Benchmarking Transformer-Based Transcription on Embedded GPUs for Space Applications

Marika E. Schubert, Alan D. George
NSF Center for Space, High-Performance, and Resilient Computing (SHREC)
University of Pittsburgh
Pittsburgh, PA
Email: {marika.schubert, alan.george}@pitt.edu

***Abstract*—Speech transcription is a necessary tool for backend applications commonly found in voice assistants. Transcription is typically performed using cloud-based servers or custom hardware, but those resources are not always amenable to space environments due to size, weight, power, and cost constraints. Therefore, it is important to determine the performance of and optimal conditions for running transcription on hardware that is feasible for deployment in a space application. This research investigates and evaluates the performance of an optimized version of the wav2vec2 speech transcription engine, the current state-of-the-art model for this domain. The target hardware, the NVIDIA Xavier NX Jetson embedded GPU, was chosen for its modern GPU architecture and small form factor. In addition to examining the input scaling behavior, we evaluate the hyperparameters of the clustered attention optimization, and average power and energy for inference relative to the operating power mode of the device. The clustered attention model outperformed the improved-clustered model for large input sizes, but the original wav2vec2 performed better for small input sizes. The clustered model energy per inference (13.90 J) was less than energy per inference of the improved-cluster model (15.03 J) and the vanilla model (15.85 J). All models meet real-time speech processing requirements necessary to perform onboard inference entirely on a space system.**

***Index Terms*—Automatic speech recognition, benchmarking, GPU, machine learning, optimization, parallel processing**

I. Introduction

As human space exploration pushes towards lunar orbit with the planned Gateway outpost [1], there will be limited connectivity with traditional terrestrial resources. Current operations on the International Space Station rely heavily on ground personnel to assist the orbiting crew with procedures and vehicle operations. For lunar missions, the need for a conversational interface capable of assisting astronauts, similar to a chatbot, will become critical for activities where it is difficult to consult a screen or manual. This chat interface will serve in place of humans in mission control for tasks such as assisting in procedures, helping to locate objects, and relaying information about the vehicle with the crew. Chatbots, like the Amazon Alexa voice assistant and supporting services [2], rely on speech transcription as the enabling technology for their backend, text-based natural language processing (NLP) applications. The vast majority of similar applications are tailored to consumer electronics, which are ill-suited for remote, extreme environments such as those found in space.

Transcription for space applications remains a vital but undetermined piece of such a conversational system. Most systems capable of transcription do so with the aid of cloud servers, including those provided by Google [3], IBM [4], and Amazon [5]. Communication to a lunar (moon-based) spacecraft from Earth, for instance, would incur a minimum of 2.25-second round-trip time. This minimum latency figure would be complicated by other factors such as loss of signal (LOS), where the link for such a communication could be missing due to spacecraft position or satellite availability [6]. This high latency and complexity of communication would reduce usefulness of the voice system and impact crew productivity when real-time feedback was necessary. This issue would be further magnified if this system were deployed in even more remote environments such as Mars.

Without a remote server, the next source of assistance may come from a edge accelerator such as a graphics processing unit (GPU). Currently, there is a general interest in flying GPUs both for their traditional rendering and display use cases, but also for allowing machine learning in remote environments [7]. For reasons of constrained size, weight, power, and cost (SWaP-C), it is valuable to assess the feasibility of running transcription on an embedded GPU.

The speech-embedding framework benchmarked in this research was wav2vec2, which is a state-of-the-art (SOTA) transformer-based model that translates sounds to phonemes, the discrete sounds that make up human speech [8], [9]. Additionally, two optimizations known as fast-transformers and attention clustering are added to reduce runtime [10], [11].

This research benchmarks the runtime of the wav2vec2 model on a representative embedded GPU for space applications. The key contributions of this paper are the comparisons of cluster sizes in wav2vec2 optimizations, insights into power and energy consumption during inference, and the effect of varying GPU power limits on the runtime for this class of transcription model.

II. Related Work

This section describes the current SOTA of speech transcription, both with transformers and more general recurrent

This work was supported by SHREC industry and agency members and by the IUCRC Program of the National Science Foundation under Grant No. CNS-1738783.

structures. It additionally explores some of the models used for complex language tasks and how those models inform the choice of wav2vec2. Moreover, it will introduce the issue of transformer scaling, as well as optimizations chosen to overcome this issue.

While the subject of speech processing for space applications is a niche area, transcription as a whole has a large body of supporting research driven by the push to integrate voice control into consumer electronics and software. Transcription, also described as speech-to-text, is a necessary function for converting audio signals for use in NLP backends designed to accept text as an input. Many of the most prominent transcription applications have, therefore, been created by companies like Amazon, Facebook, and Google, as well as large research labs. Some systems, like IBM's Watson [4], are available via API. Other transcription apps are in precompiled formats for specific hardware, like Google's on-device speech recognition [12]. Open-source code for transcription, however, is beneficial for the development of a local, offline application for space missions.

One of the early demonstrations of end-to-end automatic speech recognition was found in Deep Speech 2 [13]. Deep Speech 2 performed transcription using several layers of recurrent neural networks (RNNs) with connectionist temporal classification (CTC) loss for training. This widely adopted loss metric allows for the translation of sequences that do not have strict labeling alignment between their input and output. Additionally, Amodei et. al. demonstrated a model that could be deployed on GPU servers, but highlighted the viability of a fully trained machine-learning model for speech transcription over one that required individually designed components. This model was originally considered by the authors of this paper, but initial results demonstrated that it was too large and slow for deployment on embedded platforms.

Transcription is considered a sequence-to-sequence learning task. The goal of sequence-to-sequence tasks is to convert a sequential signal (e.g. audio) to a sequential output of a different domain (e.g. a string of text). One prominent sequence-to-sequence structure is known as a transformer [9]. These models consist of an encoder/decoder structure connected by an attention mechanism. While this structure removes recurrence from the learning process, it does require positional encoding to inform the model of the relative position of an input within a sequence, making it inefficient for memory-bandwidth-constrained hardware.

Another model for sequence-to-sequence learning that uses transformers is the Bidirectional Encoder Representations from Transformers (BERT) [14]. Devlin et al. demonstrated the accuracy of their metric on a wide variety of NLP tasks including sentence prediction, understanding, and sentence segmentation in a variety of languages, but not translation. The utility of BERT for Natural Language Understanding (NLU) tasks is also well demonstrated in its adoption by other researchers [15], [16]. However, it was not desirable for this research as it did not include a method of encoding audio to feature vectors.

RoBERTa is one particularly notable BERT-based model [16] for this use case. This model employed the base BERT training and examined optimizations of hyperparameters and more extensive training to achieve SOTA accuracies on several of the benchmarks that BERT had originally demonstrated. RoBERTa achieved this partly by drastically expanding its training data to include five publicly available corpora. RoBERTA additionally rebuilt the BERT base model using the FairSeq repository, which is a tool developed by Facebook for a variety of NLP text generation tasks including transcription [17]. This toolkit is cited in many transformer-based language papers, and is the basis for the structures used in this research.

Wav2vec2 is another transformer-based model developed using FairSeq that is able to achieve SOTA word error rate (WER) on the LibriSpeech corpus using fine-tuning with a small amount of unlabeled data [8], [18]. Baevski et al. were able to achieve this through a mixture of two approaches. First, input data was encoded using convolutional neural networks (CNNs) in a semi-supervised process to determine feature vectors. This reduces the dimension of the input data and provides feature vectors that better represent the underlying language. Second, they perform training using selective input masking to increase the model's ability to generalize from input data in a similar method as the one applied to BERT [14]. This model demonstrates a semi-supervised learning technique that is well suited for applications where there are a relatively small amount of labeled data for training. This will likely be the case for space applications as much of the spoken language will be jargon and acronyms unique to this domain. There are example models for wav2vec2 available as a starting point, which could be fine-tuned on more powerful terrestrial hardware for mission-specific terminology and speakers.

Wav2vec2 still suffers from the scaling issue inherent to transformers, specifically that computation time scales on the order of $\mathcal{O}(N^2)$ where N is the dimension of the input sequence. It is, however, possible to implement transformers with a slight alteration of the underlying equations to reduce the computation time prediction scaling to a factor of $\mathcal{O}(N)$ through an approximation method demonstrated in [10]. One can further optimize transformers using clustered attention [11]. Clustering partitions the input sequences and calculates the centroid of this data to use in place of the entire cluster, introducing a small but bounded error. Vyas et al. augment their clustering algorithm by additionally considering the attention keys that have the highest weights, referring to this algorithm as *improved clustering.* This method is intended to overcome scenarios where there are too few clusters or where the error introduced by clustering is too high. For the purposes of their experiment, the authors ran the RoBERTA translation model [16] using their modified processes and saw only marginally decreased accuracy for many benchmarks using improved clustering, but more complicated tasks saw more significant losses. Additionally, for short sequence lengths, the full model performed inferences faster that the clustered model.

III. Background

This section will describe the optimizations present in the wav2vec2 approximation used in this research. The derivation of transformer models, as well as the exact derivations for these optimizations can be found in the referenced supporting literature [9], [10], [11]. Details about the target device architecture are also provided.

A. Fast-Transformers

The fast, linear transformers discussed in Sec. II are a reformulation of the original transformer that appears more like a RNN and has an runtime that scales linearly with respect to input sequence length [10]. This is done through the application of a feature map which allows the transformer to precalculate some values to reduce the cost of an input query. A typical transformer operation cost scales relative to input size N, the dimension of the queries D, and the dimension of the values M as follows:

$$\mathcal{O}(N^2\ max(D, M)) \tag{1}$$

In [10], the authors replaced the exponential calculation in the softmax kernel with a second-degree polynomial approximation which reduces the complexity to:

$$\mathcal{O}(ND^2M) \tag{2}$$

Eq. 2 is preferable to Eq. 1 when $N > D^2$, which is true when the input sequence is very large. With wav2vec2, the CNN feature encoders increase the input samples to vectors thousands of samples long, which is significantly greater than the eight attention heads in the base model, meaning that this optimization is a strong candidate for optimizing wav2vec2.

B. Clustered Attention

Clustered attention is another method for reducing runtime that is employed by this research [11]. This method involves calculating an intermediate matrix reference to the "centroid matrix" which contains the centroids of the clusters. This is then used in place of the original query matrix, reducing the dimension of the queries by a factor equal to the cluster size C. Complexity is then reduced to:

$$\mathcal{O}(NC\ max(D, M)) \tag{3}$$

Note that this is most useful where $C << N$. This is beneficial for an architecture like wav2vec2 which has large input sequences, so most potential cluster sizes should provide measurable speedup.

C. Embedded GPUs

NVIDIA, known for its high-performance consumer- and server-grade GPUs, has an additional line of system-on-module (SoM) GPU platforms. These heterogeneous architectures combine an ARM CPU with an NVIDIA GPU on the same die and packaged in an embedded form factor. The entire system can be constrained to meet the requirements of battery-powered environments. Compared to their server-grade counterparts, these boards attain only a fraction of the memory bandwidth (∼50GB/s vs ∼900GB/s) but also operate at significantly lower power (∼15W vs ∼300W) [19], [20]. For these reasons, embedded GPUs would be able to provide acceleration for sufficiently small or optimized models onboard spacecraft.

TABLE I
Summary of Xavier NX Power Modes

Mode ID	Power Budget (W)	Online CPU Count	CPU Max Frequency (MHz)	GPU Max Frequency (MHz)	Memory Max Frequency (MHz)
0	15	2	1900	1100	1600
1	15	4	1400	1100	1600
2	15	6	1400	1100	1600
3	10	2	1500	800	1600
1	10	4	1200	800	1600

While embedded GPUs exhibit desirable SWaP-C properties, they are also less capable than their consumer- or server-grade counterparts. Most transcription models are trained on high-performance hardware, and therefore there is little data on performance for edge devices. Embedded GPUs have a shared memory path between the CPU and GPU, which limits their performance for memory-bound applications. The most important characteristics of embedded GPUs for the purpose of this research is the shared path to memory and the feasibility of deployment in space applications due to desirable SWaP-C.

This research targeted the NVIDIA Jetson Xavier NX. This system has 6 Carmel ARM-based cores (AArch64 architecture) with an NVIDIA Volta GPU and 8GB of LPDDR4x memory. The GPU portion contains 384 CUDA cores, 48 Tensor Cores, and two Deep Learning Accelerators (DLAs). Tensor Cores are designed to accelerate tensor operations, specifically matrix multiplication [19]. DLAs are a structure that accelerate other deep-learning operations like convolution [21].

The GPU supports five standard power modes. The configurations of these power modes are summarized in Table I. There are two operational power budgets: 10W and 15W. Within each power budget, the main difference between power modes is number of available CPU cores and their operating frequency. This should not have an effect on the GPU operation as the function in question should be taking place exclusively on the GPU. For all modes, it is assumed that if fewer CPU cores are active, a smaller percentage of the power budget is used for the CPU, and the GPU may run at a higher power. All modes were considered as the CPU idle power was assumed to impact the maximum GPU power.

IV. Approach

This research examined an approximation of wav2vec2 based on a fork from the FairSeq repository [10], [11], [17]. This fork augments wav2vec with both linear transformers and

input clustering. This was done on the NVIDIA Jetson Xavier NX GPU.

A. Experiment

Wav2vec2's SOTA accuracy has been demonstrated in literature [8] and so was considered out of scope for this study. The untrained small wav2vec model was used to demonstrate the ceiling for runtimes as model pruning may be training-data dependent. To test the behavior of the model, the input sequence length (32k-450k samples), model type (no clustering, clustering, and improved clustering), degree of clustering (cluster sizes of 50, 100, and 125), and GPU mode were varied. For all clustered models, the "conditional-full" option was used to prevent errors with input sizes that are smaller than predicted by introducing zero padding. The codebase leveraged for this model did not allow for batch size increases, but these are not necessary for this particular application as it targets real-time inference. All calculations were performed with single-precision floating-point.

For each size, model, cluster, and power mode, the runtime was averaged over ten model runs. GPU cache priming runs were not counted in the final results as it is assumed that the GPU cache is either used frequently enough to limit this operation, or infrequently enough that it could be primed prior to use. As a note, cache priming was required when the model was updated, and took between five and ten seconds. Therefore, priming would need to be planned for in practical settings.

Power numbers were collected on a subset of these runs using system calls during 100 evaluations with the largest tolerated data-size (input of 450k samples) in each GPU mode. Power measurements for the Xavier device are collected by using direct system calls to the I2C power monitor, which returns values at mW-precision at a 1-second sampling frequency [19]. These measurements are of the CPU and GPU combined power rail.

B. Evaluation Platform

The software environment was reproduced from the Google Colab script developed by the authors of [10] in an embedded Ubuntu 18.04 installation on the NVIDIA Xavier NX. For this application, Python v3.6.12, PyTorch v1.7.0 for AArch64 (specifically compiled for Jetson platforms) [22], fast-transformers v0.3.0, and the branch of FairSeq by Apoorv [23] were used. Note that audtorch is additionally required to load the model, and v0.6.4 was used. For AArch64 architectures, it was required that llvmlite be compiled from source to support the audtorch dependency numba.

The NVIDIA Jetson Xavier NX is run in all of its default power modes, which have power budgets of 10W and 15W. These modes additionally have varying numbers of active CPU cores. All tests were run with single-precision floating-point values.

V. Results

This section contains data collected on the power and energy consumption, inference time, and effect of cluster size on execution time. These results verify that this model is feasible for space applications from an energy and latency perspective. Additionally, the effect of attention clustering type and size is discussed to inform practical implementations. All error bars presented represent the standard deviation.

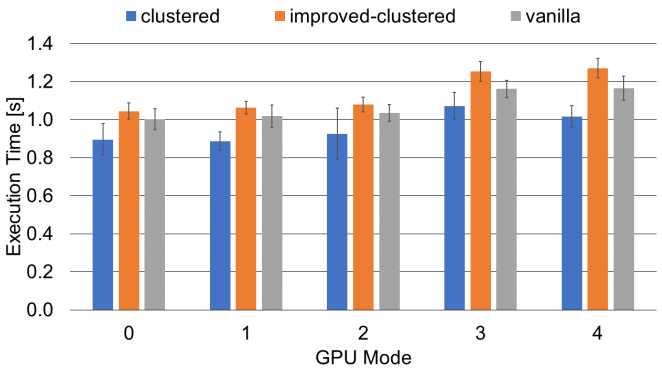

Fig. 1. Effect of GPU power modes on execution time.

A. Effect of GPU Power Modes on Execution Time

For this evaluation, the cluster size was set to 50 for the clustered models, and the times are shown for the 450k-sample input. The effect of the different GPU power modes is shown in Fig. 1. The first three power modes have similar execution times, with the best time seen in the clustered models. However, we also see that when the power budget is reduced to 10W, the execution time of the model does increase, indicating that the GPU is being throttled to meet the power constraint. Note that, in all cases, there are small increases in the runtime as additional cores are brought online. These cores consume power to keep online, reducing power in the budget available for the GPU.

B. Power and Energy

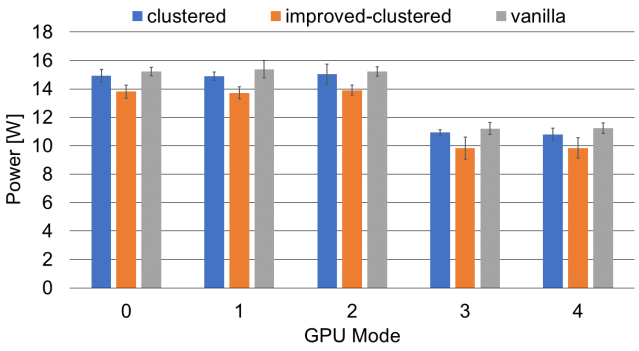

Fig. 2. Effect of GPU power modes on power consumption.

The average power of the varying GPU modes is shown in Fig. 2. This graph shows that in that the power modes do strictly dictate the power of the device. An interesting feature is that the improved-clustered model does not require

the full 15W to run in the higher power modes. This is also the slowest model, meaning that the energy per inference is a more relevant number.

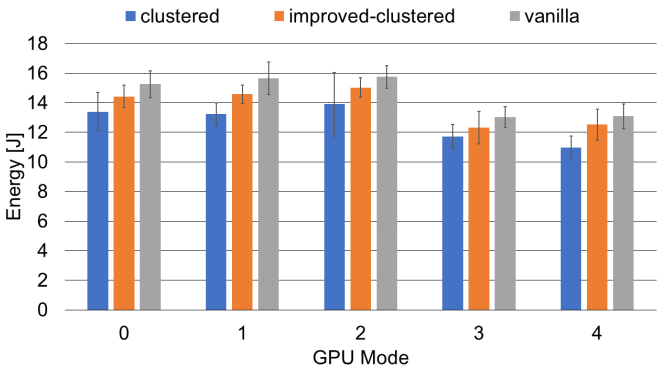

Fig. 3. Effect of GPU power modes on energy per inference.

Energy is shown in Fig. 3. These results were calculated using the average execution times for each inference and the average power consumption during inference at the largest input sample size. In terms of total energy per inference, the vanilla model has a higher energy cost than the other two models. The clustered model had the lowest inference energy, specifically in the 10W power modes. Between the 10W modes, the energy is similar for all models except the clustered model. However, the error bounds for these measurements overlap and the true values are likely more similar than their averages indicate.

C. Linear Scaling

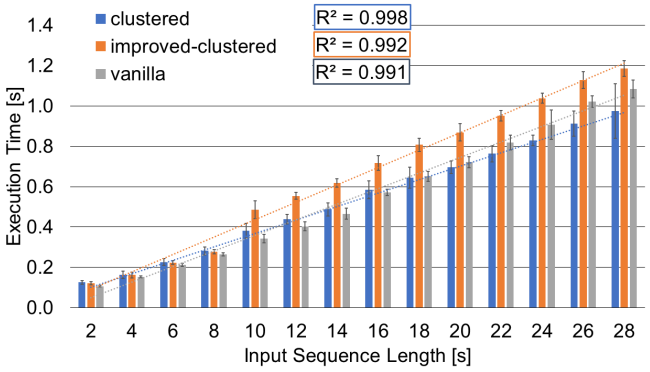

Fig. 4. Execution time compared to input sequence length.

Due to system complexity, it was prudent to confirm that runtime indeed scales linearly with input size for this model, so the results are reported for the vanilla, clustered, and improved-clustered attention models varying input sequence length. These results are shown in Fig. 4. Note that sequence length is listed in time of input audio sample (assuming a 16kHz sampling rate). Within expected input ranges, which were determined by the input size bounds in the LibriSpeech dev-clean dataset, the execution time is linear with coefficients

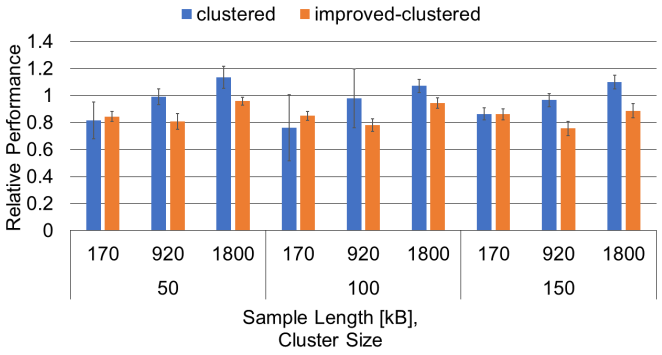

Fig. 5. Effect of cluster size on relative performance (sample size of 450k samples, GPU power mode 2).

of determination exceeding 0.99. The line fits less well for the improved-clustering trend as it executes faster on smaller input sequences and slower on longer sequences. This is likely due to the increased overhead for calculating which of the k input keys will be provided to the model.

From a practical perspective, these are all reasonable execution times for onboard processing. The largest sample is roughly 28 seconds of audio for which inference can be conducted in approximately 1.2 seconds. This sample is significantly longer than a standard query to a chatbot, which is usually on the order of several seconds. Even input sample lengths up to ten seconds executed in under half a second, which is significantly less than the floor for latency of round-trip communication with a ground server. While it is assumed there is still unknown latency associated with other backend systems, this transcription engine is sufficiently quick for other expensive computations to take place and still be advantageous.

D. Cluster Size

The effect of cluster size on relative performance is shown in Fig. 5. This data was collected in the default gpu power mode (2) with a small, medium, and large sample. For this hardware and model, these optimizations only improve performance for very large audio samples. This makes sense as clustering was intended for models with large input vectors. However, it appears for even modest audio samples, the vanilla model is faster.

It is also worth noting that the improved-clustering model is much slower than the base clustered model due to the additional computation of the relevant keys. This trend is also consistent with the performance metrics provided by Katharopoulos et al. on RoBERTa execution [10].

VI. Conclusion

In this research, it is demonstrated that an optimized wav2vec2 model run on an embedded GPU can achieve real-time inference (less than a second for realistic inputs). This is a critical find for adapting transcription for space applications as it demonstrates the feasibility of adapting an

open-source model intended for high-performance hardware to an embedded scale. Additionally, it was shown that the clustered model was a favorable optimization compared to a vanilla and improved-clustered model in terms of energy, but not runtime. The best energy consumption was derived by running the device in its 10W modes (3 and 4) over its 15W power modes (0, 1, and 2) by several joules per inference. It is also advised that these models be run with a small cluster size (50 performed best in these tests). For most expected inputs, the best runtime is achieved with the vanilla model over the clustered model at the cost of optimal energy efficiency. Overall, it was found that the fast-transformers acceleration for wav2vec2 is amenable to the specific architecture of embedded GPUs in terms of speed and power consumption.

Due to software constraints, authors were unable to leverage the Tensor Cores or DLAs on the Xavier NX. Future work includes modifying the software models used to accept automatic mixed precision. This would allow for utilization of the Tensor Cores and improve execution time.

ACKNOWLEDGMENT

This work was supported by SHREC industry and agency members and by the IUCRC Program of the National Science Foundation under Grant No. CNS-1738783.

REFERENCES

[1] NASA, "NASA's Lunar Exploration Program Overview," National Aeronautics and Space Administration, Tech. Rep. September, 2020. [Online]. Available: https://www.nasa.gov/sites/default/files/atoms/files/artemis_plan-20200921.pdf

[2] Amazon Alexa, "Alexa Voice Service v20160207 — Alexa Voice Service," 2016. [Online]. Available: https://developer.amazon.com/en-US/docs/alexa/alexa-voice-service/api-overview.html

[3] "Speech-to-Text: Automatic Speech Recognition — Google Cloud." [Online]. Available: https://cloud.google.com/speech-to-text

[4] G. Saon, G. Kurata, T. Sercu Kartik Audhkhasi, S. Thomas, D. Dimitriadis Xiaodong Cui, B. Ramabhadran, M. Picheny, L.-L. Lim, B. Roomi, and P. Hall Appen, "English Conversational Telephone Speech Recognition by Humans and Machines," *ISCA*, pp. 132–136, 2017.

[5] J. Y. Kim, C. Liu, R. A. Calvo, K. McCabe, S. C. Taylor, B. W. Schuller, and K. Wu, "A comparison of online automatic speech recognition systems and the nonverbal responses to unintelligible speech," 4 2019. [Online]. Available: http://arxiv.org/abs/1904.12403

[6] M. Sanchez Net, "Support of Latency-sensitive Space Exploration Applications in Future Space Communication Systems," Ph.D. dissertation, Massachusetts Institute of Technology, 2017.

[7] F. C. Bruhn, N. Tsog, F. Kunkel, O. Flordal, and I. Troxel, "Enabling radiation tolerant heterogeneous GPU-based onboard data processing in space," *CEAS Space Journal*, vol. 12, no. 4, pp. 551–564, 2020. [Online]. Available: https://doi.org/10.1007/s12567-020-00321-9

[8] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," *arXiv*, 2020.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin, "Attention is All you Need," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5999–6009, 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[10] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, "Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention," *Proceedings of Machine Learning Research*, vol. 119, pp. 5156–5165, 11 2020. [Online]. Available: http://proceedings.mlr.press/v119/katharopoulos20a.html

[11] A. Vyas, A. Katharopoulos, and F. Fleuret, "Fast Transformers with Clustered Attention," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 665–21 674, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/f6a8dd1c954c8506aadc764cc32b895e-Paper.pdf

[12] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shangguan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S.-y. Chang, K. Rao, and A. Gruenstein, "Streaming End-to-end Speech Recognition For Mobile Devices," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2019-May, pp. 6381–6385, 11 2018. [Online]. Available: http://arxiv.org/abs/1811.06621

[13] D. Amodei, "Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin," *International Conference on Machine Learning*, vol. 48, pp. 173–182, 2016. [Online]. Available: http://proceedings.mlr.press/v48/amodei16.pdf

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 1, pp. 4171–4186, 10 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[15] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations," *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/pdf?id=H1eA7AEtvS

[16] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," *arXiv*, 2019.

[17] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "fairseq: A Fast, Extensible Toolkit for Sequence Modeling," in *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[18] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2015-Augus, 2015, pp. 5206–5210. [Online]. Available: http://www.gutenberg.org

[19] Nvidia, "Whitepaper: NVIDIA TESLA V100 GPU ARCHITECTURE: The World's Most Advanced Data Center GPU," Tech. Rep. August, 2017. [Online]. Available: http://www.nvidia.com/content/gated-pdfs/Volta-Architecture-Whitepaper-v1.1.pdf

[20] "NVIDIA Jetson Linux Developer Guide : Clock Frequency and Power Management." [Online]. Available: https://docs.nvidia.com/jetson/l4t/index.html#page/TegraLinuxDriverPackageDevelopmentGuide/power_management_jetson_xavier.html%23wwpID0E0MS0HA

[21] "NVDLA Primer: NVDLA Documentation." [Online]. Available: http://nvdla.org/primer.html

[22] "PyTorch for Jetson," 2019. [Online]. Available: https://forums.developer.nvidia.com/t/pytorch-for-jetson-version-1-7-0-now-available/72048

[23] "apoorv2904/fairseq: Facebook AI Research Sequence-to-Sequence Toolkit written in Python." [Online]. Available: https://github.com/apoorv2904/fairseq