AI-Powered Voice Text Summarization Using Whisper and DeepSeek Coder

1. Introduction: The Landscape of AI-Powered Voice Text Summarization

The proliferation of audio and video content in the digital age has created an unprecedented demand for efficient methods to extract essential information.[1] From lengthy meeting recordings and extensive lecture archives to vast repositories of podcasts and video blogs, the sheer volume of spoken content often leads to information overload.[1] The ability to distill key insights from these sources into concise and easily digestible formats has become a critical necessity for individuals and organizations alike.[1] AI-powered voice text summarization emerges as a transformative technology that directly addresses this challenge by automatically converting spoken words into coherent written summaries.[3] This process not only saves significant time and effort compared to manual methods but also enhances accessibility, searchability, and the overall utility of spoken information.[3]

The significance of AI-powered voice text summarization spans across numerous industries, promising substantial improvements in productivity and information management.[3] In the business sector, this technology can automate the creation of meeting minutes, allowing participants to focus on discussions rather than note-taking.[5] It can also streamline the analysis of business reports and sales calls, extracting key performance indicators and customer insights.[2] Within educational settings, students can benefit from automated lecture summarization, enabling them to quickly review key concepts and enhance their learning.[6] Journalists can leverage this technology to efficiently summarize news broadcasts and interviews, accelerating the process of information dissemination.[11] The healthcare industry stands to gain immensely through the transcription and summarization of patient consultations, which can free up medical professionals from time-consuming documentation tasks and improve the accuracy of medical records.[6] Legal professionals can utilize AI to condense lengthy legal documents and contracts, facilitating quicker analysis and decision-making.[6] Furthermore, voice text summarization plays a crucial role in enhancing accessibility by making audio and video content available to individuals with hearing impairments or learning disabilities.[1] Content creators can also repurpose their audio and video content into various text-based formats like blog posts and social media updates, expanding their reach and impact.[1] The widespread applicability of this technology underscores its potential to revolutionize how we interact with and derive value from spoken information.

Despite its immense potential, implementing robust AI-powered voice text

summarization solutions presents several challenges and considerations.[15] One of the primary hurdles is the inherent complexity of spoken language, which often includes speech recognition errors, disfluencies such as hesitations and repetitions, and a lack of clear sentence structure compared to written text.[7] Ensuring the quality of the generated summaries is also paramount, requiring them to be factually accurate, informative, consistent, and fluent.[7] AI summarization models can sometimes struggle with maintaining context, leading to summaries that are uninformative or even biased, depending on the training data.[18] Technological limitations in handling long documents and maintaining factual consistency throughout the summarization process remain significant areas of research.[19] Moreover, the informal and unstructured nature of speech makes it difficult to directly apply traditional text summarization techniques, which are often designed for well-formed written text.[16] Building effective AI-powered voice text summarization systems necessitates careful consideration of these limitations and the implementation of strategies to mitigate their impact.

2. Exploring Datasets for Training and Evaluation

While the research material highlights the significance and challenges of AI-powered voice text summarization, it does not explicitly enumerate specific datasets designed for the end-to-end task of converting audio directly to summaries. This suggests that datasets specifically curated for this combined task might be less prevalent compared to datasets focusing on the individual components of speech recognition and text summarization. Consequently, researchers and developers may need to adapt existing datasets or potentially create new ones to effectively train and evaluate models for voice text summarization.

For the audio-to-text component, OpenAI's Whisper model can be fine-tuned using several publicly available datasets. The CommonVoice11 dataset stands out as a valuable resource, offering over 16,000 hours of audio data across more than 100 languages.[20] This dataset provides paired audio clips and their corresponding text transcriptions, making it directly suitable for supervised fine-tuning of Whisper to improve its accuracy on specific languages or accents.[20] The LibriSpeech dataset, comprising approximately 1000 hours of high-quality 16kHz English speech, is another widely used resource for developing general-purpose English automatic speech recognition systems.[21] Additionally, datasets like VoxPopuli and CoVoST2 are also mentioned as publicly accessible resources that can be leveraged for fine-tuning speech recognition models.[22] The Hugging Face datasets library provides a convenient and efficient way to access and manage these and other relevant datasets, streamlining the process of data loading and preparation required for

fine-tuning Whisper.[20]

For the text-to-summary component, the user's query indicates the use of the "samsum" dataset for fine-tuning the DeepSeek Coder model. While the research snippets do not provide specific details about this dataset, its application in fine-tuning a summarization model for conversational text implies that it likely contains a collection of dialogue transcripts paired with corresponding human-written summaries. This type of dataset is essential for training a language model like DeepSeek Coder to learn the task of condensing conversations into shorter, informative summaries. The choice of the "samsum" dataset suggests a potential focus on applications such as summarizing meeting discussions or analyzing customer service interactions.

Selecting appropriate datasets for training and evaluation is a critical step in developing an effective AI-powered voice text summarization system. The size and diversity of the dataset significantly impact the robustness and generalization capabilities of the trained models.[20] Ideally, the dataset should closely align with the target domain or application to ensure optimal performance in real-world scenarios.[17] For instance, fine-tuning Whisper on medical transcripts would be beneficial for healthcare applications. When working with Whisper, it is also important to consider datasets that include a variety of accents and speaking styles if the intended application needs to handle diverse audio inputs.[24] For DeepSeek Coder, the quality and characteristics of the summaries within the dataset, such as their style and length, should match the desired output of the summarization model.[14] Furthermore, it is crucial to consider ethical implications and potential biases inherent in the datasets used for both transcription and summarization to ensure the development of fair and responsible AI systems.[5] Careful consideration of these factors during dataset selection is paramount for achieving high performance and ethical outcomes in the overall voice text summarization project.

3. Data Preprocessing: Preparing Audio and Text for AI Models

Effective data preprocessing is a foundational step in building any successful AI system, and voice text summarization is no exception. This stage involves preparing both the audio data for Whisper and the resulting text data for DeepSeek Coder to ensure they are in the optimal format for model training and inference.

For audio data intended for Whisper, several standard preprocessing techniques are typically employed. Whisper models expect audio input to be sampled at a rate of 16kHz.[20] Therefore, if the original audio data has a different sampling rate, it is

necessary to resample it to 16kHz to ensure compatibility with the model.[20] Feature extraction is another crucial step, where the raw audio signal is transformed into a representation that the model can effectively process. Whisper utilizes Log-Mel spectrograms as its input feature format, and its built-in feature extractor automatically handles this conversion.[20] Due to the input length limitations of many transformer-based models, including Whisper, long audio files often need to be segmented into fixed-length chunks, such as 30-second segments.[25] To maintain context across these segments, a small overlap between consecutive chunks can be beneficial.[25] In cases where the audio quality is compromised by noise, noise reduction techniques can be applied to enhance the clarity of the speech and improve transcription accuracy.[27] Additionally, Voice Activity Detection (VAD) can be used to identify and isolate segments of the audio that contain speech, allowing the model to focus on relevant portions and potentially improve both efficiency and accuracy.[28] These audio preprocessing steps are vital for ensuring that Whisper receives input that is correctly formatted and of high enough quality to produce accurate transcriptions.

Once the audio has been transcribed into text by Whisper, further preprocessing is usually required before feeding it into a text summarization model like DeepSeek Coder. Tokenization is the initial step, which involves breaking down the text into individual units, or tokens, such as words or subwords.[3] Both Whisper and DeepSeek Coder have their own specific tokenizers that convert text into numerical representations that the models can understand.[20] Text normalization is another important technique, which aims to standardize the text by converting it to a consistent format, such as lowercase, and handling punctuation marks and special characters.[30] This can improve the consistency of the text and potentially enhance the performance of the summarization model. While the removal of stop words (common words that may not carry significant semantic meaning) is a common preprocessing step in some NLP tasks, its utility in text summarization can vary depending on the specific model and dataset. Finally, encoding the tokenized text into numerical vectors, often using techniques like word embeddings, is a critical step for providing the text data in a format that can be processed by neural network models like DeepSeek Coder.[33] These text preprocessing techniques prepare the transcribed audio for the summarization model by converting it into a suitable numerical format and reducing linguistic variations that might hinder performance.

When considering data preprocessing specifically for fine-tuning both Whisper and DeepSeek Coder, it is essential to align the preprocessing steps with how the models were originally trained.[20] For Whisper, it is generally recommended to use the feature

extractor and tokenizer provided by the Hugging Face library for the specific Whisper model size being fine-tuned.[20] This ensures that the input data is processed in the same way as the pre-training data, which can lead to more effective fine-tuning. Similarly, for fine-tuning DeepSeek Coder, the text preprocessing should be consistent with the format of the dataset used for fine-tuning, such as the "samsum" dataset, and the expected input format of the model. Furthermore, it is important to consider the impact of preprocessing choices on the evaluation metrics used to assess the performance of the models. For example, when calculating the Word Error Rate (WER) for Whisper, normalization steps like lowercasing and removing punctuation are often applied to both the predicted and reference transcripts to ensure a fair and consistent comparison.[31] Therefore, the entire preprocessing pipeline, encompassing both audio and text, should be carefully designed to meet the specific requirements of both Whisper and DeepSeek Coder, particularly in the context of fine-tuning and evaluation, as consistency in these steps is crucial for achieving reliable and meaningful results.

4. Whisper: Unpacking the Architecture and Capabilities for Speech-to-Text

OpenAI's Whisper model represents a significant advancement in the field of automatic speech recognition, demonstrating remarkable capabilities in transcribing spoken language across a wide range of conditions and languages.[23] At its core, Whisper employs a Transformer-based encoder-decoder architecture, a design that has proven highly effective for sequence-to-sequence tasks like speech recognition.[23] The encoder component of the model takes the audio input, typically in the form of a Log-Mel spectrogram, and processes it to extract relevant acoustic features.[23] These features are then passed to the decoder, which autoregressively predicts the corresponding sequence of text tokens.[23] This means the decoder generates the transcription one token at a time, conditioned on the encoded audio features and the previously generated tokens.

A key characteristic of Whisper is its nature as a multi-task model.[39] Beyond simply transcribing speech in the same language, Whisper is also capable of performing multilingual speech recognition, speech translation (translating speech from one language to another), and even language identification.[39] This versatility is achieved through a clever mechanism involving the use of special tokens that act as task specifiers.[37] These tokens inform the decoder about the specific task it should perform, such as transcribing in a particular language or translating to English. The underlying Transformer architecture is particularly well-suited for these tasks as it allows the model to capture long-range dependencies within both the audio and the text, enabling it to understand context and produce accurate transcriptions and

translations even in challenging acoustic environments. The ability to handle multiple tasks within a single model makes Whisper a powerful and flexible tool for a wide variety of speech processing applications.

Whisper is available in several different model sizes, each offering a trade-off between transcription accuracy and computational resource requirements.[25] These sizes range from the smallest "tiny" model, which has 39 million parameters, to the largest "large" model with 1.55 billion parameters.[25] Generally, larger models tend to achieve higher transcription accuracy but require more computational resources, specifically Video RAM (VRAM), and have slower inference speeds.[42] Conversely, smaller models are faster and require less VRAM, making them suitable for resource-constrained environments, but they might exhibit lower accuracy.[60] For applications focused solely on English speech recognition, there are also English-only versions of the tiny, base, small, and medium models, which often demonstrate better performance on English tasks compared to their multilingual counterparts, especially for the smaller model sizes.[43] It's worth noting that the "turbo" model is an optimized variant of the "large-v3" model that offers a faster transcription speed with only a minimal reduction in accuracy.[43] The availability of these diverse model sizes provides users with the flexibility to choose the most appropriate model based on their specific needs and the computational resources available to them.

Size	Parameters (M)	English-only Model	Multilingual Model	Approximate VRAM (GB)	Relative Speed
tiny	39	tiny.en	tiny	~1	~10x
base	74	base.en	base	~1	~7x
small	244	small.en	small	~2	~4x
medium	769	medium.en	medium	~5	~2x
large	1550	N/A	large	~10	1x
turbo	809	N/A	turbo	~6	~8x

Fine-tuning the pre-trained Whisper model on specific datasets or for particular

accents and domains can further enhance its performance. Several publicly available datasets are suitable for this purpose. CommonVoice11, with its vast multilingual audio data, is a good starting point for experimenting with fine-tuning.[20] LibriSpeech, focused on English speech, can be used to improve accuracy on English language tasks.[21] VoxPopuli and CoVoST2 offer additional options for multilingual fine-tuning.[22] For specialized applications, fine-tuning Whisper on domain-specific audio data, such as medical or legal recordings, can lead to significant improvements in transcription accuracy within those specific areas.[24] This requires the preparation of a custom dataset containing paired audio and transcriptions relevant to the target domain.[24] In situations where data is scarce, particularly for low-resource languages or specific vocabulary, synthetic data generated using text-to-speech techniques can also be a valuable resource for fine-tuning.[63] The primary tool for fine-tuning Whisper is the transformers library provided by Hugging Face.[20] This library offers pre-built functionalities for loading Whisper models, feature extractors, and tokenizers, as well as comprehensive tools for the training and evaluation process.[20] The fine-tuning process typically involves using the cross-entropy loss function, a standard approach for training sequence-to-sequence models.[23] Optimizing the fine-tuning process often requires careful adjustment of hyperparameters such as learning rates, batch sizes, and the number of training epochs.[24] Monitoring key training metrics, such as the loss and the Word Error Rate (WER) on a separate validation dataset, is crucial for assessing the progress of fine-tuning and preventing overfitting.[22] Saving model checkpoints periodically during the training process allows for the possibility of resuming training if interrupted and for selecting the model that achieves the best performance on the validation set.[22] The Hugging Face transformers library provides a well-documented and user-friendly framework that significantly simplifies these aspects of fine-tuning Whisper, making it more accessible for researchers and developers to adapt the model to their specific needs.

5. DeepSeek Coder: Analyzing its Role in Text Summarization

DeepSeek Coder is primarily presented as an open-source Mixture-of-Experts (MoE) code language model that has demonstrated performance comparable to leading models like GPT-4 Turbo in code-specific tasks.[68] This model is built upon the foundation of DeepSeek-V2 and has undergone further pre-training on a massive corpus of code, mathematics, and natural language data, totaling 6 trillion tokens.[68] This extensive pre-training has significantly enhanced its capabilities in coding and mathematical reasoning while maintaining strong performance in general language understanding.[68] Notably, DeepSeek Coder boasts an extended context length of 128K tokens, enabling it to process and generate longer sequences of code and text.[68]

While DeepSeek Coder's primary design focus is on code-related tasks, its robust language understanding capabilities, honed through extensive pre-training on a diverse dataset, make it a potential candidate for text summarization tasks as well.[70] The model's architecture, particularly the Mixture-of-Experts approach, could contribute to both efficiency and performance in various natural language processing applications, including summarization. Evidence from the research material suggests that DeepSeek Coder can indeed assist with summarization tasks, among other programming-related activities.[70] Its strong performance in coding and mathematical reasoning benchmarks indicates a sophisticated understanding of language and logic, which are also crucial for effective text summarization.[68] Furthermore, a feature request within the DeepSeek-V3 project specifically proposes the model's use for summarizing chat titles with their content, highlighting its potential for summarizing conversational text.[72]

The user's query indicates that DeepSeek Coder is being fine-tuned on the "samsum" dataset for the purpose of text summarization. Although the research snippets do not provide specific details about the "samsum" dataset, its use in this context suggests that it likely comprises a collection of dialogues paired with their corresponding human-written summaries. This type of dataset is essential for training a language model like DeepSeek Coder to learn the nuances of summarizing conversational text, such as identifying key topics, extracting important information, and generating concise and coherent summaries. Understanding the specific characteristics of the "samsum" dataset, such as the average length of the dialogues, the style and level of detail in the summaries, and the general topics covered in the conversations, would be crucial for analyzing the fine-tuning process and evaluating the expected performance of DeepSeek Coder on this task.

When comparing DeepSeek Coder to other models specifically designed for text summarization, several potential strengths and weaknesses emerge. One of DeepSeek Coder's likely strengths is its strong coding and reasoning abilities, which might translate to a more profound understanding of technical or logically structured text, potentially leading to better summarization in these areas. Its extended context window of 128K tokens could also be a significant advantage when summarizing longer documents, allowing it to retain more context than models with shorter context lengths.[68] Additionally, as an open-source model, DeepSeek Coder offers greater transparency and potential for customization. While the Wiz Research report [88] highlighted a past security vulnerability related to an exposed DeepSeek database, it's important to note that this issue was promptly addressed, and it primarily concerned the DeepSeek-R1 model, not necessarily the Coder variant. Potential weaknesses of

DeepSeek Coder for text summarization might include a possible bias towards code-related language and structures, which could affect its performance on more general or narrative text if not appropriately fine-tuned. Its performance on a diverse range of text types compared to models specifically trained and evaluated on text summarization benchmarks would need to be thoroughly assessed. Overall, DeepSeek Coder presents a unique combination of capabilities that make it an interesting candidate for text summarization, particularly for technical and conversational content, but its effectiveness compared to dedicated summarization models warrants further investigation.

6. Fine-tuning in Practice: A Detailed Examination

The practical application of AI-powered voice text summarization using Whisper and DeepSeek Coder heavily relies on the effective fine-tuning of both models. Analyzing the provided Python script for fine-tuning DeepSeek Coder on the "samsum" dataset is crucial for understanding the specific implementation details of adapting this model for conversational text summarization. This analysis would involve examining the libraries utilized within the script, such as the Transformers library from Hugging Face and the datasets library. It would also entail identifying the specific pre-trained DeepSeek Coder model being used as the starting point for fine-tuning. A key aspect of the analysis would be to understand how the "samsum" dataset is loaded, preprocessed, and formatted for training. Furthermore, the script's fine-tuning parameters, including the learning rate, the number of training epochs, and the batch size, would need to be carefully examined to understand the training configuration. The training loop itself, as well as any evaluation steps included in the script to monitor the model's performance during fine-tuning, would be important to analyze. Finally, understanding how the fine-tuned model is saved and how it can be subsequently used for generating summaries is essential for the practical deployment of the system. A thorough understanding of all these components of the fine-tuning script is paramount for replicating the process, experimenting with different configurations, and identifying potential areas for optimization to improve the summarization performance of DeepSeek Coder on conversational data.

In addition to analyzing the DeepSeek Coder fine-tuning process, researching potential strategies and techniques for fine-tuning Whisper is equally important for optimizing the audio-to-text component of the system. The Hugging Face transformers library serves as the primary resource for this endeavor.[23] The blog post titled "Fine-Tune Whisper For Multilingual ASR with 🤗 Transformers" provides a comprehensive, step-by-step guide on how to perform this fine-tuning.[23] The initial steps involve preparing the necessary environment by installing required Python

packages, including those for handling audio datasets, training transformer models, and evaluating performance.[23] Subsequently, the appropriate dataset for fine-tuning needs to be loaded, such as CommonVoice11 for multilingual ASR or a domain-specific dataset for specialized applications.[23] Preparing the data involves loading the Whisper feature extractor and tokenizer, which are specific to the chosen Whisper model size.[23] These components are then combined into a WhisperProcessor for easier handling of both audio and text data.[23] A function is typically defined to process each audio sample and its corresponding transcription, ensuring the audio is resampled to the correct rate (16kHz) and converted into the Log-Mel spectrogram format expected by Whisper, while the text is tokenized into numerical IDs.[23] The fine-tuning process itself begins by loading a pre-trained Whisper checkpoint from the Hugging Face Hub using the WhisperForConditionalGeneration class.[23] A data collator, such as DataCollatorSpeechSeq2SeqWithPadding, is defined to handle the batching and padding of the preprocessed data.[23] The Word Error Rate (WER) is commonly used as the evaluation metric, and a function to compute this metric from the model's predictions is defined.[23] The training arguments, including the output directory, batch size, learning rate, and evaluation strategy, are configured using the Seq2SeqTrainingArguments class.[23] Finally, the Seq2SeqTrainer class is initialized with all the necessary components, and the training process is launched using the trainer.train() method.[23] Techniques such as using a smaller learning rate, adjusting batch sizes based on available resources, and employing gradient accumulation can be crucial for successful fine-tuning.[24] Monitoring the WER on a validation set throughout the fine-tuning process is essential for assessing the model's progress and preventing overfitting.[23] The Hugging Face transformers library effectively streamlines these complex steps, providing a user-friendly and well-documented framework for fine-tuning Whisper on various datasets and for different speech recognition tasks.

7. Evaluating Performance: Metrics for Whisper and DeepSeek Coder

Evaluating the performance of both Whisper and DeepSeek Coder is crucial for assessing the effectiveness of the AI-powered voice text summarization system. Different metrics are relevant for each model, reflecting their distinct tasks of speech-to-text transcription and text-to-summary generation.

For Whisper, the primary evaluation metric is the Word Error Rate (WER).[21] WER is a widely used measure of the accuracy of automatic speech recognition systems.[21] It is calculated by summing the number of errors—specifically, substitutions (incorrect words), insertions (extra words), and deletions (missing words)—and dividing this sum by the total number of words in the original, reference transcript.[32] The resulting value

represents the proportion of words that were incorrectly transcribed. A lower WER indicates higher accuracy, meaning fewer errors were made by the speech recognition system.[21] For example, a WER of 20% implies that 80% of the words in the transcription are correct.[74] It is important to note that the WER can sometimes exceed 1.0 if the number of inserted words is greater than the number of correct words in the reference.[35] While WER is a valuable and widely adopted metric, it has certain limitations.[31] It treats all errors equally, regardless of their impact on the overall meaning or readability of the transcript.[31] Additionally, it does not provide insights into the underlying reasons for the errors.[73] Factors such as the quality of the audio recording, the speaker's accent, and the presence of background noise can significantly influence the WER.[73] For a fair and consistent calculation of WER, it is often necessary to normalize the text by converting it to lowercase and removing punctuation from both the predicted and reference transcripts.[31] Understanding the calculation and interpretation of WER, along with its inherent limitations, is crucial for accurately evaluating the performance of Whisper and for making informed decisions about potential improvements to the speech recognition component of the voice text summarization system.

For evaluating the performance of DeepSeek Coder in text summarization tasks, common metrics include ROUGE (Recall-Oriented Understudy for Gisting Evaluation) and BLEU (Bilingual Evaluation Understudy).[15] ROUGE is a set of metrics that measure the overlap of n-grams (sequences of words) between the summary generated by the model and one or more human-written reference summaries [Inferred]. Commonly used ROUGE metrics include ROUGE-1, which measures the overlap of unigrams (single words); ROUGE-2, which measures the overlap of bigrams (two consecutive words); and ROUGE-L, which measures the overlap of the longest common subsequence between the generated and reference summaries. Higher ROUGE scores generally indicate a better quality summary in terms of the amount of content from the reference that is captured in the generated summary. BLEU is another widely used metric that assesses the similarity between the generated text and the reference text, primarily by calculating the precision of n-grams, typically up to a length of 4 [Inferred]. Higher BLEU scores suggest that the generated text is more fluent and more similar to human-written text in terms of word choice and ordering. Both ROUGE and BLEU are commonly employed in the field of text summarization to quantitatively evaluate the performance of models, including abstractive models like those that DeepSeek Coder might be adapted for. These metrics provide valuable insights into how well the summaries produced by DeepSeek Coder align with human expectations in terms of content overlap and linguistic quality, making them essential tools for comparing different summarization approaches and for tracking progress during the

fine-tuning process.

8. Workflow Integration: Analyzing the MAIN.PY Script

Understanding the complete workflow of the AI-powered voice text summarization system necessitates a detailed analysis of the MAIN.PY script, which orchestrates the interaction between Whisper for transcription and DeepSeek Coder (via LM Studio) for summarization. This script likely handles the initial audio input, potentially from a file or a microphone. It would then invoke Whisper to transcribe the audio, and the choice of the specific Whisper model size (e.g., tiny, base, small, medium, large) for this transcription task would likely be configurable within the script. The resulting transcribed text would then need to be passed to DeepSeek Coder, which is accessed through LM Studio. This interaction likely involves sending the transcribed text to a specific endpoint provided by LM Studio, which is running a DeepSeek Coder model locally. The script would also need to include a carefully crafted prompt that instructs DeepSeek Coder on how to perform the summarization task on the input text. Finally, the script would handle the output generated by DeepSeek Coder, which is the summarized text, and might involve displaying it to the user, saving it to a file, or further processing it. Analyzing this entire workflow as implemented in the MAIN.PY script is crucial for identifying potential bottlenecks in the process, understanding how the different components are integrated, and pinpointing areas where optimizations or enhancements could be made to improve the overall performance and efficiency of the voice text summarization system.

The query explicitly mentions the use of different Whisper model sizes for transcription, suggesting that the MAIN.PY script likely incorporates a mechanism to select or potentially iterate through various available sizes.[25] This flexibility allows users or the system itself to choose a Whisper model based on the specific requirements of the task, such as the desired level of transcription accuracy versus the acceptable processing time and computational resource usage.[42] For instance, for real-time applications or when running on devices with limited resources, a smaller and faster Whisper model like "tiny" or "base" might be preferred, even if it entails a slight reduction in accuracy. Conversely, for applications where the highest possible transcription accuracy is paramount, even at the cost of increased processing time and resource consumption, a larger model like "large" or "turbo" might be selected. The MAIN.PY script could implement this selection through command-line arguments, configuration files, or even dynamically based on the characteristics of the audio input. This capability to easily switch between different Whisper model sizes provides a valuable means of tailoring the transcription component to the specific needs of

various use cases within the voice text summarization project.

The MAIN.PY script leverages DeepSeek Coder for the task of text summarization, accessing it through the LM Studio application. This implies that LM Studio is running a DeepSeek Coder model locally on the user's machine, and the script interacts with it via an Application Programming Interface (API) or a similar communication protocol. The script would likely take the transcribed text generated by Whisper as input and send it as part of a request to the LM Studio endpoint that is hosting the DeepSeek Coder model. In this request, the transcribed text would be accompanied by a carefully designed prompt that instructs DeepSeek Coder to perform the summarization task. The specific content and format of this prompt are critical factors that will significantly influence the quality, length, style, and focus of the summary generated by DeepSeek Coder.[4] For example, the prompt might specify the desired length of the summary, the key aspects of the transcribed text that should be emphasized, or the overall tone that the summary should adopt. Upon receiving the request, DeepSeek Coder, running within LM Studio, would process the transcribed text according to the instructions provided in the prompt and return the generated summary back to the MAIN.PY script. The script would then handle this summary, potentially displaying it to the user or saving it for further use. The effectiveness of the summarization achieved through this workflow will depend on several factors, including the specific DeepSeek Coder model being used within LM Studio, the quality of the transcribed text from Whisper, and, most importantly, the design and effectiveness of the prompting strategy employed in the MAIN.PY script.

9. Future Directions and Potential Enhancements

The field of AI-powered voice text summarization is continuously evolving, and there are numerous avenues for future research and development to enhance the performance and capabilities of systems like the one utilizing Whisper and DeepSeek Coder. One promising direction involves exploring advanced fine-tuning strategies for both models to achieve even better accuracy and efficiency. For Whisper, this could include fine-tuning on more extensive and diverse domain-specific audio datasets to further improve transcription accuracy in specialized areas such as medicine, law, or finance.[20] Similarly, DeepSeek Coder could be fine-tuned on larger and more diverse text summarization datasets, potentially including a wider range of conversational data beyond the "samsum" dataset, if applicable, as well as datasets covering different writing styles and topics. Experimenting with various fine-tuning techniques, such as different learning rate schedules, batch sizes, and optimization algorithms, as well as exploring the impact of training for a longer duration, could lead to optimized performance for both models.[66] Another interesting area to consider is multi-task

fine-tuning, where Whisper could be trained simultaneously on tasks like speech recognition and language identification, potentially improving its ability to handle multilingual audio. Similarly, DeepSeek Coder could be fine-tuned on summarization along with other related natural language processing tasks, such as question answering or text generation, which might enhance its overall language understanding and summarization capabilities.

Another critical aspect for future enhancement lies in exploring the impact of different prompting strategies on the quality of the summaries generated by DeepSeek Coder.[4] Investigating various prompting techniques, such as providing more detailed instructions, specifying the desired length and style of the summary, or guiding the model to focus on particular aspects of the transcribed text, could lead to more tailored and effective summaries.[4] Experimenting with few-shot prompting, where DeepSeek Coder is provided with a few examples of high-quality summaries along with the transcribed text, might help the model learn to generate summaries that better align with human expectations.[67] Furthermore, exploring the use of chain-of-thought prompting, a technique that encourages the language model to reason through the input text step by step before generating the final summary, could potentially improve the coherence and factual accuracy of the summaries.[24]

Future work could also involve investigating the integration of other complementary AI models to further enhance the voice text summarization pipeline. For instance, incorporating a dedicated noise reduction or audio enhancement model before the transcription stage could significantly improve the accuracy of Whisper, especially when dealing with audio recordings that contain background noise or other acoustic challenges.[27] Exploring the use of other state-of-the-art language models for the summarization task, either as a replacement for or in combination with DeepSeek Coder, could lead to improvements in summary quality and style.[27] Ensemble methods, where the outputs of multiple summarization models are combined, could also be investigated to potentially leverage the strengths of different models. Additionally, integrating models for topic modeling or keyword extraction could provide valuable insights into the key themes and concepts within the transcribed text, which could then be used to guide the summarization process and ensure that the most important information is captured in the final summary.

Finally, as the project progresses, careful consideration should be given to the structure and presentation of the research findings. A logical organization of the presentation, starting with a clear definition of the problem and followed by a detailed explanation of the proposed solution using Whisper and DeepSeek Coder, is essential. Providing comprehensive details on each component, including their architecture,

capabilities, and the fine-tuning strategies employed, will be important for conveying a thorough understanding of the system. The presentation should also clearly outline the evaluation metrics used and the results obtained, highlighting key findings and insights. It is crucial to openly discuss the limitations of the current approach and to clearly articulate potential areas for future improvement. Furthermore, a dedicated section should address the ethical implications and potential biases associated with the datasets and models used in the project. The presentation should conclude with a concise summary of the project's contributions and its potential impact in the field of AI-powered voice text summarization. Using visual aids such as diagrams and tables can significantly enhance the clarity and understanding of complex technical information.

10. Ethical Implications and Bias Considerations

The development and deployment of AI-powered voice text summarization systems raise several important ethical implications and concerns about potential biases that must be carefully considered. One primary ethical concern revolves around privacy, as these systems often involve recording and processing spoken data, which can contain sensitive personal information.[5] Ensuring the security and confidentiality of this data is paramount to protect individuals' privacy. Another significant ethical consideration is the potential for misuse of the technology, such as creating summaries that misrepresent the original content, selectively omit crucial information, or are used to spread misinformation or propaganda.[7] The accuracy of both the transcription (via Whisper) and the summarization (via DeepSeek Coder) is also a critical ethical concern. Hallucinations, where the models generate text that is not present in the original audio or text, can lead to inaccurate or even harmful information, especially in sensitive domains like healthcare or law.[12] Furthermore, the issue of accountability and liability needs to be addressed, particularly in high-stakes applications where errors in transcription or summarization could lead to negative consequences.[12] Establishing clear lines of responsibility and ensuring that there are mechanisms for human oversight and error correction are essential.

Potential biases in the datasets used to train Whisper and DeepSeek Coder can also lead to ethical concerns and impact the fairness and accuracy of the system. Whisper's training data, while extensive, has been noted to have a bias towards English-language audio [36], which could potentially affect its performance on other languages or accents, leading to lower accuracy for speakers from underrepresented linguistic backgrounds.[41] The performance of Whisper can also vary across different accents and dialects within the same language.[12] Similarly, the "samsum" dataset, if it predominantly features conversations from a specific demographic or with a

particular communication style, could introduce biases into DeepSeek Coder's summarization capabilities for conversational text, potentially leading to less accurate or representative summaries for other types of conversations. More broadly, AI summarization models, including DeepSeek Coder, can inherit biases that are present in their training data, resulting in summaries that reflect these biases in terms of the topics they emphasize, the language they use, or the perspectives they present.[18] It is crucial to be aware of these potential biases and to take proactive steps to identify and mitigate them throughout the development and deployment process.

To mitigate biases and ensure the responsible development of AI voice text summarization systems, several strategies can be implemented. One important approach is to use diverse and representative datasets for training and evaluation to reduce language and accent biases in Whisper.[24] Carefully curating and analyzing the "samsum" dataset or other summarization datasets for potential biases related to demographics, topics, or sentiment is also essential. Implementing techniques for bias detection and mitigation during the training and evaluation phases of both models can help to identify and address these issues. For Whisper, considering fine-tuning on specific accents or dialects that are underrepresented in the main training data could improve performance for a wider range of speakers.[20] Incorporating human oversight and review, especially in sensitive applications where accuracy and fairness are paramount, can provide a crucial safety net for catching and correcting potential errors or biases introduced by the AI models.[12] Adhering to the usage policies and recommendations provided by OpenAI for Whisper, particularly avoiding its use in high-risk decision-making contexts without thorough validation, is also a responsible practice.[13] A multi-faceted approach that combines careful data selection, bias mitigation techniques, rigorous evaluation, and human oversight is necessary to foster the ethical and responsible development and deployment of AI voice text summarization technologies.

11. Conclusion: Synthesizing Findings and Charting the Path Forward

This report has explored the landscape of AI-powered voice text summarization, focusing on the potential of utilizing OpenAI's Whisper for audio transcription and DeepSeek Coder (via LM Studio) for text summarization. The analysis indicates that this approach holds significant promise for efficiently extracting key information from spoken content and converting it into concise written summaries. The combination of Whisper's robust speech recognition capabilities, particularly its multilingual support and availability in various sizes to balance accuracy and computational cost, with DeepSeek Coder's strong language understanding and generation abilities, offers a

compelling framework for building such a system.

However, the development and deployment of a reliable AI voice text summarization system also present several challenges. These include the inherent complexities of spoken language, the need for careful data preprocessing for both audio and text, the importance of selecting appropriate model sizes and fine-tuning them effectively on relevant datasets, and the crucial task of evaluating the performance of both the transcription and summarization components using appropriate metrics like WER, ROUGE, and BLEU. Furthermore, the integration of these components into a seamless workflow, as likely implemented in the MAIN.PY script, requires careful consideration of how audio input is handled, how Whisper is invoked, how the transcribed text is passed to DeepSeek Coder via LM Studio, and the strategies used to prompt DeepSeek Coder for effective summarization.

Looking ahead, there are numerous opportunities for future research and development in this area. Advanced fine-tuning strategies for both Whisper and DeepSeek Coder, exploring the impact of different prompting techniques on summarization quality, and investigating the integration of other complementary AI models represent promising directions for enhancing the performance and capabilities of these systems. Moreover, as AI technologies become increasingly integrated into various aspects of our lives, it is paramount to address the ethical implications and potential biases associated with these systems. Careful consideration of privacy concerns, the potential for misuse, the risk of hallucinations, and the need to mitigate biases in the training data are essential for ensuring the responsible and beneficial deployment of AI-powered voice text summarization technologies. Continued research and development in this field will undoubtedly lead to more accurate, efficient, and ethically sound solutions that can significantly improve information access and productivity across a wide range of applications.

Works cited

1. AI Text Summarization Tool: A Helpful Technology - Sonix, accessed on April 28, 2025, https://sonix.ai/resources/ai-text-summarization/
2. Importance & Benefits of Auto Text Summarization – Nowigence Inc., accessed on April 28, 2025, https://www.nowigence.com/importance-benefits-of-auto-text-summarization/
3. (PDF) VOICE TO TEXT SUMMARIZATION USING NLP - ResearchGate, accessed on April 28, 2025, https://www.researchgate.net/publication/390260605_VOICE_TO_TEXT_SUMMARIZATION_USING_NLP
4. What is summarization in ASR? - Gladia, accessed on April 28, 2025,

https://www.gladia.io/blog/what-is-summarization
5. EasyChair Preprint Voice Meeting Using Text Summarization, accessed on April 28, 2025, https://easychair.org/publications/preprint/l1SsV/open
6. 20 Applications Of Automatic Summarization In The Enterprise - Frase, accessed on April 28, 2025, https://www.frase.io/blog/20-applications-of-automatic-summarization-in-the-enterprise/
7. AI Summarization: Use Cases, Challenges, & Solutions - Dialpad, accessed on April 28, 2025, https://www.dialpad.com/blog/why-ai-summarization-is-hard/
8. Voice Memo Summarizer - ScreenApp, accessed on April 28, 2025, https://screenapp.io/features/voice-memo-summarizer
9. Summarizer: Voice Transcriber - Apps on Google Play, accessed on April 28, 2025, https://play.google.com/store/apps/details?id=com.apps42.summarizer.app
10. SpeakNotes - AI-Powered Audio & Video Summary Tool, accessed on April 28, 2025, https://speaknotes.io/
11. Meeting Insights Summarisation Using Speech Recognition, accessed on April 28, 2025, https://ijisrt.com/assets/upload/files/IJISRT23APR2036.pdf
12. OpenAI's transcription tool Whisper makes up words patients have never said, accessed on April 28, 2025, https://www.healthcare-brew.com/stories/2024/11/18/openai-transcription-tool-whisper-hallucinations
13. Researchers say an AI-powered transcription tool used in hospitals invents things no one ever said - AP News, accessed on April 28, 2025, https://apnews.com/article/ai-artificial-intelligence-health-business-90020cdf5fa16c79ca2e5b6c4c9bbb14
14. How to Use an AI Text Summarizer to Repurpose Valuable Content - Typeface, accessed on April 28, 2025, https://www.typeface.ai/blog/how-to-use-an-ai-text-summarizer-to-repurpose-valuable-content
15. (PDF) Recent Advances in Automatic Speech Summarization. - ResearchGate, accessed on April 28, 2025, https://www.researchgate.net/publication/221510313_Recent_Advances_in_Automatic_Speech_Summarization
16. From Text to Speech Summarization - CS@Columbia, accessed on April 28, 2025, http://www.cs.columbia.edu/~smaskey/papers/from_txt_to_speech.pdf
17. AI-driven Text summarization: Challenges and opportunities - Addepto, accessed on April 28, 2025, https://addepto.com/blog/ai-driven-text-summarization-challenges-and-opportunities/
18. AI summarization | Google Cloud, accessed on April 28, 2025, https://cloud.google.com/use-cases/ai-summarization
19. Abstractive Text Summarization: State of the Art, Challenges, and Improvements - arXiv, accessed on April 28, 2025, https://arxiv.org/html/2409.02413v1
20. Fine-Tuning ASR Models: Key Definitions, Mechanics, and Use Cases - Gladia, accessed on April 28, 2025, https://www.gladia.io/blog/fine-tuning-asr-models

21. What is WER, or Why Benchmarks Are Misleading - Gladia, accessed on April 28, 2025, https://www.gladia.io/blog/what-is-wer
22. Fine-Tuning Whisper ASR Models | whisper_finetuning – Weights & Biases - Wandb, accessed on April 28, 2025, https://wandb.ai/parambharat/whisper_finetuning/reports/Fine-Tuning-Whisper-ASR-Models---VmlldzozMTEzNDE5
23. Fine-Tune Whisper For Multilingual ASR with Transformers - Hugging Face, accessed on April 28, 2025, https://huggingface.co/blog/fine-tune-whisper
24. How to Finetune Whisper for Speech-to-Text Transcription - MonsterAPI Blog, accessed on April 28, 2025, https://blog.monsterapi.ai/whisper-speech-to-text-transcription/
25. openai/whisper-large-v3 - Hugging Face, accessed on April 28, 2025, https://huggingface.co/openai/whisper-large-v3
26. Fine-tuning Whisper · openai whisper · Discussion #759 - GitHub, accessed on April 28, 2025, https://github.com/openai/whisper/discussions/759
27. [D] Strategies for improving Whisper/STT performance on challenging audio - Reddit, accessed on April 28, 2025, https://www.reddit.com/r/MachineLearning/comments/1fg8qtb/d_strategies_for_improving_whisperstt_performance/
28. A possible solution to Whisper hallucination · openai whisper · Discussion #679 - GitHub, accessed on April 28, 2025, https://github.com/openai/whisper/discussions/679
29. openai/whisper-medium.en - API Reference - DeepInfra, accessed on April 28, 2025, https://deepinfra.com/openai/whisper-medium.en/api
30. Text Summarization and Conversion of Speech to Text - IRJET, accessed on April 28, 2025, https://www.irjet.net/archives/V9/i11/IRJET-V9I11119.pdf
31. The Problem with Word Error Rate (WER) - Speechmatics, accessed on April 28, 2025, https://www.speechmatics.com/company/articles-and-news/the-problem-with-word-error-rate-wer
32. Enhancing AI Models: Understanding the Word Error Rate Metric - Galileo AI, accessed on April 28, 2025, https://www.galileo.ai/blog/word-error-rate%20-metric
33. (PDF) WhisperSum: Unified Audio-to-Text Summarization - ResearchGate, accessed on April 28, 2025, https://www.researchgate.net/publication/385237340_WhisperSum_Unified_Audio-to-Text_Summarization
34. Open ASR Leaderboard - a Hugging Face Space by hf-audio, accessed on April 28, 2025, https://huggingface.co/spaces/hf-audio/open_asr_leaderboard
35. Evaluation metrics for ASR - Hugging Face Audio Course, accessed on April 28, 2025, https://huggingface.co/learn/audio-course/chapter5/evaluation
36. Whisper Tiny · Models - Dataloop AI, accessed on April 28, 2025, https://dataloop.ai/library/model/openai_whisper-tiny/
37. openai/whisper-base - Hugging Face, accessed on April 28, 2025, https://huggingface.co/openai/whisper-base

38. OpenAI's Whisper speech model - an overview | Amit Bahree's (useless?) insight!, accessed on April 28, 2025, https://blog.desigeek.com/post/2023/02/openai-whisper-overview/
39. whisper/README.md at main · openai/whisper - GitHub, accessed on April 28, 2025, https://github.com/openai/whisper/blob/main/README.md
40. openai/whisper-large - Hugging Face, accessed on April 28, 2025, https://huggingface.co/openai/whisper-large
41. keess/whisper-model-internal - Hugging Face, accessed on April 28, 2025, https://huggingface.co/keess/whisper-model-internal
42. What is OpenAI Whisper? - Gladia, accessed on April 28, 2025, https://www.gladia.io/blog/what-is-openai-whisper
43. openai/whisper: Robust Speech Recognition via Large ... - GitHub, accessed on April 28, 2025, https://github.com/openai/whisper
44. whisper-tiny - PromptLayer, accessed on April 28, 2025, https://www.promptlayer.com/models/whisper-tiny-c1c0
45. openai/whisper-tiny.en - Demo - DeepInfra, accessed on April 28, 2025, https://deepinfra.com/openai/whisper-tiny.en
46. whisper-tiny | AI Model Details - AIModels.fyi, accessed on April 28, 2025, https://www.aimodels.fyi/models/huggingFace/whisper-tiny-openai
47. Whisper Base Model - PromptLayer, accessed on April 28, 2025, https://www.promptlayer.com/models/whisper-base-67d8
48. whisper-base | AI Model Details - AIModels.fyi, accessed on April 28, 2025, https://www.aimodels.fyi/models/huggingFace/whisper-base-openai
49. Run whisper-base Model Locally - Nexa AI, accessed on April 28, 2025, https://nexa.ai/openai/whisper-base
50. Whisper Small English Model - PromptLayer, accessed on April 28, 2025, https://www.promptlayer.com/models/whisper-smallen
51. Whisper Small · Models - Dataloop, accessed on April 28, 2025, https://dataloop.ai/library/model/openai_whisper-small/
52. Whisper-Small-V2 - Qualcomm AI Hub, accessed on April 28, 2025, https://aihub.qualcomm.com/models/whisper_small_v2
53. qualcomm/Whisper-Small-En - Hugging Face, accessed on April 28, 2025, https://huggingface.co/qualcomm/Whisper-Small-En
54. whisper-small | AI Model Details - AIModels.fyi, accessed on April 28, 2025, https://www.aimodels.fyi/models/huggingFace/whisper-small-openai
55. Whisper-Medium-En - Qualcomm AI Hub, accessed on April 28, 2025, https://aihub.qualcomm.com/models/whisper_medium_en
56. whisper-medium.en | AI Model Details - AIModels.fyi, accessed on April 28, 2025, https://www.aimodels.fyi/models/huggingFace/whisper-mediumen-openai
57. Whisper Large V3 · Models - Dataloop AI, accessed on April 28, 2025, https://dataloop.ai/library/model/openai_whisper-large-v3/
58. whisper-large-v3 Model by OpenAI | NVIDIA NIM, accessed on April 28, 2025, https://build.nvidia.com/openai/whisper-large-v3/modelcard
59. Whisper models for automatic speech recognition now available in Amazon SageMaker JumpStart | AWS Machine Learning Blog, accessed on April 28, 2025,

https://aws.amazon.com/blogs/machine-learning/whisper-models-for-automatic-speech-recognition-now-available-in-amazon-sagemaker-jumpstart/
60. Tips for File Processing - Whisper - OpenAI - Research Guides at Emory University Libraries, accessed on April 28, 2025, https://guides.libraries.emory.edu/c.php?g=1442123&p=10711508
61. OpenAI's Whisper: Reading the Fine Print - Deepgram Blog ⚡, accessed on April 28, 2025, https://deepgram.com/learn/whisper-issues-smart-formatting
62. Here's how we optimized Whisper ASR for enterprise scale - Gladia, accessed on April 28, 2025, https://www.gladia.io/blog/heres-how-we-optimized-whisper-asr-for-scale
63. Whisper turbo fine tuning guidance : r/LocalLLaMA - Reddit, accessed on April 28, 2025, https://www.reddit.com/r/LocalLLaMA/comments/1i401lt/whisper_turbo_fine_tuning_guidance/
64. vasistalodagala/whisper-finetune: Fine-tune and evaluate Whisper models for Automatic Speech Recognition (ASR) on custom datasets or datasets from huggingface. - GitHub, accessed on April 28, 2025, https://github.com/vasistalodagala/whisper-finetune
65. spellingdragon/whisper-large-v3-handler - Hugging Face, accessed on April 28, 2025, https://huggingface.co/spellingdragon/whisper-large-v3-handler
66. How to optimise Hyperparameters for Whisper finetuning? - Stack Overflow, accessed on April 28, 2025, https://stackoverflow.com/questions/78184210/how-to-optimise-hyperparameters-for-whisper-finetuning
67. Fine-tuning - OpenAI API, accessed on April 28, 2025, https://platform.openai.com/docs/guides/fine-tuning
68. DeepSeek-Coder-V2: Breaking the Barrier of Closed-Source Models in Code Intelligence, accessed on April 28, 2025, https://arxiv.org/html/2406.11931v1
69. DeepSeek-Coder-V2: Breaking the Barrier of Closed-Source Models in Code Intelligence - arXiv, accessed on April 28, 2025, https://arxiv.org/pdf/2406.11931
70. DeepSeek coder extra data · Issue #1403 · ollama/ollama - GitHub, accessed on April 28, 2025, https://github.com/jmorganca/ollama/issues/1403
71. DeepSeek v3 Tested - Coding, Data Extraction, Summarization, Data Labelling, RAG, accessed on April 28, 2025, https://www.youtube.com/watch?v=cbKr3gfbj0g
72. Summarize Chat Title with Content · Issue #550 · deepseek-ai/DeepSeek-V3 - GitHub, accessed on April 28, 2025, https://github.com/deepseek-ai/DeepSeek-V3/issues/550
73. What is WER? What Does Word Error Rate Mean? - Rev, accessed on April 28, 2025, https://www.rev.com/resources/what-is-wer-what-does-word-error-rate-mean
74. Understanding Word Error Rate (WER) in Automatic Speech Recognition (ASR) - Clari, accessed on April 28, 2025, https://www.clari.com/blog/word-error-rate/
75. Is Word Error Rate Useful? - AssemblyAI, accessed on April 28, 2025, https://www.assemblyai.com/blog/word-error-rate

76. Measuring Speech-to-Text Accuracy: Word Error Rate Explained - Picovoice, accessed on April 28, 2025, https://picovoice.ai/blog/measuring-word-error-rate/
77. Word error rate - Wikipedia, accessed on April 28, 2025, https://en.wikipedia.org/wiki/Word_error_rate
78. WER - a Hugging Face Space by evaluate-metric, accessed on April 28, 2025, https://huggingface.co/spaces/evaluate-metric/wer
79. Word Error Rate reveals the accuracy of the speech recognition system - Spoken, accessed on April 28, 2025, https://www.spokencompany.com/word-error-rate-reveals-the-accuracy-of-the-speech-recognition-system/
80. How Word Error Rate (WER) Works (Calculations And Improvements) - PlayHT, accessed on April 28, 2025, https://play.ht/blog/word-error-rate-wer/
81. Word Error Rate (WER) Explained - Measuring the performance of speech recognition systems - YouTube, accessed on April 28, 2025, https://www.youtube.com/watch?v=hoEWRdHi7dI&pp=0gcJCfcAhR29_xXO
82. What is Word Error Rate (WER)? - Deepgram Blog ⚡, accessed on April 28, 2025, https://deepgram.com/learn/what-is-word-error-rate
83. What is the Word Error Rate (WER) in speech recognition? - Milvus, accessed on April 28, 2025, https://milvus.io/ai-quick-reference/what-is-the-word-error-rate-wer-in-speech-recognition
84. Humanizing Word Error Rate for ASR Transcript Readability and Accessibility - Apple Machine Learning Research, accessed on April 28, 2025, https://machinelearning.apple.com/research/humanizing-wer
85. OpenAI Whisper In-House Transcription: Is It Worth the Cost? - Vatis Tech, accessed on April 28, 2025, https://vatis.tech/blog/openai-whisper-in-house-transcription-is-it-worth-the-cost
86. The risks of OpenAI's Whisper audio transcription model - Baldur Bjarnason, accessed on April 28, 2025, https://www.baldurbjarnason.com/2024/openai-whisper-risks/
87. The Reality Behind OpenAI's Whisper Transcription Accuracy: A Deeper Look | InfluxMD, accessed on April 28, 2025, https://www.influxmd.com/blog/the-reality-behind-openais-whisper-transcription-accuracy-a-deeper-look
88. Wiz Research Uncovers Exposed DeepSeek Database Leaking Sensitive Information, Including Chat History, accessed on April 28, 2025, https://www.wiz.io/blog/wiz-research-uncovers-exposed-deepseek-database-leak