whisper struggles to maintain alignment when transcribing longform data[24], so the approximate 10-minute length of each recording requires shortening to maintain system performance;
2. TextGrids are not human-readable; and
3. The output of the ASR system should be stored alongside the reference transcripts to ease evaluation, which is not possible using TextGrids.

The solution to the first problem would best be solved by splitting the conversation recordings into individual utterances. Luckily, the human-evaluated references include metadata which shows the start- and stop-times of each word and non-word part of speech, meaning it can easily be split into individual utterances without using voice activation detection or other automatic techniques. Using the documentation for LifeLUCID it was possible to determine that there are two types of non-speech token;

1. 'Break' tokens – these are tokens which denote the speaker is not mid-utterance; either listening to the other participant or engaged in irrelevant discussion (these latter parts are silenced in the recordings).

2. 'Junk' tokens – these denote either:

   - The speaker has paused (but the other participant is not speaking).
   - A bell or dog bark is being played as part of their task (these are silent in the recording).
   - Hesitations (e.g., 'umm', 'uhhh', etc.)
   - Other non-speech, non-breaking tokens (not specified in their documentation but present in the transcripts).

For the purpose of this work, an utterance is defined as an uninterrupted piece of speech without any long pauses. By providing threshold values for the minimum length of a 'break' token and maximum length of a 'junk' token, the boundaries at which utterances start and end can be easily computed from the reference TextGrids. The utterance boundaries can then be used to extract individual utterances from the longform recordings, resulting in a series of numbered audio files.

The second and third problems can be solved together by changing from the TextGrid format to JSON (JavaScript Object Notation). This format is human-readable[30] and able to hold all the data and metadata required for this work, including a way to reference the original piece of audio it represents.

4.2 Running Whisper

Whisper comes packaged with models of various sizes, requiring between approximately 1 and 10GB of VRAM and an increasing amount of time to produce transcripts. Considering the objective to create an automatic transcription system which uses entirely free and open-source software, it is worth assuming that the individuals or institutions who would benefit the most from this system are those without the resources to rely on professional manual transcription. It follows, then, that these 'target users' would not have access to high-powered computers and instead rely on consumer-grade hardware to generate transcriptions. Thus, for the purposes of this work, the medium English-language model was selected as it requires only 5GB of VRAM and takes approximately half as much time to produce transcripts as the larger models[24].

4.2.1 High-Powered Computing

4.3 Confidence Scoring

Whisper does not have a clear confidence scoring system in its unaltered state. Therefore, for it to be viably used to aid a human transcriber, some method to estimate system

confidence must be implemented. At a high level, there are two sources from which to estimate confidence: Whisper's standard output, and its internal processes. Table 4.1 gives a brief comparison of the benefits and drawbacks associated with each of these sources of confidence;

Score Source	Benefits	Drawbacks
Standard Model Output	Does not require any modification to Whisper.	Only shows an average probability score per utterance.
Model Internal Scoring	Allows access to per-word scoring and the steps the model takes to converge on an output.	Requires modification to Whisper.

Table 4.1: Comparison between sources of model confidence

The following subsections provide a detailed explanation of how each approach works and should help illustrate the benefits and drawbacks of each.

4.3.1 Confidence From Model Output

Though it does not yield a clear confidence score, Whisper does output various statistics relating to its processing of the input data. These include;

- `avg_logprob` – The average of the log token-probability for a segment of speech (discussed further below)
- `temperature` –
- `compression_ratio`
- `no_speech_prob`

4.3.2 Confidence From Model Internals

4.4 A System For Transcription

Chapter 5

Implementation and Testing

5.1 Preparing the Data

5.1.1 Generating Utterances

The contents, beginning, and end of every utterance where computed using the data available in the *TextGrid* files using a Python script named `get_utterances`. This script operates over a directory containing *TextGrid* files, writing out the utterances as files in *JSON* format.

The script also takes as args; a minimum time between tokens required to end the utterance and a maximum pause time allowed within one utterance. These thresholds allow utterances to be fine-tuned by a user, leading to fewer drawn-out or unreasonably short utterances.

JSON was selected due to its ability to be easily read and understood by a human, unlike *TextGrids*. This allowed for simple verification of the data without the need for more specific software to view the files.

5.1.2 Audio Segmentation

Another Python script named `segment_audio` was created to generate audio files for each utterance. Given two directories as input; one containing `.json` files (as output by the `get_utterances` script) and the other containing `.wav` files representing each audio recording, the audio is split along the beginning and end times of each utterance and output to a new directory.

This script uses the *python-soundfile* module[31] to load audio files into *NumPy*[32] arrays. By multiplying the sampling rate of the audio by the start- and end-times of each utterance, the array indices at the start and end of each utterance are computed. Array slices between these indices represent each utterance, which can then be saved to new audio files using the *python-soundfile* module.

5.2 ASR With Whisper

Whisper is available as a Python module named `whisper`[33]. The module features a `transcribe()` function to transcribe audio files given as a parameter to the function and return an object containing the output of Whisper.

Chapter 6

Results and Discussion

Chapter 7

Conclusions

Bibliography

[1] L. R. Rabiner, "Automatic Speech Recognition - A Brief History of the Technology Development," *Scinapse*, Jan. 2004. [Online]. Available: https://www.scinapse.io/papers/187290754

[2] D. W. H., "The vocoder," *Bell. Labs. Rec.*, vol. 18, p. 122, 1939. [Online]. Available: https://cir.nii.ac.jp/crid/1572261551231523968

[3] K. H. Davis, R. Biddulph, and S. Balashek, "Automatic recognition of spoken digits," *The Journal of the Acoustical Society of America*, vol. 24, no. 6, pp. 637–642, 1952. [Online]. Available: https://doi.org/10.1121/1.1906946

[4] W. S. Horton, D. H. Spieler, and E. Shriberg, "A corpus analysis of patterns of age-related change in conversational speech." *Psychol. Aging*, vol. 25, no. 3, p. 708, 2010.

[5] J. K. Baker, *Stochastic modeling as a means of automatic speech recognition.* Carnegie Mellon University, 1975.

[6] Y. Bengio *et al.*, "Markovian models for sequential data," *Neural computing surveys*, vol. 2, no. 199, pp. 129–162, 1999.

[7] X. Dong, W. Cao, H. Cheng, and T. Zhang, "Hidden markov model-driven speech recognition for power dispatch," in *Tenth International Conference on Applications and Techniques in Cyber Intelligence (ICATCI 2022)*, J. H. Abawajy, Z. Xu, M. Atiquzzaman, and X. Zhang, Eds. Cham: Springer International Publishing, 2023, pp. 760–768.

[8] E. B. Dynkin, *Theory of Markov Processes*, T. Köváry, Ed. Pergamon Press, 1960.

[9] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[10] D. Wang, X. Wang, and S. Lv, "An overview of end-to-end automatic speech recognition," *Symmetry*, vol. 11, no. 8, p. 1018, 2019.

[11] M. K. Mustafa, T. Allen, and K. Appiah, "A comparative review of dynamic neural networks and hidden markov model methods for mobile on-device speech recognition," *Neural Computing and Applications*, vol. 31, pp. 891–899, 2019.

[12] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*. PMLR, 2016, pp. 173–182.

[13] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in joint ctc-attention based end-to-end speech recognition with a deep cnn encoder and rnn-lm," *arXiv preprint arXiv:1706.02737*, 2017.

[14] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Mar. 2017, pp. 4835–4839.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[16] N. Zeghidour, Q. Xu, V. Liptchinsky, N. Usunier, G. Synnaeve, and R. Collobert, "Fully convolutional speech recognition," *arXiv preprint arXiv:1812.06864*, 2018.

[17] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *International conference on machine learning*. PMLR, 2014, pp. 1764–1772.

[18] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," *arXiv preprint arXiv:1803.02155*, 2018.

[19] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 12 449–12 460. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf

[20] Y. Zhang, D. S. Park, W. Han, J. Qin, A. Gulati, J. Shor, A. Jansen, Y. Xu, Y. Huang, S. Wang, Z. Zhou, B. Li, M. Ma, W. Chan, J. Yu, Y. Wang, L. Cao, K. C. Sim, B. Ramabhadran, T. N. Sainath, F. Beaufays, Z. Chen, Q. V. Le, C.-C. Chiu, R. Pang, and Y. Wu, "BigSSL: Exploring the frontier of large-scale semi-supervised learning for automatic speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1519–1532, oct 2022. [Online]. Available: https://doi.org/10.1109%2Fjstsp.2022.3182537

[21] Y.-A. Chung, Y. Zhang, W. Han, C.-C. Chiu, J. Qin, R. Pang, and Y. Wu, "W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," 2021. [Online]. Available: https://arxiv.org/abs/2108.06209

[22] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[23] Y. Zhang, J. Qin, D. S. Park, W. Han, C.-C. Chiu, R. Pang, Q. V. Le, and Y. Wu, "Pushing the limits of semi-supervised learning for automatic speech recognition," 2020.

[24] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," arXiv:2212.04356, 2022.

[25] S. Watanabe, M. I. Mandel, J. Barker, and E. Vincent, "Chime-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings," *CoRR*, vol. abs/2004.09249, 2020. [Online]. Available: https://arxiv.org/abs/2004.09249

[26] A. Baevski, W.-N. Hsu, A. CONNEAU, and M. Auli, "Unsupervised speech recognition," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 27 826–27 839. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/ea159dc9788ffac311592613b7f71fbb-Paper.pdf

[27] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[28] O. Tuomainen, L. Taschenberger, and V. Hazan, "LifeLUCID Corpus: Recordings of Speakers Aged 8 to 85 Years Engaged in Interactive Task in the Presence of Energetic and Informational Masking, 2017-2020," *UK Data Service*, May 2021. [Online]. Available: https://reshare.ukdataservice.ac.uk/854350

[29] "Praat: doing Phonetics by Computer," Mar. 2023, [Online; accessed 8. Apr. 2023]. [Online]. Available: https://www.fon.hum.uva.nl/praat

[30] N. Nurseitov, M. Paulson, R. Reynolds, and C. Izurieta, "Comparison of json and xml data interchange formats: a case study." *Caine*, vol. 9, pp. 157–162, 2009.

[31] B. Bechtold, "python-soundfile," 2013, [Online; accessed 8. Apr. 2023]. [Online]. Available: https://github.com/bastibe/python-soundfile

[32] C. R. Harris, K. J. Millman, S. f. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. Fernández del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, 2020.

[33] "openai-whisper," Apr. 2023, [Online; accessed 10. Apr. 2023]. [Online]. Available: https://pypi.org/project/openai-whisper

Appendices

Appendix A

Example of the TextGrid Format

This example serves to illustrate the lack of readability of a TextGrid file. To aid formatting, it is displayed in two columns, though the original file is a long series of *intervals*.

It represents a piece of audio which is under three seconds in length and consists of the text "a bush with a yellow duck on top".

```
intervals [12]:
  xmin = 20.899
  xmax = 20.971783458461772
  text = "SIL"
intervals [13]:
  xmin = 20.971783458461772
  xmax = 21.05
  text = "a"
intervals [14]:
  xmin = 21.05
  xmax = 21.47
  text = "BUSH"
intervals [15]:
  xmin = 21.47
  xmax = 21.66
  text = "with"
intervals [16]:
  xmin = 21.66
  xmax = 21.720024609817834
  text = "A"
intervals [17]:
  xmin = 21.720024609817834
  xmax = 22.1
  text = "SIL"
intervals [18]:
  xmin = 22.1
  xmax = 22.49
  text = "yellow"
intervals [19]:
  xmin = 22.49
  xmax = 22.84
  text = "duck"
intervals [20]:
  xmin = 22.84
  xmax = 23.06
  text = "ON"
intervals [21]:
  xmin = 23.06
  xmax = 23.769
  text = "top"
```

Appendix B

Another Appendix

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu, consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis, feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum. Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi. Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem quam semper libero, sit amet adipiscing sem neque sed ipsum. Nam quam nunc, blandit vel, luctus pulvinar, hendrerit id, lorem. Maecenas nec odio et ante tincidunt tempus. Donec vitae sapien ut libero venenatis faucibus. Nullam quis ante. Etiam sit amet orci eget eros faucibus tincidunt. Duis leo. Sed fringilla mauris sit amet nibh. Donec sodales sagittis magna. Sed consequat, leo eget bibendum sodales, augue velit cursus nunc.