https://doi.org/10.2352/issn.2168-3204.2025.22.1.20

Simple and effective ASR for archives

Anssi Jääskeläinen, South-Eastern Finland University of Applied Sciences, Mikkeli, Finland

Abstract

Archives are traditionally identified as holders of text-based information. However, they also possess audio and video materials, which are the focus of this paper. In archival institutions, the absence of transcriptions for audio and video materials presents significant challenges. These materials often hold historical, cultural, and research value, but without transcriptions, their accessibility and usability are limited. The lack of transcriptions makes it difficult to index and search the content, hindering effective utilization. While existing ASR (Automatic Speech Recognition) technologies can assist, these may suffer from mediocre accuracy, especially with older or poor-quality materials. This work addresses the challenge by utilizing state of the art multilingual LLM (Large Language Model), simple to use UI (User Interface) and GPU (Graphics Processing Unit) ready containers to create a simple and effective multilingual transportable ASR module.

Motivation

The primary motivation for this work is to overcome the inefficiencies of manual transcription and make speech recognition technology more accessible to users without technical expertise. Manual transcription is not only labor-intensive and time-consuming but also impractical for large volumes of audio and video content. This inefficiency is particularly problematic in archival institutions, where the sheer amount of material can make manual transcription an unfeasible solution. By automating the transcription process, this work aims to reduce the time and effort required, thereby enabling more efficient utilization of archival resources.

The secondary motivation is improving accessibility. Audio and video materials often contain valuable historical, cultural, and research information that remain unsearchable or even inaccessible without transcriptions. The availability of open-source ASR tools have opened new possibilities, making speech recognition technology more accessible to researchers, practitioners, and end users. In the broader comparison of ASR technologies, Whisper and Wav2vec stand out for their accuracy while Wav2vec excels in unsupervised learning, allowing it to adapt to various languages and dialects with minimal training data. Kaldi and Kokoro TTS are highlighted for their extensive customization options, which enable users to tailor the systems to specific needs. This flexibility makes them a popular choice in academic research and commercial applications, Kokoro TTS, especially, is favored for generating natural-sounding speech in multiple languages [8]. These tools democratize access to advanced ASR capabilities, allowing institutions with limited resources to benefit from cutting-edge technology without incurring prohibitive costs.

Furthermore, improving accessibility aligns with legal and ethical obligations. For example, in Finland, the Act on the Provision of Digital Services (306/2019), along with the European Accessibility Act (EAA) and the Web Accessibility Directive, mandates that public sector entities provide subtitles for all important audio or video content. This legal framework underscores the necessity of making audio and video materials accessible to all users, including those with hearing impairments. By leveraging open-source ASR tools, compliance with the regulations while also enhancing the overall accessibility of archival materials is achievable.

In addition to legal compliance, there is a broader ethical imperative to make information accessible. Archives serve as custodians of cultural heritage and knowledge, and it is their responsibility to ensure that this information is available to as wide an audience as possible. By improving the accessibility of audio and video materials, this paper contributes to the inclusivity and democratization of knowledge.

The integration of ASR technology into archival workflows also has the potential to transform research practices. Researchers can more easily search and analyze audio and video content, leading to new insights and discoveries. Practitioners in various fields, from education to media production, can also benefit from more efficient access to transcribed materials. End users, including the general public, gain the ability to engage with archival content in new and meaningful ways.

Xamk university and the Digitalia research center are in possession of one of the fastest supercomputers in Finland and its power has been harnessed into R&D use. Digitalia has successfully co-operated with various archival actors in Finland and Europe and is familiar with the old and new challenges faced by the archival community. This suggested ASR solution was designed and implemented as a proof of concept, but it has now been extended into fully dockerized and portable version which is simple to take into use within the archival workflows.

Challenges

Developing a cost-effective but competent ASR system involves several major challenges, especially when archives or other memory institutions are the primary target:

1. **Lack of technical expertise:** Many archival institutions and users lack the technical know-how to develop and implement ASR systems [1]. While the AI model used in this paper is freely available online, creating a functional user interface or API (Application Programming Interface) that leverages this model can be cumbersome without the necessary competency. Implementation and integration into existing systems require knowledge of programming, AI, APIs, and user interface design. This gap in expertise can hinder the adoption of advanced ASR technologies, leaving institutions reliant on outdated or manual transcription methods.
2. **Computational resources:** ASR and AI solutions, even open-source ones, often require powerful hardware, such as computers with dedicated GPUs to run efficiently. GPUs are supercharged processors that can handle numerous calculations quickly by using thousands of parallel cores. However, many archival institutions may not have access to such high-performance hardware, limiting their ability to utilize advanced ASR systems. This challenge is compounded by the need for continuous updates and maintenance of hardware to keep up with evolving AI technologies.

3. **Costs:** Commercial ASR systems are simpler to take into use and utilize but can be expensive, especially for large volumes of data. Archives often operate with limited budgets and are not prioritized when public funds are allocated. The high costs associated with commercial ASR solutions can be prohibitive, making it difficult for archival institutions to adopt these technologies. Additionally, ongoing subscription fees and costs for cloud-based services can add to the financial burden.
4. **Privacy concerns:** Using commercial ASR systems often means sending data to remote servers, which can raise privacy and security issues. This is particularly concerning for archives that contain sensitive or confidential information. Research settings with strict ethical guidelines may prohibit the use of cloud-based solutions due to concerns about data sovereignty and control. Ensuring compliance with these constraints is challenging, as it requires robust data protection measures and adherence to legal and ethical standards.
5. **Transcription quality:** Several factors can affect the accuracy of transcriptions, such as the quality of the recordings, background noise, and the presence of multiple speakers. These issues can lead to errors in the transcriptions, making them less reliable. Older or poor-quality materials, which are common in archival collections, pose significant challenges for ASR systems. The variability in audio quality and the presence of unique accents or dialects further complicate the transcription process.
6. **Integration with existing systems:** Integrating ASR technology into existing archival workflows and systems can be complex. Many archival institutions may have decades-old processes and systems that may not be compatible with new technologies. Compatibility issues with software, databases, and user interfaces must be addressed to achieve seamless integration.
7. **User acceptance**: Even with a user-friendly interface, there may be resistance to adopting new technologies among archival staff. Training and support are essential to ensure that users are comfortable with the utilization of new technology.
8. **Scalability:** Ensuring that the ASR functionality can scale to handle large volumes of audio and video content is crucial. As archival collections grow, the system must be able to process increasing amounts of data efficiently. Scalability involves not only hardware and software capabilities but also the ability to manage and store large datasets. Developing scalable solutions that can adapt to the evolving needs of archival institutions is a significant challenge.

Approach

To address these challenges, the work behind this paper developed a user-friendly proof of concept solution that can run on both CPUs and GPUs, making it easily accessible to users with different types of hardware.

From the beginning of this development work, practicality and ease of use were prioritized over scientific relevancy and influence. This decision was made to ensure that the developed solution could be widely adopted and effectively utilized by a broader audience without technical competence. By focusing on creating a user-friendly and accessible solution, the goal was to lower the barriers to entry, allowing individuals and organizations with varying levels of technical expertise to benefit from this work. This approach also facilitates quicker implementation and integration into existing workflows, maximizing the immediate impact and utility of the solution.

Lack of technical expertise: The first practical selection is the utilization of Docker container, which solves or partially solves challenges 1 to 4. It is recognized that for non-technical people containers might sound like a difficult concept, however these greatly simplify the deployment process thus setting up one is noticeably easier than installing Linux, Python environment, necessary drivers and AI libraries separately. Using containers tackles all version and library issues by creating an isolated environment based on a shared "recipe" called a Dockerfile. This file includes all necessary dependencies, codes, libraries, files, etc. into the container and ensures consistent functionality across different runtime platforms, eliminating compatibility and know-how problems. Furthermore, Docker containers are portable and can be easily shared, ensuring the same experience for every user.

Computational resources: Even though Docker doesn't directly solve the computational resources challenge, running GPU-conformant code within a Docker container is straightforward with the utilization of *--gpus* parameter. This parameter allows Docker container to access the GPU resources of the host machine, enabling efficient execution of tasks that require high computational power. If a GPU is not defined or available, Docker automatically utilizes the CPU to run the ASR code which is conformant with both technologies. This ensures that the solution remains functional even with less powerful hardware and makes it possible to run ASR tasks on a wide range of devices, making the technology more accessible.

Costs: An open-source AI ASR model was selected for this development work for several compelling reasons. The most obvious advantage is that it is free to use, significantly reducing costs compared to commercial alternatives. While open-source models offer numerous benefits, it is important to note that commercial ASR alternatives were not tested in this study. Commercial systems, such as Microsoft's Speech to Text service on Azure, offer robust solutions with potentially higher accuracy and additional features. However, these services can be costly, with for example Azure providing five free audio hours per month and charging 16 to 30 cents per hour thereafter [5]. When dealing with large volumes of audio, the cost difference soon becomes substantial.

Privacy concerns: Docker also helps in addressing privacy concerns by making it possible to run the application / service locally on the user's machine or closed on premise server environment, eliminating the need to transfer sensitive data to remote servers. This ensures that archival materials remain secure and under the user's control, complying with strict ethical guidelines and data sovereignty requirements.

Transcription quality: While this work does not focus on enhancing transcription quality, the solution is designed to be flexible. It is simple to swap a model and replace the included one with a model that has been further trained to better match the content. This is a major advantage of open-source AI models. In practice, this means an ability to fine-tune ASR model on specific datasets, improving accuracy and performance for certain audio, content or language types. Retraining also allows the model to better handle domain-specific terminology, accents, and variations in speech, making them more effective for specialized applications.

By retraining base models, it can be ensured that the ASR system is optimized for the unique requirements, whether it's transcribing historical recordings, academic lectures, or

multilingual content. This capability enhances the overall utility and reliability of the ASR system, providing more accurate transcriptions and better user experience. This kind of adaptability is crucial for archival institutions that may have unique requirements, for example older spoken language or need to process diverse qualities of audio content. This is something that's not commonly available with paid services even with extra costs.

Integration with existing systems: This PoC (Proof of Concept) solution prioritizes ease of integration with existing archival workflows and systems. By using Docker containers, the ASR system can be deployed in a consistent and isolated environment, reducing compatibility issues. The user-friendly interface developed with Gradio ensures that the system can be easily adopted and integrated into existing processes without causing disruptions. In addition to UI, Gradio also creates an API which makes it simpler to embed the functionality into existing API compatible workflows. API utilization of course requires more specific technical knowledge and is out of scope of this paper.

User acceptance: Since this is a PoC solution, no comprehensive user documentation or guides are available, but the GitHub is equipped with two separate simple instructions which clearly defines how the environment is installed within Windows and how the cloned GitHub repository is taken into use.

Scalability: The ASR solution itself doesn't scale up but it utilizes the available host resources quite effectively and the AI model can be replaced with a more performant one if needed. However, the use of containers ensures that the system can be deployed on various hardware configurations, from personal computers to high-performance supercomputers. Additionally, any conformant container can be bound with e.g. Kubernetes which can automatically deploy more resources, do load balancing and replicate or clone the container if needed.

Implementation and utilization

The implemented ASR PoC is based on the Whisper large-v3-turbo model which was selected for its balance of speed, performance, and multilingual capabilities. No scientific comparison was made between the models, just pure rational thinking with the aid of Open ASR Leaderboard by Hugging Face [3]. Metrics such as WER (Word Error Rate) and RTFx (Real Time Factor) were employed to select the most appropriate model. Basically, WER means accuracy and RTFx means speed [2]. Compared to the Whisper large model, the turbo version is faster and noticeably smaller due to fewer decoding layers, making it more efficient for large-scale transcription tasks [3]. Nvidia models, which seem to be both fast and accurate, were dropped due to their requirement for CUDA (Compute Unified Device Architecture, NVIDIA) and this would have been a potential issue for a dockerized solution.

The AI model is loaded using the transformers library from Hugging Face, which provides robust tools for handling AI model operations. Comprehensive functions for loading and preprocessing audio files are also included in the script shared within GitHub [4]. These aim to tackle the possible issues with incompatible audio/video types. Uploaded and preprocessed files are handled in chunks to manage memory usage and improve transcription accuracy.

Developing user interfaces was another important aspect, as the average user is not comfortable using the command line. A well-designed UI can significantly enhance user experience by providing an intuitive and accessible way to interact with the ASR system. Instead of requiring users to input complex commands, the UI allows them to perform tasks through simple, graphical elements such as buttons, sliders, and file upload fields.

In this work a simple Python UI using Gradio for experimenting with the ASR PoC was created. The UI makes it possible to upload files, transcribe audio and see the results with a single button click. The results are received as .json formatted for the programmers to utilize and .srt due to its conformance with major online video services. Gradio which makes this simplicity possible, is a Python library that makes it easy to create interactive web applications for various purposes, including AI model utilization [6]. It allows the creation of web-based interfaces where users can input data, interact with models, and see the results in real-time, without writing any specific UI code such as HTML or TypeScript.

Although the average end user should not need to access the background Python code, the utilization of AI models is still bound to runtime parameters that are used to initialize the model. In the implementation of the ASR system, several key parameters were utilized to optimize performance and ensure flexibility. The best effort has been made to explain the parameters below in a manner accessible to users without technical expertise. The complete codebase, including detailed comments explaining each section, is available on GitHub [4] for those interested in exploring the implementation further.

Data type (dtype) parameter specifies the type of data used for calculations. Common types include float32, which provides high precision, and float16, which uses less memory and speeds up calculations, especially on GPUs.

Chunk length (chunk_length) parameter defines the length of audio segments processed by the model. Smaller segments are more accurate and use less memory but take longer to process. Larger segments process faster but may use more memory and be less accurate.

Batch size (batch_size) parameter determines how many audio segments are processed simultaneously. Larger batch sizes speed up processing but require more memory, while smaller batch sizes use less memory but take longer.

Device (device) parameter specifies the hardware used to run the model, such as CPU or CUDA. Using a CUDA can significantly speed up processing as can be seen on Figure 1, which compares transcribing 2-minute 48-second video with three different devices.

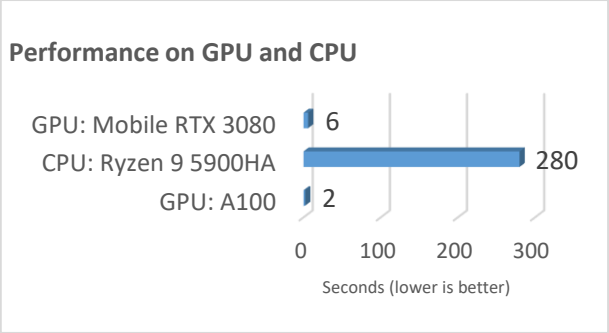

***Figure 1**. Processing times on two GPU:s and one CPU*

Figure clearly indicates that even though the solution is runnable on CPUs it greatly benefits from the parallelization of CUDA cores. So if you happen to have CUDA available feel free

to experiment with the NVIDIA ASR models available in Hugging Face.

Safe tensors (use_safetensors) parameter ensures the use of safe tensors, which aims to prevent memory issues and improves security, making results more reliable.

Low CPU memory usage (low_cpu_mem_usage) parameter optimizes the model to use less CPU memory, which is crucial for running the model on computers with limited resources.

The provided code automatically handles the selection of used device and the data type based on the device. Also, technical parameters like low CPU memory usage and use of safe tensors are automatically set up for the user. The few parameters that might require tweaking are chunk length and batch size, which should be decided according to available system resources.

Simplicity was the main priority, therefore, in addition to functional Docker container and UI, written guidance on installing and running Docker, building containers, and accessing the service via a browser is provided. The simplified instructions for running the app on Windows environment are provided below and the full details within GitHub [4].

1. Install Docker Desktop and ensure it is running correctly
2. Clone the provided GitHub repository [4]
3. Build the Docker image which takes approximately 10 minutes depending on hardware
4. When built is completed, start the image via Docker Desktop and configure a few settings
5. Once the container is up and running, open a web browser and navigate to http://localhost:port

Results and conclusions

A fully functional PoC version is running on our Hippu supercomputer and can be accessed via: https://memorylab.fi/demot/asr/. If the code behind a provided GitHub link is installed in the local environment, the visual experience should be like Figure 2, but there is a chance that before the conference the appearance and functionalities will still change slightly.

The results of this work on automatic speech recognition focus on practical application and experimentation with ASR systems, rather than traditional research findings. WER and RTFx were not measured and the results from transcriptions were only evaluated on real world context. So far, tests have been conducted with random sample materials, such as a two-day seminar recording in both Finnish and English. This seminar recording was transcribed in about six minutes using one A100 GPU card, and the accuracy was deemed suitable for general usage. Not surprisingly, the primary issue encountered was related to the transcription of Finnish names. To address this challenge, the base model can be fine-tuned on a dataset that includes Finnish names and other relevant linguistic features. Fine-tuning involves training a pre-trained model on a new dataset that is more representative of the target domain or language. As of this writing, Hugging Face hosts 228 models that fine-tune the whisper-large-v3-turbo base model, which itself is a fine-tuned version of whisper-large-v3. For example, the whisper-large-v3-turbo-es model, fine-tuned for Spanish audio with the Spanish subset of Common Voice 17.0 dataset, reduced the Word Error Rate (WER) from 6.91% to 5.34%. This approximately 23% improvement in accuracy is a substantial enhancement, however the model page neither tells the time taken for training nor the devices used for the training [7].

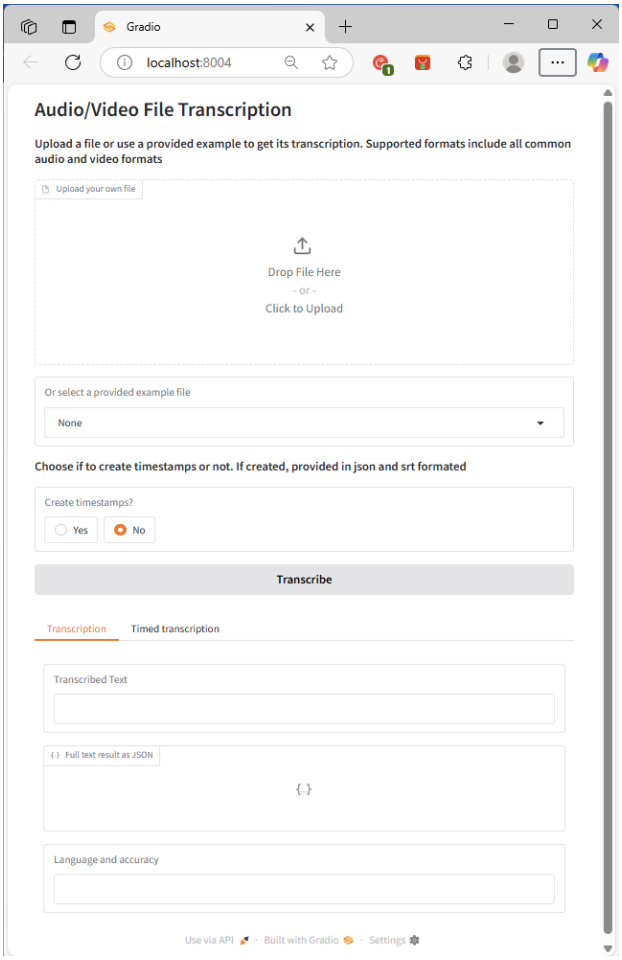

Figure 2. *ASR running on localhost in Edge browser*

Given the practical nature of this work, it holds significant potential for archival sector organizations, which frequently encounter diverse and unpredictable audio sources that require efficient and accurate processing. In addition, the ASR functionality might also be beneficial for all GLAM (Galleries, Libraries, Archives, Museums) sector organizations. By demonstrating the effectiveness of ASR in practical scenarios, this work can attract interest from institutions looking to enhance their digital transformation efforts.

To fully realize the benefits, further testing and refinement of ASR is necessary. Collaboration with interested organizations can provide valuable real-world data and feedback, helping to address specific challenges such as the accurate transcription of names and specialized terminology. Additionally, integrating ASR with other technologies, such as NLP (natural language processing) and machine learning, can enhance its capabilities and adaptabilities.

The results of this study, along with the dockerized ASR solution, were disseminated to the DLM Forum community through a free webinar. Webinar content is not publicly available, but member organizations can access it via member pages. Furthermore, this solution was part of the eArchiving Initiative webinar which recording is freely available [9]. Both webinars served as a platform for the author to present the key findings and demonstrate the practical applications and benefits of utilizing ASR within GLAM sector.

Next development stage for the ASR will be the speaker recognition implementation, which might be available for a live demo during the conference. This functionality will be updated to GitHub when it has been accomplished.

References

[1] C. Goble, S. Cohen-Boulakia, S. Soiland-Reyes, D. Garijo, Y. Gil, M. R. Crusoe, K. Peters, D. Schober, and E. Fairweather, "FAIR Computational Workflows," Data Intelligence, vol. 2, no. 1-2, pp. 108-121, 2020.

[2] N. Sethiya and C. K. Maurya, "End-to-End Speech-to-Text Translation: A Survey," Computer Speech & Language, vol. 78, pp. 101-123, Jun. 2024.

[3] Hugging Face, "Open ASR Leaderboard," Hugging Face, Available: https://huggingface.co/spaces/hf-audio/open_asr_leaderboard. [Accessed: Feb. 21, 2025]

[4] GitHub, “DigitaliaASR” Available: https://github.com/Digitalia-Xamk/DigitaliaASR. [Accessed: Feb 24, 2025]

[5] Microsoft, "Azure AI Speech Pricing," Available: https://azure.microsoft.com/en-us/pricing/details/cognitive-services/speech-services/. [Accessed: May 2, 2025]

[6] Gradio web site, Available: https://www.gradio.app/ [Accessed: May 5, 2025]

[7] Whisper Large V3 Turbo – Spanish, Available: https://huggingface.co/adriszmar/whisper-large-v3-turbo-es [Accessed May 5, 2025]

[8] A. Mohammad, "Orpheus 3B vs. Kokoro TTS: Comparison of Open-Source AI Voice Synthesis Models," Codersera, Available: https://codersera.com/blog/orpheus-3b-vs-kokoro-tts-comparison-of-open-source-ai-voice-synthesis-models. [Accessed: May 6, 2025]

[9] YouTube, "The transformative impact of AI on eArchiving processes," Available: https://www.youtube.com/watch?v=122ZXue4-5Y. [Accessed: May 6, 2025]

Author Biography

PhD Anssi Jääskeläinen is a research manager at South-Eastern Finland University of Applied Sciences (Xamk) in Mikkeli, Finland. He possesses extensive expertise in Open Source and AI development, with a particular focus on implementing AI demonstrations and proof-of-concept solutions using Python and Docker containers. His work is driven by a commitment to making advanced technologies accessible and practical for a wide range of users, including those in archival institutions.