1.5em 0pt

Universiteit Leiden
The Netherlands

Bachelor Computer Science

Creating a business-focused benchmark for automated speech recognition models.

Thomas Vermet

Supervisors:
Tyron Offerman
Natalia Amat Lefort

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

17/08/2026

Abstract

This Thesis investigates the impact of automated speech recognition models' performance on documentation pipelines involving Generative AI. The case study was conducted during an internship at The Salvation Army's AI team, which developed their own software product, 'de LuisterLinie'. This model uses Whisper in combination with GPT-4 to transcribe and summarize conversations. The impact of an ASR model is evaluated based on technical summarization metrics. It is notoriously hard to evaluate summaries without human input, and these metrics are not perfect or aligned 1-to-1 with human-perceived accuracy.

The study was executed through 2 experiments. One measures the accuracy of the ASR models, and one compares those results to those of the summarization metrics. They both achieve this by running the ASR models on a custom dataset and then running those transcripts through GenAI models to create the summaries. The ground truth transcriptions are first generated and then manually corrected using the reference audio, which is stripped from YouTube. These are used to calculate the transcript scores. The summary metrics are calculated using a combination of comparing the summary to the summary generated on the corresponding ground truth and comparing it to the ground truth directly.

The ASR models performed similarly to existing benchmarks, except for Whisper. The correlation between those scores and the summarization scores was weak at best, but at some points significant. But looking at specific GenAI models, it becomes clear that the weaker models are mostly responsible for this correlation, while the stronger ones appear to be more robust to errors.

With just these metrics alone, it was not possible to recommend investing in changing ASR models. Cohere did look good, and it would most likely be better than Whisper. However, the biggest increase in accuracy and reliability would be changing from GPT-4 to a better model, which shows better consistency and overall quality.

Acknowledgements

I would foremost like to thank my supervisor, Tyron Offerman, for providing guidance and support throughout the entire process, especially at the end, making sure I succeeded in finishing my thesis. I would also like to thank the AI Team Leader at 'Leger des Heils', Niek Schreuder, for making the internship possible, as well as all the nice people working on the AI team: Joshua van Schravendijk, Gerwin de Kruijf, Wido van Heemstra, and Lars van der Hooft, for helping me during my stay. It was a great experience to be part of the team.
Finally, to Manuel Mol, a special thanks, as my main point of contact on the team and for helping me the most during my project, and Thijmen Bouman, with whom I did the internship, for bouncing ideas and preparing practical things for the University. I deeply appreciate everyone involved, and I feel lucky to have had the opportunity to be involved in such a project.

Contents

1 Introduction

Automated Speech Recognition (ASR) models are used across many Natural Language Processing tasks. This makes them valuable to businesses seeking AI-driven automation solutions for traditionally human-performed tasks. For instance, it can be used in healthcare in combination with a generative AI (GenAI) model to transcribe meetings between professionals and their clients, summarize them, and extract relevant data for documentation purposes [FER+18]. With state-of-the-art ASR models being released as recently as this year, like Qwen3 [SWG+26] and VibeVoice [PYC+26], this raises the question for any organization looking to incorporate such a model into their process flows: How well would such a model work in our current environment compared to others?

The Salvation Army, a Christian non-profit organization that helps the less fortunate through shelter, food, and rehabilitation programs. Within this organization, many different, smaller parts exist with their own goals and methods, like the branch child protective services. Their goal is to assist children in need by providing necessary care and ensuring their safety, taking appropriate action where needed [Leg26]. Through conversations between healthcare professionals and children, in which the child's current situation is assessed, and, based on that discussion, decisions can be made about how to proceed. But processing the notes and incorporating the relevant data into their documentation systems takes up a lot of time. So to assist workers, 'de LuisterLinie' was created, a software product that records their conversations and summarizes them.

This thesis will examine the current implementation of 'de LuisterLinie', specifically the ASR model that is used, Whisper. Evaluating the value models like Whisper create is hard, especially relative to other potential models. When talking about 'creating value', it is in regard to achieving their goal more effectively by increasing the total care, improving the quality, or cutting costs. De LuisterLinie creates value by speeding up documentation, potentially increasing the number of interviews that can be conducted, or leaving more time to assess them and be more engaged during conversations, thereby increasing quality. Classical methods for evaluating ASR models usually focus on accuracy or retaining semantic meaning, but it lacks any insight into how these ASR models impact the bottom line.

1.1 The problemstatement

'De LuisterLinie' was first introduced in 2024 to address the increased documentation burden faced by its healthcare professionals by reducing the time spent on administrative tasks, reducing manual errors, and increasing worker satisfaction [Mol25]. The software first transcribes conversations in 30-second intervals with Whisper and then summarizes the transcription using GPT-4o by inserting the transcription alongside a prompt [Mol25]. Although improvements, such as saving time and awareness/focus during meetings, were immediately noticeable after rollout, the primary source of dissatisfaction among its users remained the quality and accuracy of the output [Mol25].

The current model was selected based on a broader consensus of the model's high performance at the time. This makes comparing the current model with a potentially better one hard to ground in any real data, let alone justify switching to a new one. Many metrics can be used to calculate

performance, as mentioned before, but these are limited to transcription or summarization accuracy. As the head of the AI team, these are not enough to confidently decide which model to use and why. From a business perspective, where data-driven decision-making is the norm, this is a problem. This requires a benchmark that can assess the actual value different models provide, so real considerations can be made.

This research will attempt to bridge the gap between the many technical scoring systems for ASR models and GenAI models tasked with summarization, and the result at the end of the chain of processes it supports. Many parts of the pipeline can impact the result, such as the ASR model, the GenAI model, and the prompt. But also, the overall cost of implementation, hosting cost, and the size of the models are things that do not necessarily impact the process itself but are relevant to the business side of things. This research can be relevant for any business looking into adopting a documentation pipeline using ASR and GenAI models, specifically in Dutch healthcare, looking for a more comprehensive starting point to the question of which one to use.

There are many different techniques to improve the quality of existing models. These will not be covered in this research. Things like domain-specific fine-tuning [KUS25], prompt engineering for enhanced summarization [VZWH+23], creating a dataset of real conversations, and upgrading the hardware used during conversation will not be explored during this research.

1.2 Research question

This leads to the following question: How can the performance of ASR models be measured based on the value they provide to a business? This question will be answered by answering the 3 sub-questions:

1. How well can state-of-the-art ASR models transcribe Dutch medical conversations based on technical metrics?
2. How strongly are these metrics correlated with the technical summarization metrics?
3. Can the business value these models provide be expressed using the previous findings, and which considerations have to be made when selecting one?

2 Background and related work

There has already been extensive testing and research done on ASR models and documentation pipelines in healthcare that use them. By investigating existing methods and important theory, this section lays the groundwork for a new benchmark.

2.1 ASR models

Most ASR models nowadays are end-to-end, meaning they combine elements like acoustic models, punctuation models, and language models, usually present in the encoder and decoder, into a single model [SXK+20]. With the addition of transformer-based models and large language models, these models have improved immensely in recent years. Making large, single complex systems able to take in raw audio data and output a prediction-based transcription. There are many things to consider when selecting a model.

The inverse real-time factor (RTFx) takes the duration of the input audio and divides it by the total amount of time it took to transcribe it [SZB+25]. Especially when there are multiple users at the same time using the model, this can vastly impact performance and cost.
Not all systems support real-time transcription, meaning they require the entire audio file first (non-streaming ASR). For some cases, like voice assistants and translation tools, streaming is necessary, but it has its disadvantages. Since streaming requires prefetching input data and making multiple potentially redundant requests, it increases computing time and the overall cost [LGY+21].
In addition to speech transcription, another common NLP task for ASR is speaker diarization. It labels transcripts with speaker roles. Sometimes this is supported by large ASR systems such as Whisper through speaker embedding models, but if not, it can be included separately, which costs additional time and computation.
Some ASR systems are only commercially available, meaning they can't be hosted locally or in a closed environment. For healthcare specifically, these are automatically off the table since the data contains personal information of clients that is too sensitive to be sent to third parties. The models have to be open-source, giving the organization full control of the data streams.

There are also inference-specific hyperparameters. Inference is the act of using a model on never-before-seen data with the sole purpose of obtaining output. These have mainly to do with a trade-off between accuracy and speed and computational power. Like a smaller bit-size data type increases memory usage during inference but increases accuracy, batching the data speeds up the process by running multiple calculations in parallel but increases memory usage, or Beam Search that reduces speed but accomplishes a slightly higher accuracy score [WKA+18]. Beam search is an algorithm used in end-to-end systems. Instead of selecting the output with the highest probability at each point, it analyses sequences of multiple words. The optimization of these will not be further explored in this research.

2.2 ASR accuracy metrics

ASR models are tested using a labeled dataset of natural spoken language. This consists of the raw audio data and an accurate transcription, the ground truth (GT). By feeding the audio data

into the model, it generates a transcription, the hypothesis (H). This section will go over existing techniques for measuring the accuracy of the hypothesis and interpreting its results.

First, the Ground Truth and hypothesis are aligned based on the Levenshtein distance, the minimum number of edits (substitutions, deletions, insertions) that would have to be made to transform the hypothesis into the ground truth [ZK12]. The most popular metric is the Word Error Rate (WER), which divides the number of edits by the total number of words in the ground truth [SZB+25]. This is effective for measuring overall performance, but circumstances vary widely across use cases, depending on the language spoken, the use of technical jargon, hardware specifications, noise, etc, making accuracy volatile between domains [GEGK21] and thus requiring better benchmarks [AvEFG21].
Medical Concept WER (MC-WER) takes into account that informal speech in healthcare during consultations, conversations, and meetings contains a large amount of filler words that do not add any meaning. Instead of measuring the accuracy of every single word, it only tests medical terms like medication, diseases, and symptoms. This does require a way to annotate medical terms in the ground truth and hypothesis [AJD24].
Errors can also be rated on a different scale than just 0 or 1. Semantic distance (SemDist) [KAL+21] is calculated by the difference in angle of two multidimensional sentence embedding vectors. This gives a single score between 0 and 1, where lower scores indicate a higher semantic similarity.
There is also a human factor to consider. The perceived accuracy by a person reading the transcript, even though highly correlated, can differ from the accuracy score of technical metrics [MLG11]. Two potential ways of achieving this are side-by-side comparison of small transcripts and rating a transcription on a scale of how well it captures the semantic meaning [GEGK21].

The metrics from the last section are intrinsically domain-specific, given that understanding of text is bound to its subject matter. For these scores to be significant, the models must be tested on datasets that consist of realistic speech that resembles potential real-world scenarios.

Speaker diarization has to be measured separately, in case this is included or relevant for the documentation process. Assuming that the number of people present and their roles (healthcare professional, client) are predefined, a simple Diarization Error Rate (DER) suffices. It works the same as the WER. It overlaps the hypothesis with the ground truth, both containing the role labels, based on the Levenshtein distance. But the edits are now given as false alarms, missed detections, and confusions. The total number of edits is divided by the total number of role labels in the ground truth to get the final score [Bre17].

2.3 Dataset

The metrics from the last section are intrinsically domain-specific, given that understanding of text is bound to its subject matter. For these scores to be significant, the models must be tested on datasets that consist of realistic speech that resembles potential real-world scenarios. A dataset can be distinguished based on several features [SZB+25]:

- **Total Duration (h):** The total duration of every audio sample combined
- **Average Duration (s):** The average duration of an audio sample

- **Language:** The language in which the words are spoken
- **Source:** Where did the samples originate from (meetings, casual conversations, voice commands, audiobooks, etc)
- **Style:** The style of speech used (human-to-robot, spontaneous, read, oratory)
- **Noise:** The type of noise that is created (background noise, noise caused by hardware)
- **Transcription:** How is the transcription formatted (punctuated, normalized, cased)
- **License:** Dictates the limits and constraints under which the set can be used
- **Type:** Data type of the audio (.mp3, .mp4, .wav, etc)

In this case, that would be long conversations mainly in Dutch, but occasionally in English, with frequent use of medical terminology. The speech comes from meetings, is spontaneous and less oratorical, between two to three people, and sometimes with high levels of noise or bad audio quality.
Many datasets cross some boxes, like Common Voice Scripted [Moz26], a Dutch dataset with a broad spectrum of accents and different types of speakers, or DRES [dGZMS26], also a Dutch set with background noise and spontaneous speech spoken by multiple people at the same time. There are also classical larger Datasets in English, like [CAB+05], which is in English, but does contain a large number of meeting recordings that are roughly in the same setting, but not healthcare-related.

2.4 Generative AI models

An important part of the documentation pipeline is summarizing the transcripts and extracting data. GenAI makes use of transformer-based models, which use attention to assign weights based on the in-context importance [VSP+17]. This makes them exceptionally good at summarization and identifying relations between words across sentences. These models can perform many NLP tasks. For summarization, they need to be instructed to do so using a prompt. These prompts can have a major impact on a model's performance, and giving a model more instructions or tweaking the instructions based on the results generally increases accuracy [ZLD+24].

2.5 Summarization metrics

Unlike transcripts, summaries have more factors to consider than just accuracy and efficiency, like faithfulness, compression, and extractiveness [GRB25]. These factors all have their own scores that can be computed, some requiring the original text and some a 'ground truth' reference summary, preferably made by a human. For accuracy, the most basic summarization metric, ROUGE-n, remains widely used to this day due to its simplicity and ability to measure the overlap between the summary and the original by counting recurring sequences of n words [Lin04]. The literature shows ROUGE-2 works well for single-document summaries, which is applicable here. However, it does not take into account how these errors might impact the meaning of the text. Similar to SimDist, BERTscore calculates how semantically alike sequences from both sources are using contextual embeddings [ZKW+19].

Faithfulness metrics are calculated with the original text instead of a reference summary and focus on detecting hallucinations or info that is not present in the transcription in this case. SummaC is a natural language interference model. It works by cutting up the original document and summary into sentences and creating a matrix where each cell is a permutation of a sentence from the original document and the summary. High interference results in a high score. If there is a column with no hits, that means it was probably incorrect and not in the original text [LSBH22]. When correct information is preferred over complete information, this metric is perfect.
Another thing to consider is the compression of a summary. The goal of a summary is to condense the data as tightly as possible. The compression rate is simply the number of tokens in the summary compared to the original text [GNA18]. This metric should be used in combination with others and does not say anything about the text itself or the quality.
Where accuracy metrics look at recurring strings between summaries, extractiveness looks at the original text, how many sequences of tokens match, and their length. These can be used to compute Density and Coverage [GNA18]. This can be considered when retaining the original text as much as possible is important, such as in meeting notes.
Every summary evaluation problem has its own prerequisites, and creating the right solution will require a well-thought-out combination of these metrics. This depends on the target audience and can be identified by engaging with users. A model's ability to comprehend text can also be measured by real people comparing the model's summary [HMP+25]. This, however, is incredibly time-consuming and labor-intensive.

2.6 LLM-as-a-judge

Instead of asking a human to evaluate, a GenAI can be tasked with assigning scores to computer-generated output. It combines human-like judgment and evaluation with the scalability of technical metrics. It works by sending a prompt to a model with instructions. These instructions tell the model how many models it evaluates at the same time (usually one, two, or a list of candidates) and how to rate them (Likert scale, best one, binary scale). This field of artificial intelligence is relatively new and still being heavily researched. The overall reliability of an LLM-as-a-judge depends on the models underlying the probability function, the input being evaluated, and the general context. Even though models like ChatGPT 4 are considered to be quite robust, they still suffer from certain biases that need to be accounted for [GJS+24]. Position bias occurs when a model favors input based on its relative position inside the prompt. This can be accounted for by running the same evaluation twice and swapping the candidates around. This is only relevant when the number of candidates at a time exceeds one. Length bias causes a model to prefer candidates of a certain length, which unfortunately does not have a clear workaround. Others, like citation or authority bias and biases that can be exploited by malicious attacks, are not relevant for this case.
This study incorporates a few methods for improving the overall evaluation results. Few-shot prompting is used, which is unlike zero-shot prompting, a method of generating text by sending consecutive prompts to improve the final output and increase overall quality. By giving the model multiple high-quality example evaluations, it can learn from them and better evaluate the real candidates. Decomposing the evaluation and criteria into small, concrete steps also improves the understanding a model has of the task at hand. Finally, a well-structured output format is required for the model to consistently score each candidate. Picking the best among two candidates, which one is the best, is considered to be the best evaluation technique [GRB25].

3 methodology

3.1 Research design

This research is a single embedded case study that examines the current implementation of 'de LuisterLinie' within the Salvation Army's child protection services. It will mainly focus on the ASR model used, possible alternatives, and their respective performance.
It consists of 2 experiments. The first one uses existing ASR metrics to examine the performance of several ASR models on a domain-specific dataset. The second experiment investigates the relation between those metrics and the quality of the summary.
Finally, a benchmark is created for ASR models used in healthcare documentation that combines both the pipelines used during experimentation.

3.2 Experiment 1: ASR model accuracy on domain-specific data

3.2.1 Gathering data

A good benchmark uses audio with a ground-truth transcript that captures all the features specific to the type of audio in the use case it tries to evaluate, such as noise, type of speech, language, etc. Previous literature research has not led to the discovery of such an all-around dataset that resembles the use case at The Salvation Army. This means that the final dataset will consist of many different sources of data combined.
Most of the dataset will be gathered by searching on platforms such as Hugging Face, Google Scholar, and Kaggle. Most datasets are already categorized based on the features defined in the background theory, but if they are not, this will be done manually.
To create new datasets, there are a few possibilities: performing a script, transcribing existing audio, or recording audio and then transcribing it. Reading from a script gives the ability to perfectly match the features, but it is extremely time-consuming. Transcribing audio gives fewer personalization options, but with large sources like YouTube, there are a lot of possibilities. Recording audio and transcribing it has both downsides, making it hard to match features and to set up.
The goal is to gather as much data as possible with a total length of 10 hours. The order of gathering methods will be searching online, manually transcribing audio, manually recording and reading scripts, and lastly recording general conversations or meetings and transcribing those. The raw audio and the ground truth are stored together in a directory tree sorted per category.
The creation of ground-truth transcripts starts with using an existing model to create a first version. Even though speaker diarization accuracy is not measured, by adding the roles now they can be reused in the future. These preprocessing steps speed up the transcription process compared to doing everything by hand. Finally, this version is manually corrected based on the original audio. The ground-truth will be as close to the original audio as possible, meaning it will include obvious errors, stutters that form real words, filler words, and short layered responses. What is not included in the ground-truth are noises that do not relate to speech, like laughing or coughing. Finally, parts that there is no way of saying for sure what is being said are transcribed as '[unaudible]'. This is extremely rare, but these parts can later be left out of the transcription metrics by removing them and every word in the hypothesis that points to it after alignment.

3.2.2 Model selection

There are many ASR models in the current space of audio transcription. Most have already been benchmarked on general audio datasets with basic metrics. Instead of selecting the top models from these existing benchmarks, every available model that can be found on the ASR leaderboards benchmark will be included regardless of current accuracy performance [hf6].
However, not every model fits the use case at The Salvation Army. By consulting the AI-innovation team's lead developers, a set of constraints will be defined. Filtering all the ASR models through these constraints leaves a reduced list of potential candidates. These will be used for running the actual experiment.

3.2.3 Existing benchmarks

These models have already been benchmarked on large English datasets and even Dutch ones. As part of the first experiment, the results will be compared to those of existing benchmarks. The benchmarks that will be used are the ASR-Leaderboards (version 16-07-2026) [hf6] and the ASR Dutch leaderboards [tvo26]. These consist mostly of the three datasets FLEURS [Goo26], a well-articulated single-speaker dataset of very short fragments, VoxPopuli [Fac26b], shorter fragments of up to a minute of public speakers in parliament, formal and oratory speech, and Multi Lingual Librespeech (MSL) [Fac26a], which are short audiobook readings. All these datasets can be filtered to include only the Dutch or English entries.

3.2.4 Creating the pipeline

By creating a pipeline, new metrics, data, and models can be used in the future, incorporating them with ease. The dataset is split up into categories, and each category into an audio part and a transcript part.
For experiment 1, there are 2 separate scripts: one for running the models, and one for calculating the scores. The models are downloaded and set up in such a way that they take in an audio file of type '.wav' or '.mp3' and put out a transcript, the time it took to process, and a summary of the GPU, CPU, and memory usage during the calculations. The process time is used to calculate the RTFx. Some models batch audio internally, which can lead to errors due to sentences being cut up halfway. Others require the audio to be batched beforehand or cannot handle large audio files entirely. To fix all these problems, a built-in torch VAD model is used. This batches the audio into whole sentences. These are then grouped to create up to 45-second segments. Whisper has its own build in implemantation of this model, so it receives the original audio. This ensures consistent transcription circumstances for each model instead of variance caused by internal chunking, as well as making every model able to deal with large audio files. The code is written in Python with inference on a CUDA-type architecture in mind. The inference hyperparameters are consistent throughout. The batch size was set to 2, which sped up the program slightly while not resulting in a memory overflow. This did occur at 4 since the code loads the ASR model in its entirety onto the GPU's memory. Because of the problem with memory, the dtype was set to float16, and Beamsearch was turned off. Loading the model onto the CPU and optimizing these properties can result in higher accuracy scores in the future.

The second script takes the ground truth and transcript, and cleans and normalizes them. There are 3 levels of normalization:

1. **punctuation:** This is done by the built-in Whisper normalizer. It removes special tokens, lowercases everything, removes double spaces, and removes punctuation.

2. **lexicon:** A really small lexicon is applied that removes meaningless filler words like 'uhm', 'uhh', 'err' and changes some words to their standardized form, like "m'n" to 'mijn'.

3. **stutter:** These are words that do have a significant meaning but are repeated multiple times. These do not add any additional information and are almost always redundant and not grammatically correct. There are cases where it is correct, like 'je je' in some context, but no meaning is lost by deleting these cases, so these can be removed as well when calculating WER and SimDist.

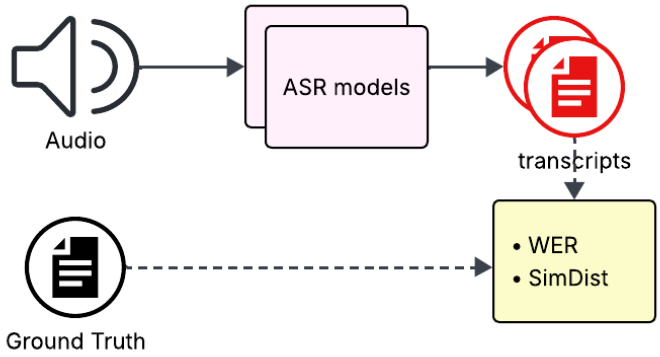

Figure 1: Running the models and calculating metrics

Putting everything together will create a table consisting of 6 scores for each model on each audio file. These are the SimDist and WER for each level of normalization. The performance and speed are also stored inside this table. Analysing this table will answer the first research question.

3.2.5 Analysis

The transcript in each row of the generated table is uniquely determined by an audio file and a model. Given an audio file $a \in A$, and an ASR model $m \in M_{ASR}$, the first part of the pipeline p_1 maps them 1-to-1 to a transcript $f \in F_{transcript}$ such that:

$$p_1(a, m) \to f$$

Each transcript also has a ground truth transcript associated with it $f_{gt} \in F_{gt}$, in a many-to-one relation:

$$h(f) \to f_{gt}$$

Note that an audio file a has just one corresponding ground truth transcript. There are many factors to consider for each audio file that could play a role. Due to the small sample size, only three

were incorporated in the analysis. These were audio length, audio density, and category. Different factors in audio that might influence transcription accuracy, such as noise and types of speech, are hard to ground in truth and require human feedback to identify. These are not included.

$$A : \{A_l, A_d, A_c\}$$

The second part of the pipeline s_1 calculates the transcript metrics WER X_{wer} and SimDist $X_{SimDist}$, by taking in the output transcript from the previous part p_1:

$$s_1(f, h(f)) \rightarrow \{f_{WER}, f_{SimDist}\}$$

Removing the intermediate variables and functions gives the following relation between variables:

$$(m, a_t, a_d, a_c) \rightarrow \{f_{WER}, f_{SimDist}\}$$

It is assumed that there is no underlying relation between the audio file variables and that these are independent. This section will analyze the relation between these 4 variables and the performance metrics.
This will be done by analyzing general performance of each model, the performance compared to benchmarks, the relation between audio files and accuracy scores accounting for model variance, the relation between accuracy scores, and the resource breakdown of each model.

3.3 Experiment 2: Correlation between transcript accuracy and the summary's quality

3.3.1 Gathering data

No further data will be collected during this step except for a prompt used in the LuisterLinie application. This prompt will stay consistent throughout testing. It will be adjusted by removing any specific identifiers of the LuisterLinie or The Salvation Army. It also contains different instructions for the model depending on whether it detects an interview or a (scripted) conversation between a psychiatrist or doctor and a client.

3.3.2 Model selection

For generating summaries, GenAI models are used. The model that is used in the LuisterLinie is ChatGPT4. This model will be the baseline. Different models are selected to cross-validate the results. The prompt is specifically made for the current model, which might lead to skewed results. The GenAI model normally has to fit a set of constraints, but since the focus lies on the ASR models and the correlation between them, these are ignored in favor of selecting stronger models with the chance of improved summary quality.

3.3.3 Creating the pipeline

Again, two scripts are responsible for running the models and calculating the scores. First, the prompt is passed through a model as a system message. Then, one by one, the transcript of each

audio file from each ASR model, including the ground truth, is passed through. This generates a large set of structured summaries. This is done for each GenAI model.

For the second script, two different types of scores will be calculated. All the generated summaries are scored using summarization metrics. First, the accuracy scores, ROUGE-2 and BERTScore, are calculated. These require a reference summary, which will be the summary made by feeding a GenAI model the ground truth text file. Now Each model, each model's transcript summary will be compared to this reference summary to calculate these scores. These scores can now be used to analyse the correlation between summary accuracy and transcript accuracy.

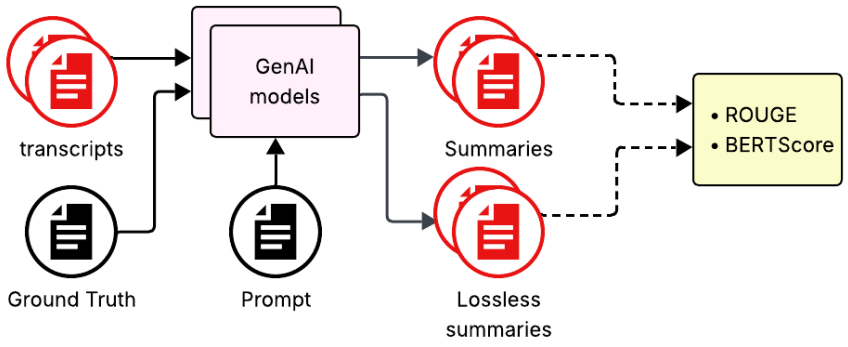

Figure 2: Running the models and calculating accuracy metrics

The second part calculates the quality of the summary in general by using the ground truth without summarizing it, with metrics like SummaC, density, coverage, and compression rate. Generated summaries are already normalized due to the nature of LLMs, but the ground truth needs to go through the same normalization steps as in the second script of the first pipeline.

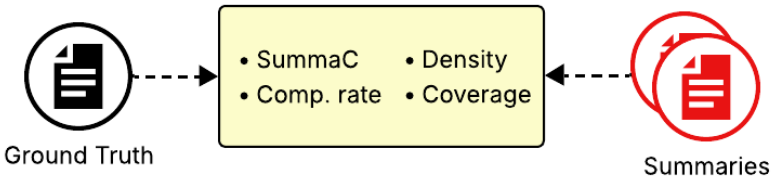

Figure 3: Calculating the non-accuracy metrics

This new table will be analyzed to answer the second research question.

3.3.4 Analysis

For this experiment, GenAI models M_{GenAI} are used to generate the summaries $F_{summary}$. The second pipeline p_2 takes the variables from experiment 1, as well as a GenAI model $n \in M_{GenAI}$

and maps them to a summary $f_s \in F_{summary}$

$$p_2(f, n) \rightarrow f_s$$

Four of the scores are calculated using the ground truth that belongs to the transcript the summary was generated on. Every summary has exactly 1 ground truth transcript associated with it, just like the transcripts, so h works for any generated piece of text.

$$s_{2.1}(f_s,\ h(f_s)) \rightarrow \{f_{sSummaC}, f_{sCoverage}, f_{sDensity}, f_{sCompression}\}$$

The accuracy scores are calculated separately using a summarization of the ground truth.

$$s_{2.2}(f_s,\ p_2(h(f_s),\ n)) \rightarrow \{f_{sROUGE}, f_{sBERTScore}\}$$

ROUGE-2 is used during this experiment but will be referred to as ROUGE. The correlation between X_{ROUGE}, $X_{BERTScore}$, and X_{SummaC} and $X_{SimDist}$, and X_{WER} is the primary focus of this experiment. Since $X_{SimDist}$ and X_{WER} are strongly correlated, they are analyzed separately, instead of being added as independent variables into a single formula. During the experiment, multiple methods will be applied to ensure the findings are significant. The variance caused by the different audio files and GenAI models will be accounted for. Using an Ordinary Least Squares (OLS) model, the GenAI model can be added as a dummy variable to account for model performance. This is done separately for SimDist and WER for ROUGE, BERTScore, and SimDist.

4 Results

4.1 Data collection

The main and only method used for gathering data was downloading audio from YouTube and transcribing it. Searching on large dataset platforms for Dutch medical speech did not yield any results. The total length of gathered data came out to around 6 hours and 53 minutes. This was spread over 3 categories. *DokterPatient* has small audio files of scripted conversations between actors. They roleplay medical consultations between a dokter and a client. These consist of very well-spoken, formal speech and high-quality audio. The category *Psychologischegespreksvoering* contains similar conversations but about mental health instead. These have lower-quality audio and are less formal. The final category *interviews* was selected due to the lack of good audio. These are not conversations between clients and instead interviewers. The topics remain similar, but the structure does not resemble that of the use case being investigated.

Category	**Speech Type**	**Conv. Type**	**Entries**	**Tot. Duration (h)**
Dokter Patient	Read	Consultation	8	0:24
Psych. gespr.	Spontaneous	Consultation	7	1:17
interviews	Spontaneous	Interview	10	5:17

Table 1: Specifications of the Audio Dataset

4.2 Model Selection

By talking to the team, 3 constraints were identified that an ASR model had to abide by.

- **Open-source** It has to be hostable on the cloud computing platform Azure. Due to strict privacy regulations, third-party models were not an option.
- **Multilingual** It has to be able to transcribe Dutch spoken language
- **Size** The model has to be relatively small, preferably having no more than 2 billion parameters.

Additionally, multiple versions of the same model are excluded. Only the best instance of a specific model is tested. The total pool of available models consists of 96 different models. By applying the constraints one by one, this total pool of candidate models shrinks till there are 8 models left.

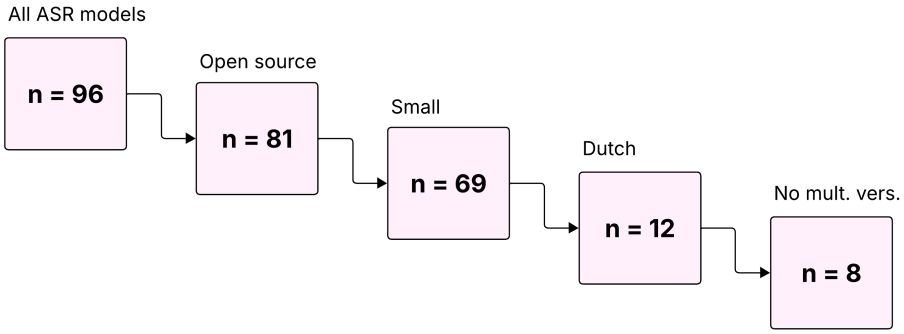

Figure 4: Gathering ASR models

Three of these: GLM-ASR-Nano-2512, owsm_ctc_v4_1B, and mms-1b-all were ultimately not tested during the experiment. The first one was not yet available during the first runs, and the other two, which were selected based on average WER, were dropped due to time constraints. The final set consists of the following models. The average WER was selected directly from the paper instead of the website.

Model Name	**Parameters**	**Languages**	**WER**
Cohere Transcribe [Lab26]	2B	14	5.42
Whisper v3 [Ope24]	1.55B	99	7.44
Qwen3 ASR [Qwe26]	1.7B	30	5.76
Canary v2 [NVI25a]	1B	25	7.15
Parakeet [NVI25b]	0.6B	25	6.32

Table 2: set of ASR Models

Most of these models process the input data in parallel, using the self-attention mechanisms of the transformer architecture. These provide every token with information about its neighbors, providing important semantic context. This is generally good for accuracy, but decreases the RTFx. That's why Parakeet was trained using a sequential architecture for its decoder, greatly improving the RTFx and reducing overall size. It still performs better on standardized benchmarks than both Canary and Whisper.

Since there were no constraints regarding the GenAI models, the only limit was pricing. This meant the pool of GenAI models consisted of ones that were available without requiring further payment. This meant GPT-4 did not get included. The final set consists of an arbitrary number of the best models from each provider. Sometimes the same model was selected but with a different level of thinking.

Model Name	Thinking levels	Provider
Gemini 3.1 pro	high, low	Google
Gemini 3.5 pro	high, low	Google
GPT 5	-	OpenAI
GPT 5.4	high, medium	OpenAI
GPT 5.5	high, medium	OpenAI
Haiku 4.5	-	Anthropic
Opus 4.8	-	Anthropic
Sonnet 5	-	Anthropic

Table 3: Set of GenAI models

4.3 Experiment 1

To confirm everything went smoothly, the expected number of entries for both pipeline outputs is calculated. The expected size of the DataFrame D_1 is defined by the product of the number of ASR models M_{ASR} and the number of audio files F in the dataset in the following formula:

$$|D_1| = |M_{ASR}| \cdot |F| = 6 \cdot 25 = 150$$

This matched the size of the dataframe, confirming everything ran. However, during the run, every model's output transcript length was compared to that of the ground truth length. Two files were flagged for having a deviation of over 25%. This happened on the *Verzorings-staat.mp3* file for the *Canary* and *Qwen* models. Rerunning and changing starting conditions did not help and returned the same errors.

General performance To get an overview of the model's overall performance across all data, the metrics for each file are graphed in a boxplot per model. It is a good way to visualize the mean, standard error, and outliers that might skew results.

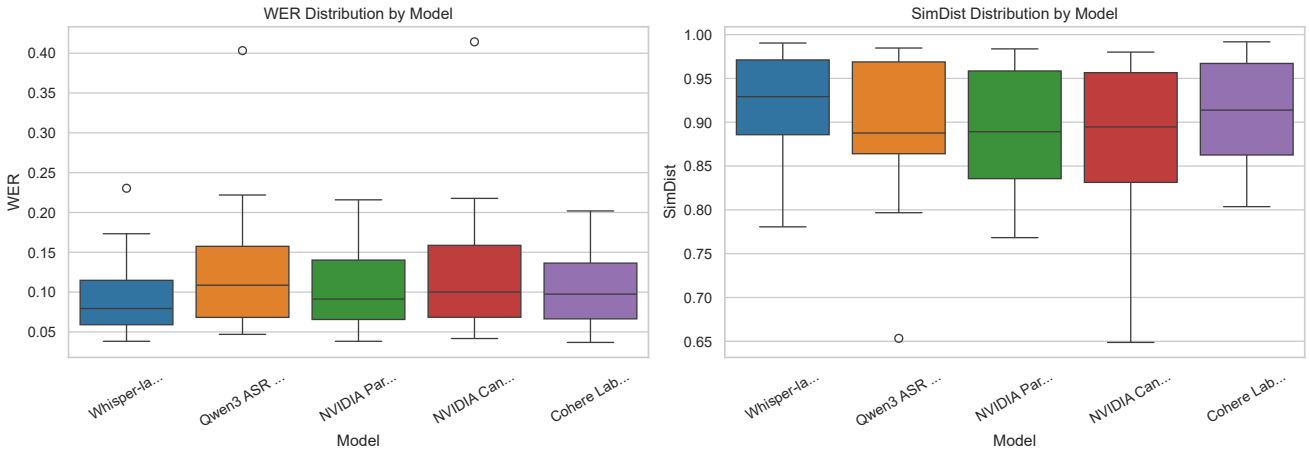

Figure 5: General performance per ASR model

The first thing to notice is that the two data points that were flagged earlier do show up as outliers for Qwen and Canary. For now, these points will be excluded from the dataset during later analysis. Whisper significantly outperformed the other models, having a really low interquartile range of 5.6 and the lowest mean WER of 9.5. Parakeet and Cohere perform relatively similarly, having the same standard deviation and WERs of 10.4 and 10.1, respectively. The SimDist tells the same story.

Benchmark performance The models were not set up with speed in mind during this experiment, and this lack of optimization drastically reduced the performance, making the comparison to the competitive RTFx scores useless.

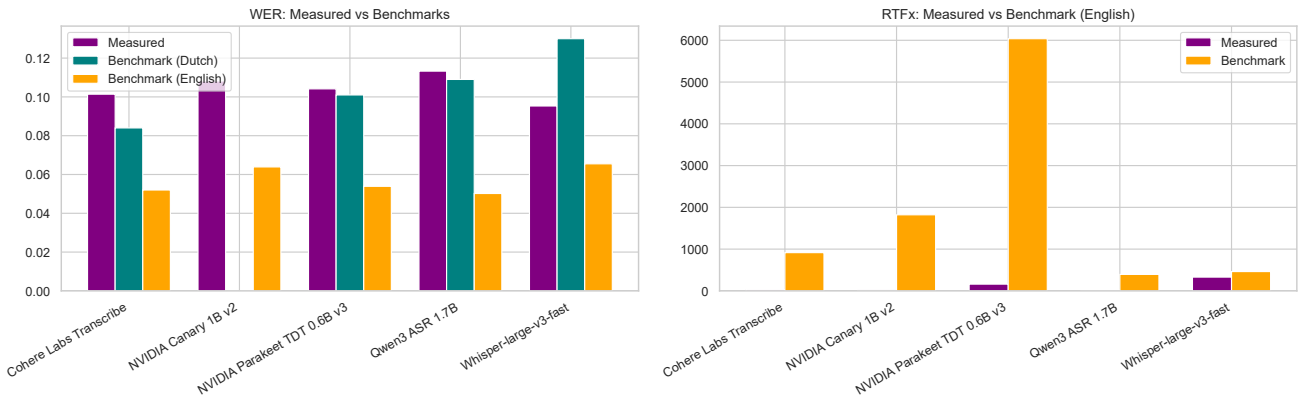

Figure 6: WER and RTFx across benchmarks

The models perform worse on Dutch audio compared to English. In turn, the models performed worse on this benchmark compared to both of them, with one exception, which is Whisper's performance.
To visualize this, the percentage point change in the total share of WER between benchmarks is plotted for each model. The idea is that if all the models' accuracy is constant across different data, their change in share would be 0.

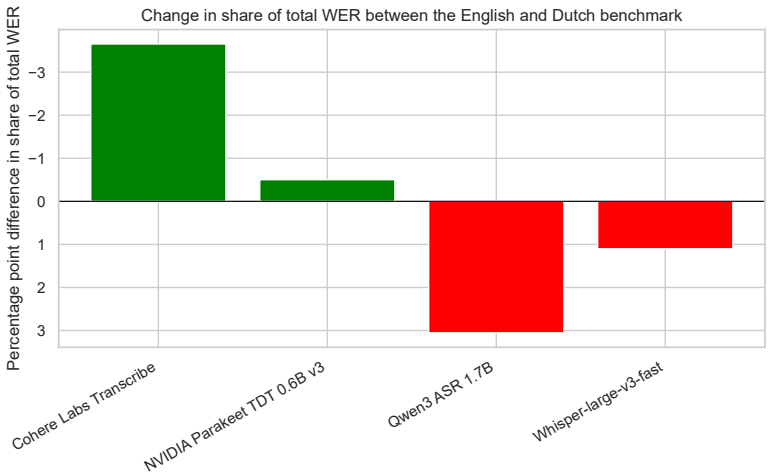

Figure 7: Change in performance between Dutch and English

Cohere's relative accuracy increases significantly compared to the English benchmark. It now has the lowest WER by almost 2 points. On the other hand, Qwen performs worse on Dutch, as shown in the second graph.

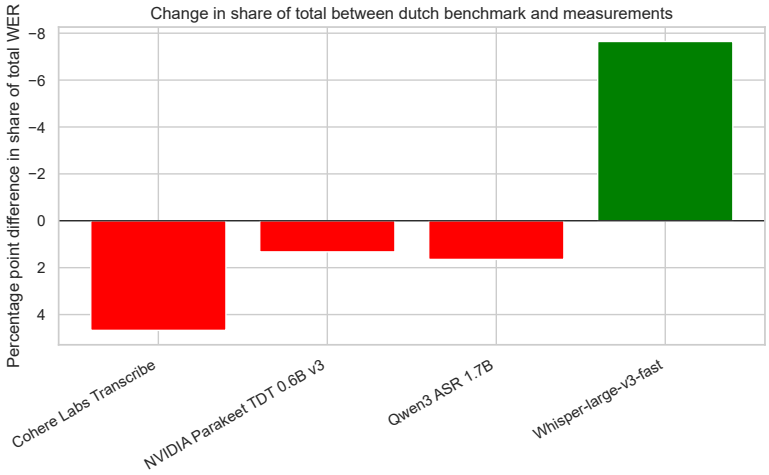

Figure 8: Change in performance between Dutch and domain-specific audio

What is different is the accuracy of Whisper-large-v3-fast and Cohere. In the current setup, Whisper achieves an incredible WER. Where it previously ranked last on both benchmarks, it now performs the best.

Audio-file-specific Performance To evaluate the models' domain-specific performance, the accuracy is analyzed across categories, audio density, and audio file duration. For the categories, a heatmap is created for both the average WER and SimDist of each model. The darker the color, the worse the score.

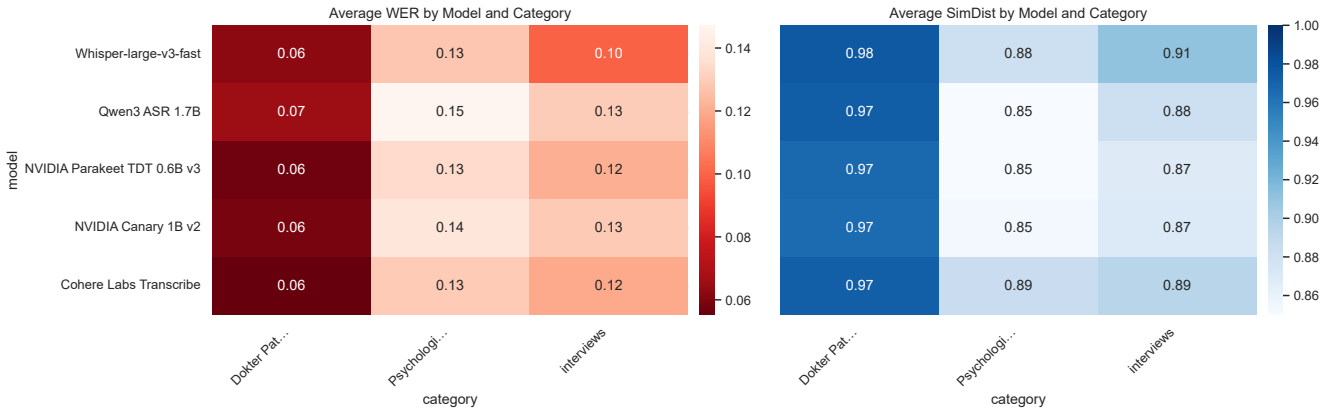

Figure 9: Metrics by Category

With respect to overall model performance, nothing changed compared to the general performance box plot. Cohere and Whisper remain the best models across all categories. The variance is mainly caused by the categories and not the models. The scripted short conversations report the best accuracy scores. These contain no stutters or muffling. These are present in both the 'interviews' and 'psychologische gespreksvoering' categories.

For comparing accuracy to the audio length, a regular graph is used since the duration of audio files lies on a continuous interval. Each model has its own set of connected dots.

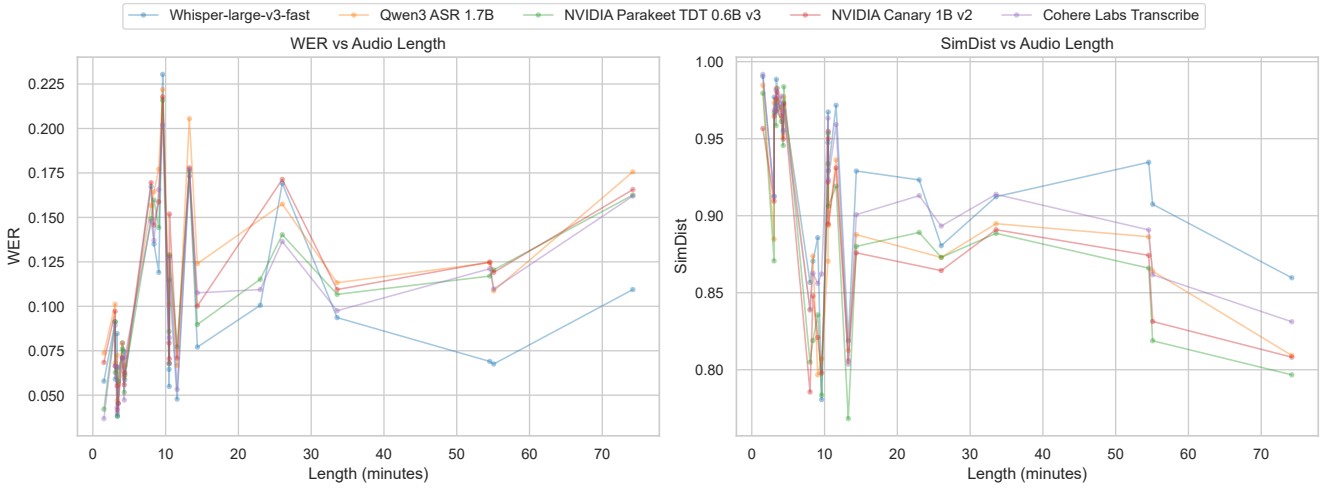

Figure 10: Metrics vs audio file length

There is no correlation between the length of the audio file and the accuracy scores, with an $r^2 = 0.09$ for $X_{WER} \sim A_d$ and an $r^2 = 0.20$ for $X_{SimDist} \sim A_d$. It shows high variance per audio file and lower variance per model. Same as with the categories.
The audio density actually captures how fast speakers talk. It is calculated by dividing the number of words in the ground truth by the audio length. It uses the same type of graph.

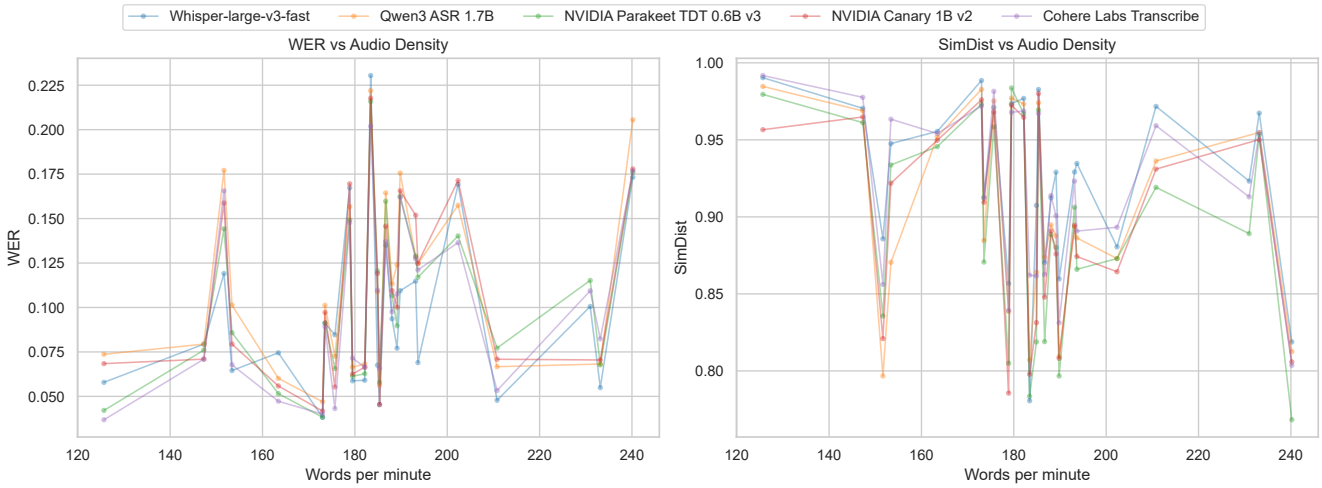

Figure 11: Metrics vs audio density

It shows the same patterns. Low variance within models and no correlation.

Hardware Performance The performance of each model in terms of memory, GPU, CPU usage, and speed is not significant. The models are not optimized at all.

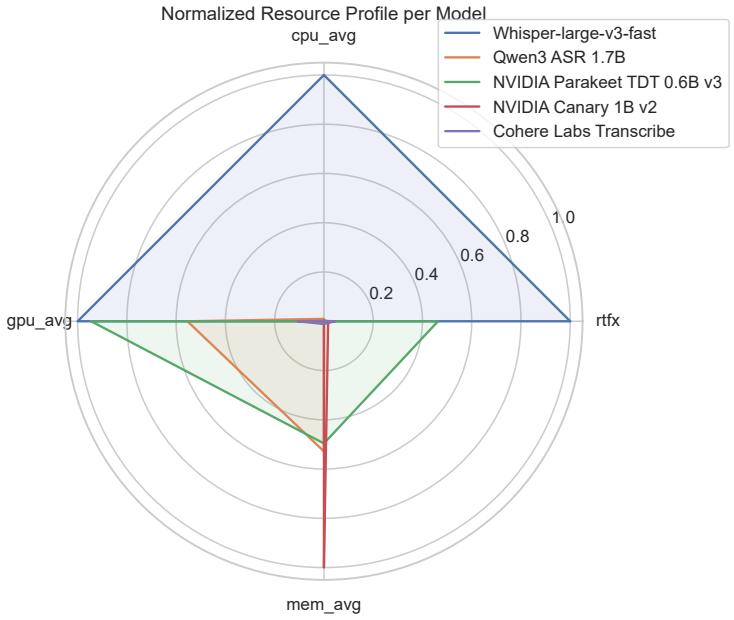

Figure 12: Metrics vs audio density

Instead, this graph shows where optimizations are possible to boost RTFx. For instance, Canary does not use the GPU and CPU at all and instead does everything in RAM, which is not optimal. Most notable is that Cohere uses the least amount of resources by a large margin.

ASR vs SimDist Comparing the accuracy scores $X_{WER} \sim X_{SimDist}$ using a scatterplot, coloring each model separately.

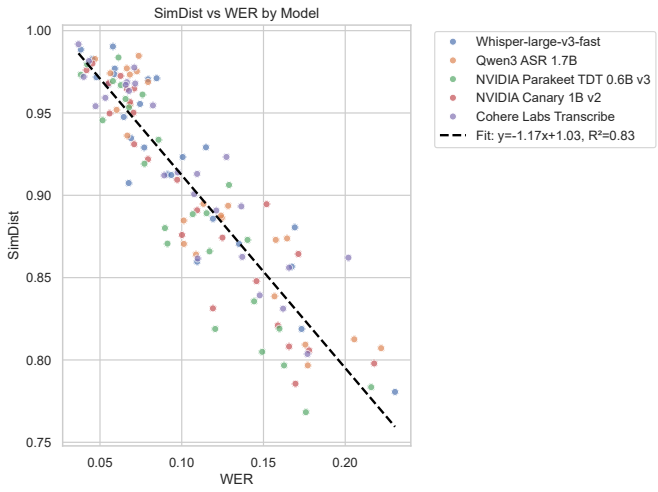

Figure 13

In this dataset, the correlation is really apparent with $r^2 = 0.83$.
Both the files used for calculating the accuracy scores were transcribed using different levels of normalization at first. The following graph shows the impact

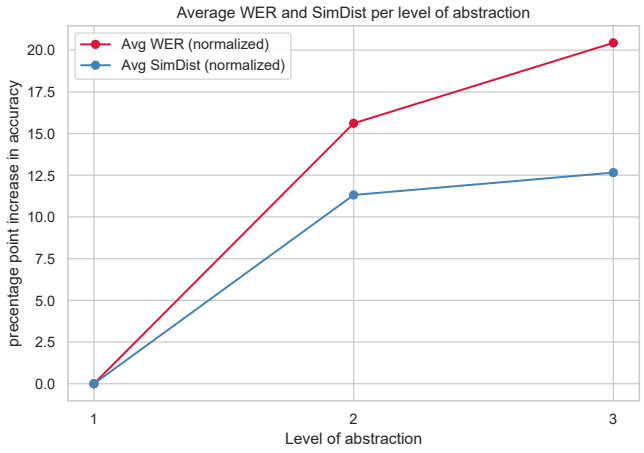

Figure 14: Accuracy scores vs abstraction

4.4 Experiment 2

The expected number of entries in the second DataFrame D_2 for the summaries is defined by the product of the number of ASR models, plus one for the ground truth files themselves, the number of GenAI models M_{GenAI}, and the number of files in the dataset in the following formula:

$$|D_2| = (|M_{ASR}| + 1) \cdot |M_{GenAI}| \cdot |F| = (5 + 1) \cdot 12 \cdot 25 = 1800$$

The second DataFrame was 12 entries short. The missing entries were all from transcripts generated by the *Canary*, and 11 of those were from one audio file. Summaries were generated over two different runs, with the second one having different models from the first. The set of

models used in the first run was also used in the second run except for *GPT-5*. Because of this, it was dropped entirely, bringing the model count down to 11 and the total number of entries to 1658.

While exploring the data further, some GenAI models seem to have produced incorrect data.

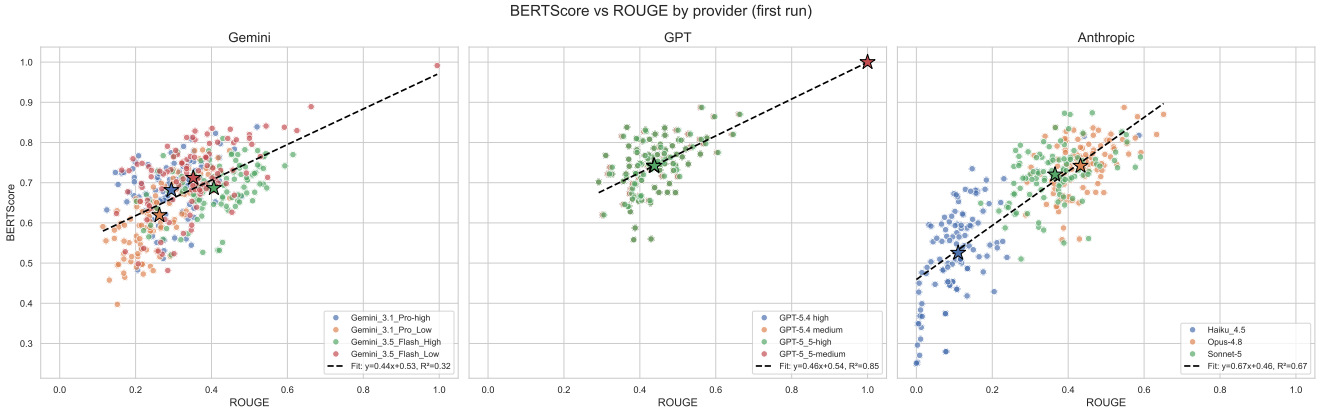

Figure 15: Exploration of the ROUGE and BERTScore metrics per provider

This graph makes them visible. The GPT models show no variance. This means they created the same summary for each transcript. Most likely because they remembered each other's output. *GPT 5.5* medium did have different output but instead remembered the ground truth summary and produced the same summary 109 times. Moving forward, only the best GPT model on the highest thinking setting is used for analysis. *Gemini 3.5 Flash Low* has 1 file with a perfect score. Instead of removing the entire model, that one entry is deleted. The final analysis is done using 8 models.

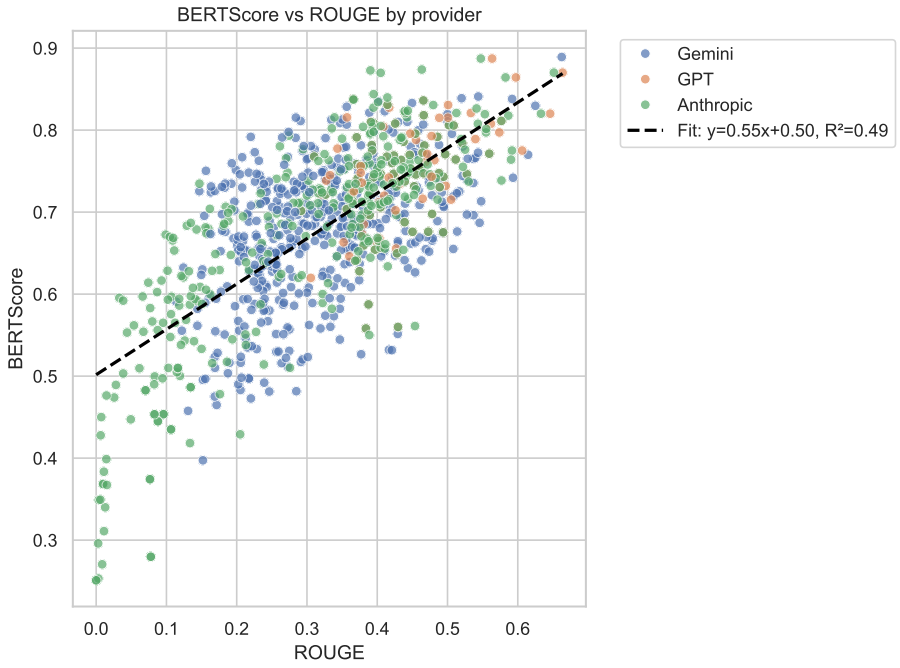

Figure 16: ROUGE and BERTScore metrics per provider

They all show some to no correlation, with a weak correlation of $r^2 = 0.49$ when combined into one.

Exploring summarization metrics

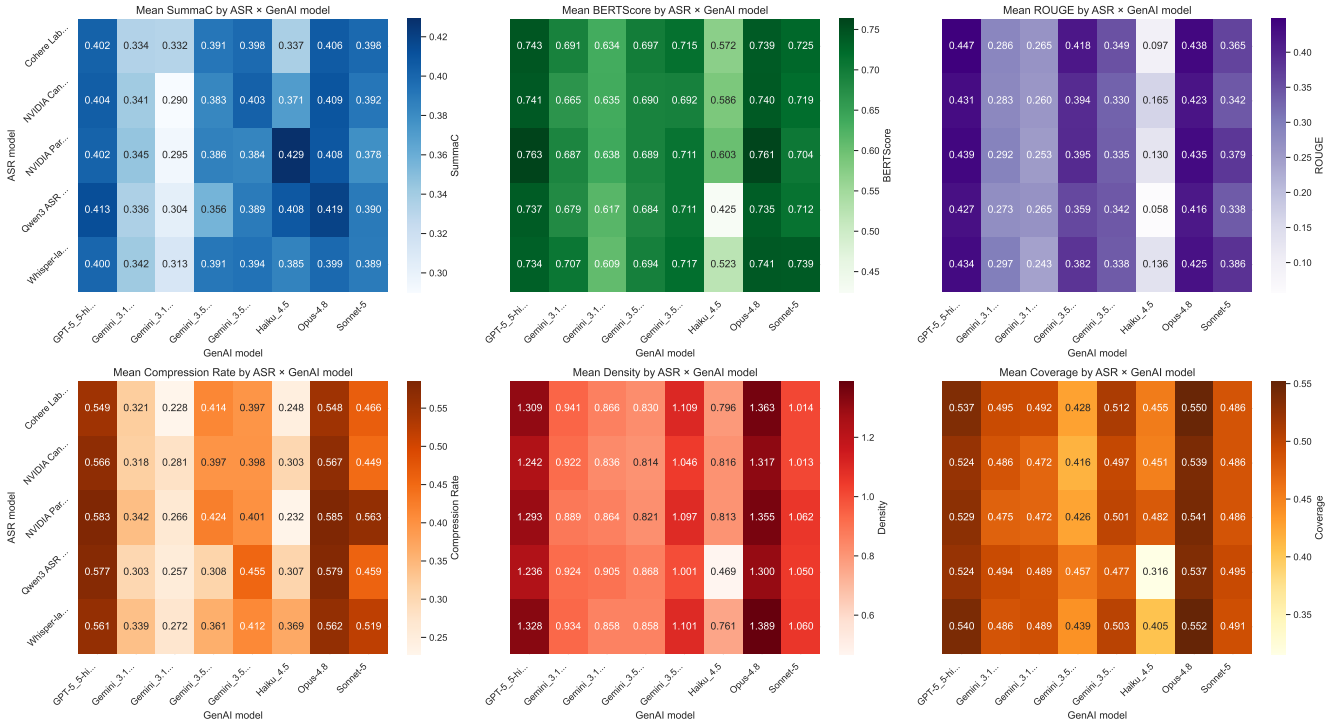

Figure 17: General performance per ASR model x GenAI model

It becomes clear from the graph that the variance is mainly caused by the different GenAI models. Generally, newer models perform better according to every metric. There is some variance between ASR models (like with Haiku), but this variance is not consistent across GenAI models. Experiment one established that most of the variance in WER was caused by the difference in audio files and not the ASR models themselves, so this should be taken into account. Ignoring the ASR models and instead ranking the GenAI models based on their average score for each of the three core metrics produces this table.

GenAI Model	SummaC	BERTScore	ROUGE	Avg. Rank
Opus 4.8	**1**	**1**	2	1.33
GPT 5.5 High	2	**1**	**1**	1.33
Sonnet 5	4	3	4	3.67
Gemini 3.5 Flash Low	3	4	5	4.00
Gemini 3.5 Flash High	5	5	3	4.33
Gemini 3.1 Pro high	7	6	6	6.33
Gemini 3.1 Pro Low	8	7	7	7.33
Haiku 4.5	6	8	8	7.33

Table 4: GenAI model ranking by core metrics

There is also a high chance, from the looks of the heatmaps, that every metric is just correlated with the compression rate. This would mean that a GenAI model isn't inherently better or more robust, but instead generating larger outputs results in better scores.

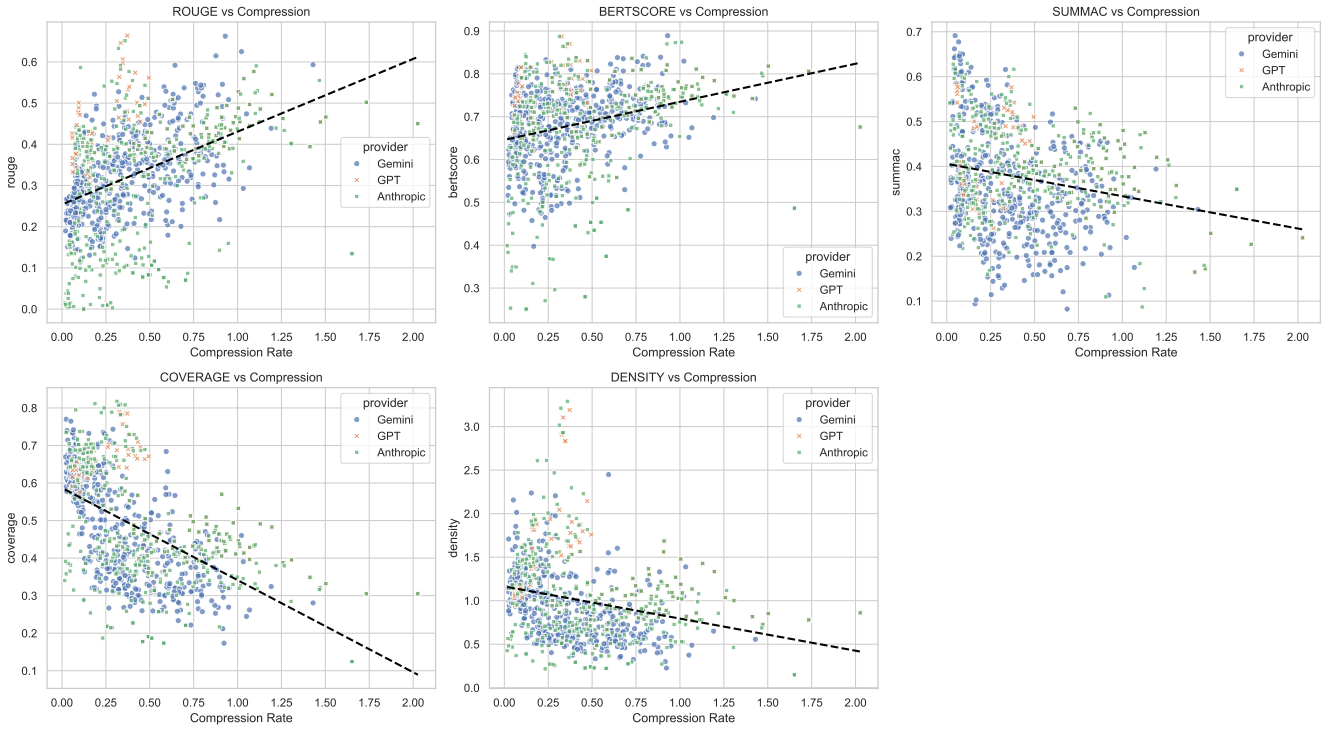

Figure 18: Summary metrics correlation with compression rate

Since the data contains a lot of outliers across the metrics, Spearman's r score is used. It compares the change in rank between the 2 variables instead of their change in numerical values to account for the outliers.

Metric	**Spearman's** r	p**-value**	n
Coverage	-0.615	0.000	1175
Density	-0.364	0.000	1175
ROUGE	0.349	0.000	1175
BERTScore	0.217	0.000	1175
SummaC	-0.177	0.000	1175

Table 5: Spearman correlation between summary metrics and compression rate

SummaC shows the lowest score (-0.177) while Density and Coverage have the highest (-0.364, -0.615 respectively). These metrics do not depend solely on compression rate, and the GenAI models could still have a relevant impact.

Baseline Correlations transcription accuracy The accuracy scores for the summary express a drift in summary accuracy away from the ground truth summary. The actual value cannot

be well interpreted, since the ground truth summary is not necessarily the optimal summary. Only that the less similar the ground truth and transcript are, the less accurate you expect the respective summaries to be compared to each other. Plotting the metrics against each other, without considering previous observations that variance might also be caused by GenAI models or audio files, produces the following graphs.

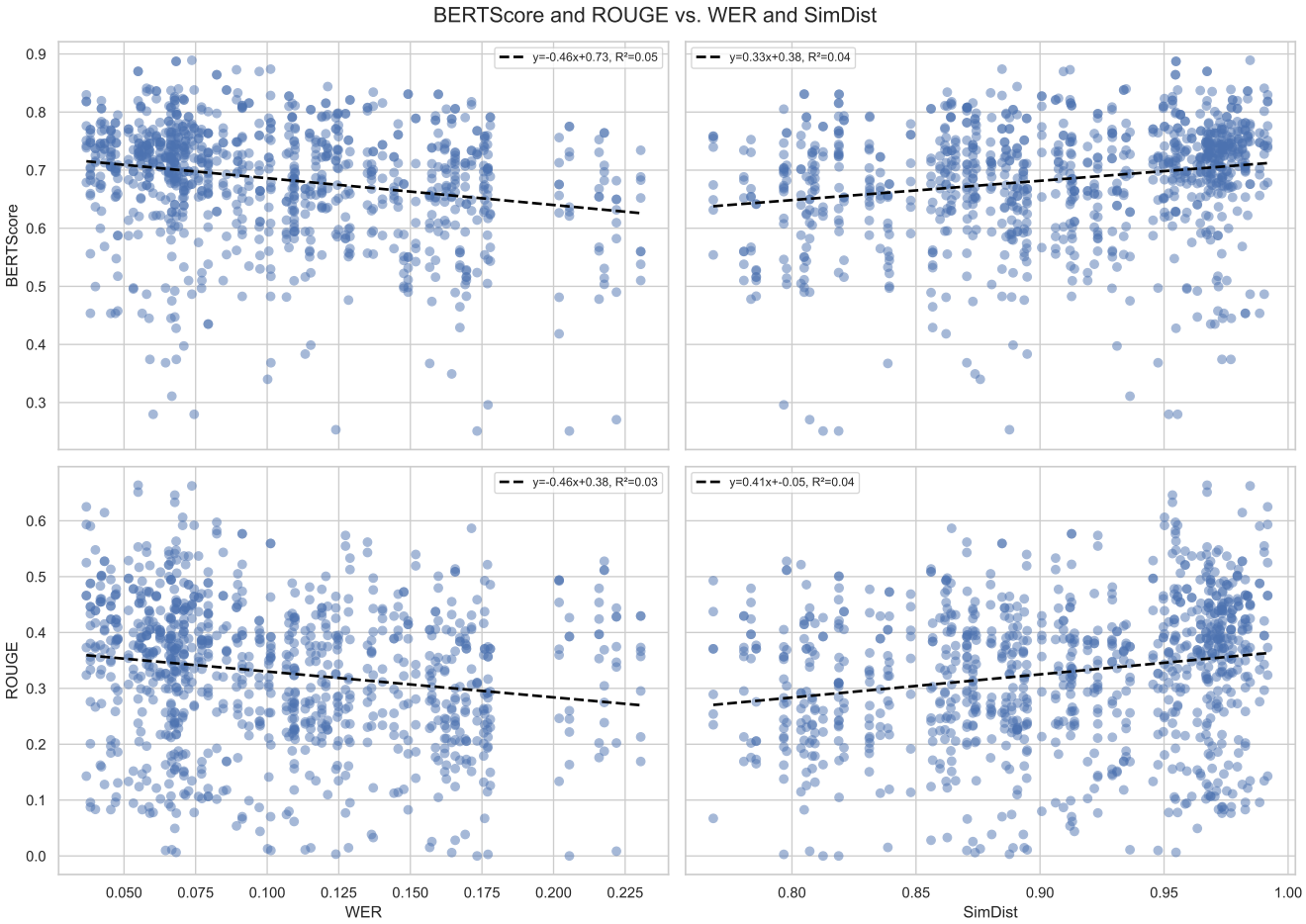

Figure 19: Transcription vs summarization accuracy metrics

It shows no correlation between any of the summary metrics and any of the transcript metrics. All the slopes do point in the expected directions.
To analyse whether an increase in WER decreases faithfulness, the model should look at the change in SummaC instead of the absolute score. For any summary $f_s \in F_{summary}$, $\Delta f_{sSummaC}$ is defined by subtracting the ground truth summary's SummaC score from the actual one. A positive value implies a decrease in faithfulness.

$$f_s : \Delta f_{sSummaC} = f_{sSummaC} - h(f_s)_{SummaC}$$

$X_{Simdist}$ will also be concentrated around 0. A positive value implies a decrease in semantic meaning.

$$f_s : \Delta f_{sSimDist} = 1 - f_{sSimDist}$$

Note that $X_{WER} = \Delta X_{WER}$ since $h(f_s)_{WER} = 0$ for all $f_s \in F_{summaries}$. Using these for the correlation produces the following scatter plot.

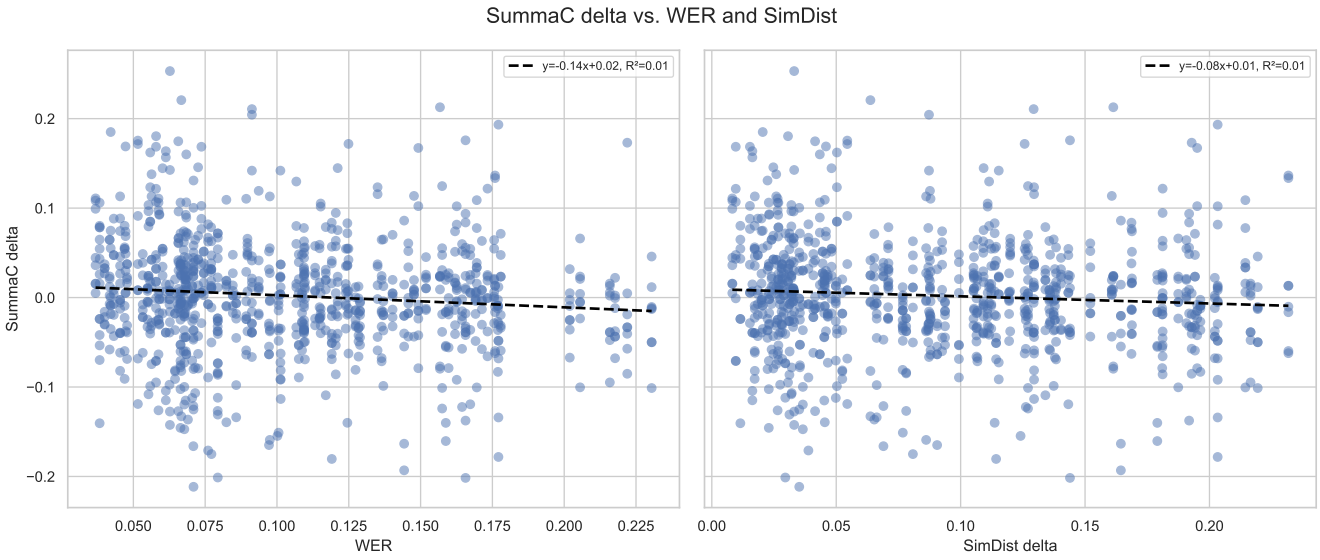

Figure 20: General performance per ASR model

This flips the SimDist slope. It shows no correlation, but the slopes do point in the right direction

GenAI model variance Even though delta SummaC compares 2 scores within the same GenAI model, the model could still influence the metric since it might be more or less susceptible to changes in WER or SimDist. OLS is performed with the GenAI models as a dummy variable. First, GPT is taken as the baseline model, and each model's intercept is predicted at WER 0 relative to the baseline model.

	coef	std err	t	P¿—t—	[0.025	0.975]
Intercept	-0.0193	0.013	-1.485	0.138	-0.045	0.006
C(genai_model)[T.Gemini_3.1_Pro-high]	-0.0213	0.018	-1.159	0.247	-0.057	0.015
C(genai_model)[T.Gemini_3.1_Pro_Low]	0.0459	0.018	2.496	0.013	0.010	0.082
C(genai_model)[T.Gemini_3.5_Flash_High]	0.0295	0.018	1.605	0.109	-0.007	0.066
C(genai_model)[T.Gemini_3.5_Flash_Low]	0.0153	0.019	0.829	0.407	-0.021	0.052
C(genai_model)[T.Haiku_4.5]	-0.0411	0.018	-2.231	0.026	-0.077	-0.005
C(genai_model)[T.Opus-4.8]	0.0068	0.018	0.371	0.711	-0.029	0.043
C(genai_model)[T.Sonnet-5]	-0.0096	0.018	-0.522	0.602	-0.046	0.026
wer	0.1713	0.113	1.513	0.131	-0.051	0.393
wer:C(genai_model)[T.Gemini_3.1_Pro-high]	0.0324	0.160	0.202	0.840	-0.282	0.347
wer:C(genai_model)[T.Gemini_3.1_Pro_Low]	-0.1329	0.160	-0.830	0.407	-0.447	0.181
wer:C(genai_model)[T.Gemini_3.5_Flash_High]	-0.1050	0.160	-0.656	0.512	-0.419	0.209
wer:C(genai_model)[T.Gemini_3.5_Flash_Low]	-0.1048	0.161	-0.652	0.515	-0.420	0.211
wer:C(genai_model)[T.Haiku_4.5]	0.0919	0.160	0.573	0.567	-0.223	0.407
wer:C(genai_model)[T.Opus-4.8]	-0.0624	0.160	-0.389	0.697	-0.377	0.252
wer:C(genai_model)[T.Sonnet-5]	-0.0057	0.160	-0.036	0.972	-0.320	0.308

Table 6: $\Delta X_{SummaC} \sim X_{WER} \cdot C(M_{GenAI})$

The same analysis was done, but looking at the effects of SimDist delta.

	coef	std err	t	P¿—t—	[0.025	0.975]
Intercept	-0.0091	0.010	-0.926	0.355	-0.028	0.010
C(genai_model)[T.Gemini_3.1_Pro-high]	-0.0280	0.014	-2.012	0.045	-0.055	-0.001
C(genai_model)[T.Gemini_3.1_Pro_Low]	0.0444	0.014	3.194	0.001	0.017	0.072
C(genai_model)[T.Gemini_3.5_Flash_High]	0.0240	0.014	1.728	0.084	-0.003	0.051
C(genai_model)[T.Gemini_3.5_Flash_Low]	0.0103	0.014	0.734	0.463	-0.017	0.038
C(genai_model)[T.Haiku_4.5]	-0.0388	0.014	-2.782	0.006	-0.066	-0.011
C(genai_model)[T.Opus-4.8]	0.0025	0.014	0.180	0.857	-0.025	0.030
C(genai_model)[T.Sonnet-5]	-0.0175	0.014	-1.258	0.209	-0.045	0.010
simdist_delta	0.0827	0.088	0.936	0.350	-0.091	0.256
simdist_delta:C(genai_model)[T.Gemini_3.1_Pro-high]	0.1081	0.125	0.865	0.387	-0.137	0.353
simdist_delta:C(genai_model)[T.Gemini_3.1_Pro_Low]	-0.1332	0.125	-1.066	0.287	-0.378	0.112
simdist_delta:C(genai_model)[T.Gemini_3.5_Flash_High]	-0.0588	0.125	-0.470	0.638	-0.304	0.186
simdist_delta:C(genai_model)[T.Gemini_3.5_Flash_Low]	-0.0630	0.125	-0.502	0.616	-0.309	0.183
simdist_delta:C(genai_model)[T.Haiku_4.5]	0.0780	0.125	0.623	0.534	-0.168	0.324
simdist_delta:C(genai_model)[T.Opus-4.8]	-0.0237	0.125	-0.189	0.850	-0.269	0.222
simdist_delta:C(genai_model)[T.Sonnet-5]	0.0785	0.125	0.628	0.530	-0.167	0.324

Table 7: $\Delta X_{Summac} \sim \Delta X_{SimDist} \cdot C(M_{GenAI})$

This method can also be used for the summarization accuracy metrics.

	coef	std err	t	P¿—t—	[0.025	0.975]
Intercept	0.4643	0.017	27.812	0.000	0.432	0.497
C(genai_model)[T.Gemini_3.1_Pro-high]	-0.1297	0.024	-5.495	0.000	-0.176	-0.083
C(genai_model)[T.Gemini_3.1_Pro_Low]	-0.1164	0.024	-4.930	0.000	-0.163	-0.070
C(genai_model)[T.Gemini_3.5_Flash_High]	-0.0444	0.024	-1.881	0.060	-0.091	0.002
C(genai_model)[T.Gemini_3.5_Flash_Low]	-0.0423	0.024	-1.780	0.075	-0.089	0.004
C(genai_model)[T.Haiku_4.5]	-0.3522	0.024	-14.880	0.000	-0.399	-0.306
C(genai_model)[T.Opus-4.8]	-0.0009	0.024	-0.038	0.970	-0.047	0.045
C(genai_model)[T.Sonnet-5]	-0.0202	0.024	-0.855	0.393	-0.067	0.026
wer	-0.2596	0.145	-1.787	0.074	-0.545	0.025
wer:C(genai_model)[T.Gemini_3.1_Pro-high]	-0.1968	0.205	-0.958	0.338	-0.600	0.206
wer:C(genai_model)[T.Gemini_3.1_Pro_Low]	-0.6066	0.205	-2.952	0.003	-1.010	-0.203
wer:C(genai_model)[T.Gemini_3.5_Flash_High]	-0.0166	0.205	-0.081	0.936	-0.420	0.387
wer:C(genai_model)[T.Gemini_3.5_Flash_Low]	-0.5264	0.206	-2.552	0.011	-0.931	-0.122
wer:C(genai_model)[T.Haiku_4.5]	0.3226	0.206	1.568	0.117	-0.081	0.727
wer:C(genai_model)[T.Opus-4.8]	-0.0683	0.205	-0.332	0.740	-0.472	0.335
wer:C(genai_model)[T.Sonnet-5]	-0.5044	0.205	-2.455	0.014	-0.908	-0.101

Table 8: $X_{ROUGE} \sim X_{WER} \cdot C(M_{GenAI})$

	coef	std err	t	P¿—t—	[0.025	0.975]
Intercept	0.7722	0.016	47.828	0.000	0.741	0.804
C(genai_model)[T.Gemini_3.1_Pro-high]	-0.0255	0.023	-1.116	0.265	-0.070	0.019
C(genai_model)[T.Gemini_3.1_Pro_Low]	-0.0723	0.023	-3.165	0.002	-0.117	-0.027
C(genai_model)[T.Gemini_3.5_Flash_High]	-0.0402	0.023	-1.762	0.078	-0.085	0.005
C(genai_model)[T.Gemini_3.5_Flash_Low]	-0.0041	0.023	-0.179	0.858	-0.049	0.041
C(genai_model)[T.Haiku_4.5]	-0.1849	0.023	-8.078	0.000	-0.230	-0.140
C(genai_model)[T.Opus-4.8]	0.0036	0.023	0.156	0.876	-0.041	0.048
C(genai_model)[T.Sonnet-5]	0.0026	0.023	0.112	0.911	-0.042	0.047
wer	-0.2598	0.141	-1.849	0.065	-0.536	0.016
wer:C(genai_model)[T.Gemini_3.1_Pro-high]	-0.3092	0.199	-1.556	0.120	-0.699	0.081
wer:C(genai_model)[T.Gemini_3.1_Pro_Low]	-0.4293	0.199	-2.160	0.031	-0.819	-0.039
wer:C(genai_model)[T.Gemini_3.5_Flash_High]	-0.1132	0.199	-0.570	0.569	-0.503	0.277
wer:C(genai_model)[T.Gemini_3.5_Flash_Low]	-0.2984	0.200	-1.496	0.135	-0.690	0.093
wer:C(genai_model)[T.Haiku_4.5]	-0.1564	0.199	-0.786	0.432	-0.547	0.234
wer:C(genai_model)[T.Opus-4.8]	-0.0372	0.199	-0.187	0.851	-0.427	0.353
wer:C(genai_model)[T.Sonnet-5]	-0.2544	0.199	-1.280	0.201	-0.644	0.136

Table 9: $X_{BERTScore} \sim X_{WER} \cdot C(M_{GenAI})$

	coef	std err	t	P¿—t—	[0.025	0.975]
Intercept	0.2041	0.102	2.007	0.045	0.005	0.404
C(genai_model)[T.Gemini_3.1_Pro-high]	-0.3230	0.144	-2.246	0.025	-0.605	-0.041
C(genai_model)[T.Gemini_3.1_Pro_Low]	-0.5998	0.144	-4.170	0.000	-0.882	-0.317
C(genai_model)[T.Gemini_3.5_Flash_High]	-0.1233	0.144	-0.857	0.392	-0.406	0.159
C(genai_model)[T.Gemini_3.5_Flash_Low]	-0.4720	0.144	-3.271	0.001	-0.755	-0.189
C(genai_model)[T.Haiku_4.5]	-0.0286	0.144	-0.198	0.843	-0.311	0.254
C(genai_model)[T.Opus-4.8]	-0.0727	0.144	-0.505	0.614	-0.355	0.210
C(genai_model)[T.Sonnet-5]	-0.3615	0.144	-2.513	0.012	-0.644	-0.079
simdist	0.2569	0.112	2.297	0.022	0.037	0.476
simdist:C(genai_model)[T.Gemini_3.1_Pro-high]	0.1903	0.158	1.204	0.229	-0.120	0.501
simdist:C(genai_model)[T.Gemini_3.1_Pro_Low]	0.4629	0.158	2.927	0.004	0.153	0.773
simdist:C(genai_model)[T.Gemini_3.5_Flash_High]	0.0850	0.158	0.537	0.591	-0.225	0.395
simdist:C(genai_model)[T.Gemini_3.5_Flash_Low]	0.4130	0.159	2.602	0.009	0.101	0.724
simdist:C(genai_model)[T.Haiku_4.5]	-0.3196	0.158	-2.017	0.044	-0.631	-0.009
simdist:C(genai_model)[T.Opus-4.8]	0.0712	0.158	0.450	0.653	-0.239	0.382
simdist:C(genai_model)[T.Sonnet-5]	0.3181	0.158	2.011	0.045	0.008	0.628

Table 10: $\mathrm{X}_{ROUGE} \sim X_{SimDist} * C(M_{GenAI})$

	coef	std err	t	P¿—t—	[0.025	0.975]
Intercept	0.5898	0.100	5.886	0.000	0.393	0.786
C(genai_model)[T.Gemini_3.1_Pro-high]	-0.3089	0.142	-2.180	0.029	-0.587	-0.031
C(genai_model)[T.Gemini_3.1_Pro_Low]	-0.4190	0.142	-2.957	0.003	-0.697	-0.141
C(genai_model)[T.Gemini_3.5_Flash_High]	-0.1758	0.142	-1.241	0.215	-0.454	0.102
C(genai_model)[T.Gemini_3.5_Flash_Low]	-0.2631	0.142	-1.850	0.065	-0.542	0.016
C(genai_model)[T.Haiku_4.5]	-0.1780	0.142	-1.255	0.210	-0.456	0.100
C(genai_model)[T.Opus-4.8]	-0.0364	0.142	-0.257	0.797	-0.315	0.242
C(genai_model)[T.Sonnet-5]	-0.2517	0.142	-1.776	0.076	-0.530	0.026
simdist	0.1712	0.110	1.554	0.120	-0.045	0.387
simdist:C(genai_model)[T.Gemini_3.1_Pro-high]	0.2768	0.156	1.777	0.076	-0.029	0.583
simdist:C(genai_model)[T.Gemini_3.1_Pro_Low]	0.3328	0.156	2.136	0.033	0.027	0.639
simdist:C(genai_model)[T.Gemini_3.5_Flash_High]	0.1364	0.156	0.875	0.382	-0.169	0.442
simdist:C(genai_model)[T.Gemini_3.5_Flash_Low]	0.2510	0.156	1.605	0.109	-0.056	0.558
simdist:C(genai_model)[T.Haiku_4.5]	-0.0256	0.156	-0.164	0.870	-0.332	0.281
simdist:C(genai_model)[T.Opus-4.8]	0.0398	0.156	0.255	0.798	-0.266	0.346
simdist:C(genai_model)[T.Sonnet-5]	0.2509	0.156	1.611	0.108	-0.055	0.557

Table 11: $X_{BERTScore} \sim X_{SimDist} * C(M_{GenAI})$

The numbers are practically the same, which is to be expected considering the correlation between both WER and SimDist, and BERTScore and ROUGE.

Audio File Variance In the previous experiment, the audio file was decomposed into variables that described the nature of the audio. This time, those are irrelevant, since they are lost during the conversion to text. It is unknown what makes a conversation hard to summarize, but it is likely that things like jargon, structure of the conversation, etc, have some effect. The audio file's

transcripts are mostly the same across ASR models, and no relation between them and the audio file variables was found. The audio file is added as a fixed effect (FE) to the previous OLS model as a dummy constant $C(A)$. The question becomes, given a model at a certain WER value, how well can you predict the SummaC score if you know which file was used.

Dep. Variable	**Predictor**	**Coef. (no FE)**		**Coef. (with FE)**		R^2 **(no FE)**		R^2 **(with FE)**	
		β	p	β	p	R^2	**adj.** R^2	R^2	**adj.** R^2
ΔX_{SummaC}	X_{WER}	0.1713	0.131	0.0727	0.683	0.103	0.089	0.188	0.155
ΔX_{SummaC}	$\Delta X_{SimDist}$	0.0827	0.350	-0.0130	0.919	0.101	0.087	0.191	0.158
$X_{BERTScore}$	X_{WER}	-0.2598	0.065	-0.1792	0.377	0.456	0.448	0.587	0.569
X_{ROUGE}	X_{WER}	-0.2596	0.074	-0.3060	0.154	0.642	0.636	0.716	0.704
$X_{BERTScore}$	$X_{SimDist}$	0.1712	0.121	-0.0868	0.555	0.451	0.442	0.586	0.569
X_{ROUGE}	$X_{SimDist}$	0.2569	0.022	0.0209	0.893	0.651	0.646	0.714	0.702

Table 12: Difference between previous OSL approach and adding the audio file as a FE

5 Discussion

5.1 Experiment 1

Whisper's general (*Fig 5*) and benchmark performance *Fig 6* stand in sharp contrast with each other, which becomes apparent when comparing their shares of total WER (*Fig 8*). This is likely due to the fact that during the creation of the baseline ground truth files, Whisper was used as the baseline transcript for around 40% of the files. Even though these were put through a different GenAI model first to assign speakers and fix obvious mistakes, as well as being rigorously checked for errors and corrected using the reference audio file by hand, small human errors can still slip into the ground truth. And in case they do, say a small error was not spotted, they favor the model that created the transcript. A few were not even put through a GenAI model to begin with, which were the 50+ minute transcripts. These are clearly visible (Fig 10) as outliers. When transcribing audio manually to create ground truth transcripts, it is recommended to use an ASR model that is not present in the actual set that is run during the experiment to prevent this bias.

The hardware performance has no real explanatory value. The lack of optimizations makes them incomparable to the highly competitive RTFx scores used in the other benchmark (*Fig 6*). The Resource profile does show that there is a lot of room for optimization. Especially for *Cohere*.

The different properties of audio as defined in the methodology did not seem to have any correlation with accuracy (*Fig 10-11*), literature [AvEFG21] and existing benchmarks on datasets like MSL [Fac26a] suggest that models do perform better on certain types of audio. Given the complexity of audio, small datasets, and arbitrary categories, no conclusions can be drawn, but so far it seems that pacing and audio length have no direct relation, and instead they are different factors.

When comparing WER to SimDist performance, the paper that created the SimDist [KAL+21] suggests that there would be a clear relationship between these with an R-squared of around 0.42 to 0.52 based on the dataset. However, in this experiment the correlation is much stronger, with an R-squared of 0.83 (*Fig 13*). This can be caused by many different things, like the Dutch embedding model being worse than the English one or the heavy abstraction and high level of normalization. The second possibility is corroborated by the increase in WER and SimDist per level of abstraction. Applying the mentioned methods pushes WER down faster than SimDist up, making them correlate more strongly. Overall, the results of these metrics seem to line up.

5.2 Experiment 2

The findings from Spearman's r coefficient of summary metrics and the compression rate (*Table 5*) show a concern regarding bias within those metrics. For the important ones, Summac and BERTScore, this does not seem to be relevant, but ROUGE does show some correlation with a score of 0.349. This is a known bias towards longer summaries and is a limitation of ROUGE as a metric for comparing different generated outputs [SSDN19]. However, 0.349 is not as high, so ROUGE itself is still an independent variable that encodes some information about the models

without being solely dependent on the compression rate.
Regarding the GenAI models' individual performance (*Table 4*), it appears to be in line with general GenAI benchmarks [LLM26].

GenAI Model	Rank	External rank	External Rank (relative)
Opus 4.8	1	10	1
GPT 5.5 High	1	14	3
Sonnet-5	3	13	2
Gemini 3.5 Flash Low	4	31	6
Gemini 3.5 Flash High	5	31	6
Gemini 3.1 Pro high	6	29	4
Gemini 3.1 Pro Low	8	29	4
Haiku 4.5	8	135	8

Table 13: GenAI model performance compared to existing LLM-benchmarks

The benchmark did not differentiate between different thinking levels, but the models that score well on the general benchmark tend to perform better in this experiment as well.

The transcript accuracy metrics showed no direct correlation with the summarization ones (*Fig 19-20*). Adding the in-model variance from the GenAI models did not increase the predictive capabilities of the OSL model.

WER vs SummaC Delta (*Table 6*) An $coef$ of 0 is expected for each model since we know for a fact that for any summary $f_s \in F_{summary}$, $h(f_s)_{WER} = 0$ and $\Delta f_{sSummac} = 0$ since the ground truth was used to calculate it. This means that the higher the absolute value of the $coef$, the more volatile and inconsistent a model is. Two models had significant p-values. *Haiku* had a significant ($p = 0.026$) deviation ($coef = -0.0411$) from the baseline model. *Gemini 3.1 Pro Low* ($coef = 0.0459$, $p = 0.013$) had the same. The sign doesn't really matter here, just how large the number is. This shows weaker models have lower consistency and more hallucinations. The following plot visualizes the effect of high $coef$s by comparing a high $coef$ model with a low one (*GPT-5.5*, the intercept). It plots every audio file as a separate line plot for each model.

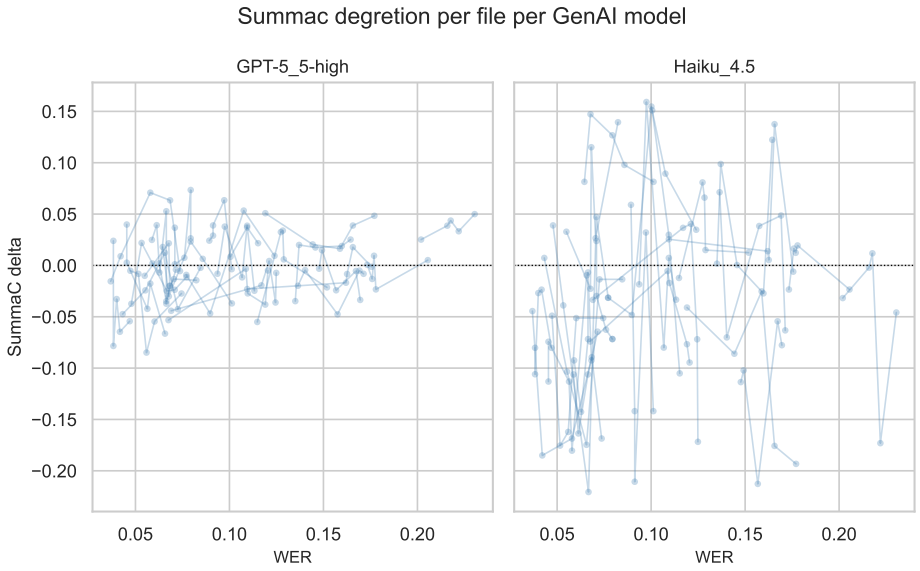

Figure 21: Difference in consistency across two models

The WER's slope of 0.17 shows low correlation and is insignificant ($p = 0.131$) as well. No model had any significant effect once the WER was added to the model. This means that, within a model, none of them show any degradation of faithfulness once the WER increases. The model itself has an adjusted R-squared score of 0.089, which is practically as low as without model variance ($r^2 = 0.01$, *Fig 20*).

SimDist vs SummaC Delta (*Table 7*) SimDist also had similar predicted intercepts for each model with significant p-values for Haiku ($coef = -0.0388$, $p = 0.06$) and Gemini 3.1 Pro Low ($coef = 0.0444$, $p = 0.01$). This time, the third weakest model, Gemini 3.1 Pro High, also became significant ($coef = -0.0280, p = 0.045$). The SimDist slope of 0.0827 is not significant ($p = 0.35$) but points in the right direction. Adjusted R-squared (0.087) shows no correlation between SimDist, the GenAI model, and SummaC. Overall, the effect of GenAI models is negligible.

WER vs ROUGE (*Table 8*) In this case lower intercepts imply lower general accuracy. Hence why, *Opus* scores similarly to the baseline intercept of *GPT-5.5*, whereas *Haiku* performs much worse. This reflects the general performance from earlier. Now the regression model explains a large amount of the variance with an adjusted R-squared of 0.636. The slope of the WER (-0.260) is almost significant ($p = 0.074$). But this relation seems to be only strong for the weaker models. The stronger models, like *Opus* (-0.068) and *Gemini 3.5 Flash High* (-0.0166), show really high robustness to errors in the transcript.

WER vs BERTScore (*Table 9*) Even though the numbers differ slightly, the same behavior as with ROUGE appears in this table for each model. However, the overall correlation of WER and BERTScore, accounting for models, drops compared to that with ROUGE. It achieves an adjusted R-squared of 0.448. To visualize the identified robustness of the better models, a similar graph to that of *Fig*21 is created.

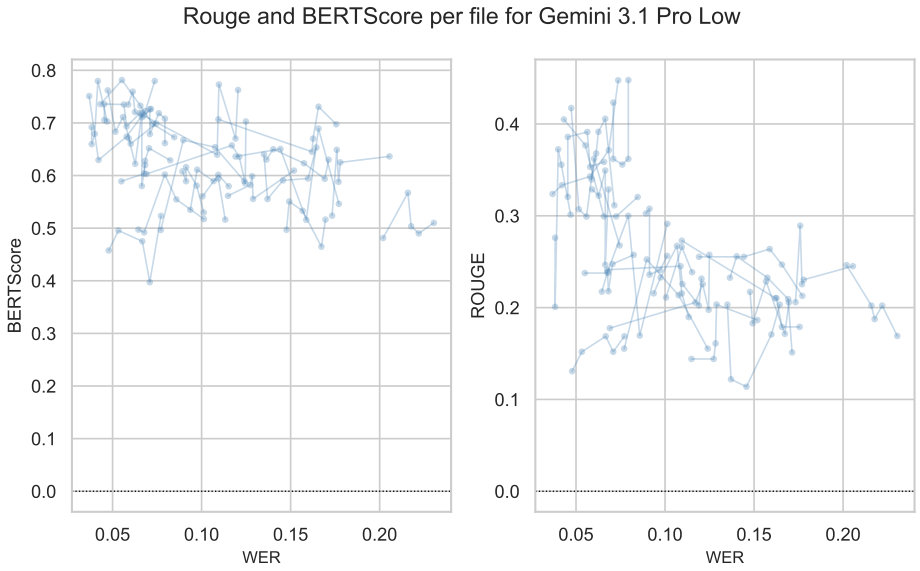

Figure 22: Non-robust GenAI model's relation with WER

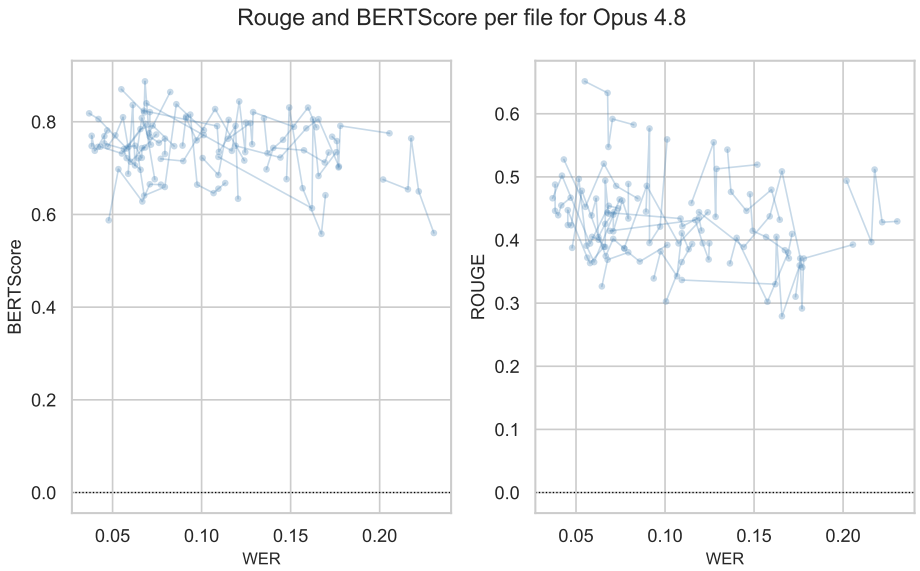

Figure 23: Robust GenAI model's relation with WER

SimDist vs ROUGE and BERTScore (*Table 10-11*) SemDist has the same problem of having a significant correlation with ROUGE, but having the better models be way less affected by it.

Including the audio files as FE This did not work as hoped. This was supposed to show relations within files, but the variance within a file across models was too large. Also, the dataset was too small to properly predict and overfit the data, resulting in incredibly high p-scores. The increase in the R-squared was also really small and not comparable to the increase in predictive capabilities by adding the models.

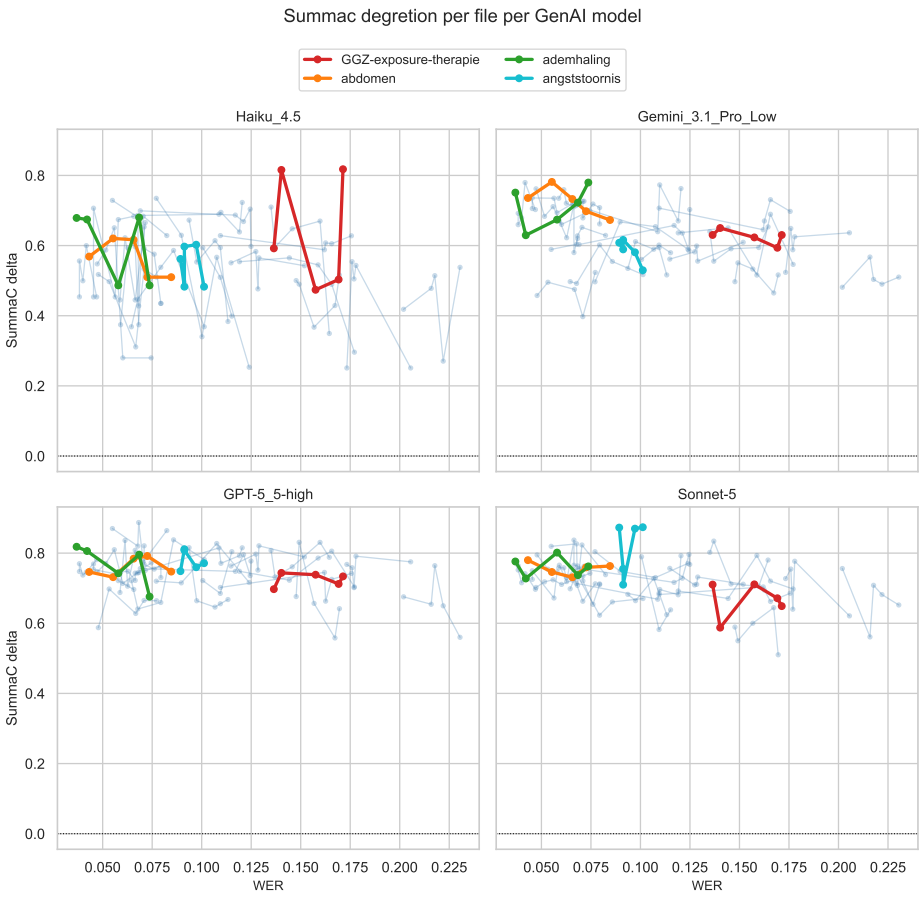

Figure 24: Specific audio files across different models

Again, the volatility with the weaker models is visible, but now it also shows how a specific audio file can lead to completely different scores based on the GenAI model. This difference in score is not even absolute but also relative, creating different line graphs per model.

6 Conclusion

This research has shown that, with the current set of selected metrics, it is not possible to identify an obvious best ASR model for The Salvation Army, or that a slightly better ASR model even improves the documentation capabilities at all. *Whisper* does appear to be a good choice for a model, but so does *Cohere*, which doesn't have the ground truth bias, has already proven to perform best at Dutch audio, and has a lot of space for optimization. From the current metrics, SemDist is the best for identifying the best ASR model, being resilient to noise while remaining accurate. The summary metrics are not strong enough to confidently rate a summary. Creating a benchmark that can do so with certainty will require more work. Changing from GPT-4 to a stronger model is expected to result in higher quality summaries than a change in ASR model could.

6.1 Research Questions

The models did score fine on the domain-specific audio. It performed slightly worse on general Dutch audio datasets, which was expected. No relation between the audio-specific features and the accuracy scores was found.

The accuracy metrics showed near-significant correlations for WER with ROUGE and BERTScore and a significant correlation with SimDist and ROUGE. However, when looking at specific models, this correlation was not present with the better models, showing high robustness to errors. So, given that, for state-of-the-art GenAI models, summarization metrics show no correlation with transcript metrics, it can not be said that better ASR models lead to more accurate and higher-quality summaries.

Since the previous question showed no convincing correlation, this benchmark is unable to determine the business value an ASR model produces since its performance can not be properly linked to the final summary. This is an combination of the lacking correlation with higher models and the compounding effect of technical summary metrics not perfectly aligning with human judgment.

6.2 Future Work

Ultimately, this research has raised more questions than it answered. Audio files are incredibly complex and have many different factors that could affect transcription. Due to the small amount of data, no conclusions could be made about what makes an audio file easy or hard to transcribe. Exploring the robustness of models by manually introducing noise or using human feedback to categorize audio could be interesting for better benchmarking.
The audio was also not specific enough to the use case. As was already apparent early on, a lot of work still has to be done in creating high-quality, long-form audio of real conversations between clients and healthcare professionals.

This research was limited to technical methods and did not take into account which info is important and which is not. Coming back to the background section $MedicalWER$ still seems to be a great option but requires annotation. Developing annotation techniques for Dutch medical speech or business-specific terminology can make it easier to incorporate this powerful metric.

Besides the type of metrics, it also didn't predict speaker roles through diarization. These can have a great impact on the final summary. The created dataset can be used for this purpose and includes speaker roles.
Human feedback on summary quality would also be a great addition but requires a lot of work and well thought out set up.

However, this research does show that for businesses themselves, the most promising improvements to existing documentation pipelines remain in fields outside of ASR models. The power of the stronger GenAI models to fill in missing data and correct for transcription errors clearly exists, but to what extent is not sure. They might be good at trivial sentences but struggle with more specific things.

Finally, LLM-as-a-judge, which ended up not being incorporated in the experiments, is a great alternative to human feedback and is in an early stage of research. Incorporating this in benchmarks has the potential to align them better with human judgment while being scalable and cheaper.

References

[AJD24] Ayo Adedeji, Sarita Joshi, and Brendan Doohan. The sound of healthcare: Improving medical transcription asr accuracy with large language models, 2024.

[AvEFG21] Alëna Aksënova, Daan van Esch, James Flynn, and Pavel Golik. How might we create better benchmarks for speech recognition? In *Proceedings of the 1st workshop on benchmarking: Past, present and future*, pages 22–34, 2021.

[Bre17] Hervé Bredin. pyannote. metrics: A toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems. *hypothesis*, 100(60):90, 2017.

[CAB+05] Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, et al. The ami meeting corpus: A pre-announcement. In *International workshop on machine learning for multimodal interaction*, pages 28–39. Springer, 2005.

[dGZMS26] Dimme de Groot, Yuanyuan Zhang, Jorge Martinez, and Odette Scharenborg. A semi-spontaneous dutch speech dataset for speech enhancement and speech recognition, 2026.

[Fac26a] Facebook. Multilingual librispeech, 2026.

[Fac26b] Facebook. Voxpopuli, 2026.

[FER+18] Gregory Finley, Erik Edwards, Amanda Robinson, Michael Brenndoerfer, Najmeh Sadoughi, James Fone, Nico Axtmann, Mark Miller, and David Suendermann-Oeft. An automated medical scribe for documenting clinical encounters. In Yang Liu, Tim Paek, and Manasi Patwardhan, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 11–15, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[GEGK21] Ludmila Gordeeva, Vasily Ershov, Oleg Gulyaev, and Igor Kuralenok. Meaning error rate: Asr domain-specific metric framework. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 458–466, 2021.

[GJS+24] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *The Innovation*, 2024.

[GNA18] Max Grusky, Mor Naaman, and Yoav Artzi. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719, 2018.

[Goo26] Google. Fleurs, 2026.

[GRB25] Shannon Gallagher, Swati Rallapalli, and Tyler Brooks. Evaluating llms for text summarization: An introduction. Carnegie Mellon University, Software Engineering Institute's Insights (blog), Apr 2025. Accessed: 2026-Apr-21.

[hf6] hf_audio. hf-audio/open_asr_leaderboard, 2026.

[HMP+25] Qasim Haniff, Zaiqiao Meng, Tanatapanun Pongkemmanun, Zheng Chuang Sia, Heather Newport, Yuning Ooi, and Bhautesh D Jani. Use of artificial intelligence to transcribe and summarise general practice consultations. *Journal of Medical Artificial Intelligence*, 8:43, 2025.

[KAL+21] Suyoun Kim, Abhinav Arora, Duc Le, Ching-Feng Yeh, Christian Fuegen, Ozlem Kalinli, and Michael L. Seltzer. Semantic distance: A new metric for asr performance analysis towards spoken language understanding, 2021.

[KUS25] Betty Kurian, Abhinav Upadhyay, and Abhijeet Sengupta. Domain-specific adaptation for asr through text-only fine-tuning. In *Proceedings of the 1st Workshop on Multimodal Models for Low-Resource Contexts and Social Impact (MMLoSo 2025)*, pages 78–85, 2025.

[Lab26] Cohere Labs. cohere-transcribe-03-2026, 2026.

[Leg26] Leger des Heils. Jeugdbescherming. 2026.

[LGY+21] Bo Li, Anmol Gulati, Jiahui Yu, Tara N Sainath, Chung-Cheng Chiu, Arun Narayanan, Shuo-Yiin Chang, Ruoming Pang, Yanzhang He, James Qin, et al. A better and faster end-to-end model for streaming asr. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5634–5638. IEEE, 2021.

[Lin04] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries, 2004.

[LLM26] LLM Stats. Llm stats: Aggregated llm benchmark results, 2026.

[LSBH22] Philippe Laban, Tobias Schnabel, Paul N Bennett, and Marti A Hearst. Summac: Revisiting nli-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177, 2022.

[MLG11] Taniya Mishra, Andrej Ljolje, and Mazin Gilbert. Predicting human perceived accuracy of asr systems. In *INTERSPEECH*, volume 11, pages 1945–1948, 2011.

[Mol25] Manuel Mol. How can the addition of genai enhance the record-keeping capability for interviews and meetings in public healthcare organizations? In *The impact of generative AI on the documentation capability in the public health sector*, pages 1–4, 2025.

[Moz26] Mozilla. Common voice scripted speech 25.0 - dutch, 2026.

[NVI25a] NVIDIA. Nvidia canary 1b v2, 2025.

[NVI25b] NVIDIA. parakeet tdt 0.6b v3, 2025.

[Ope24] OpenAI. Openai whisper large v3, 2024.

[PYC+26] Zhiliang Peng, Jianwei Yu, Yaoyao Chang, Zilong Wang, Li Dong, Yingbo Hao, Yujie Tu, Chenyu Yang, Wenhui Wang, Songchen Xu, et al. Vibevoice-asr technical report. *arXiv preprint arXiv:2601.18184*, 2026.

[Qwe26] Qwen. Qwen3 asr 1.7b, 2026.

[SSDN19] Simeng Sun, Ori Shapira, Ido Dagan, and Ani Nenkova. How to compare summarizers without target length? pitfalls, solutions and re-examination of the neural summarization literature. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 21–29, 2019.

[SWG+26] Xian Shi, Xiong Wang, Zhifang Guo, Yongqi Wang, Pei Zhang, Xinyu Zhang, Zishan Guo, Hongkun Hao, Yu Xi, Baosong Yang, et al. Qwen3-asr technical report. *arXiv preprint arXiv:2601.21337*, 2026.

[SXK+20] Gabriel Synnaeve, Qiantong Xu, Jacob Kahn, Tatiana Likhomanenko, Edouard Grave, Vineel Pratap, Anuroop Sriram, Vitaliy Liptchinsky, and Ronan Collobert. End-to-end asr: from supervised to semi-supervised learning with modern architectures, 2020.

[SZB+25] Vaibhav Srivastav, Steven Zheng, Eric Bezzam, Eustache Le Bihan, Nithin Koluguri, Piotr Żelasko, Somshubra Majumdar, Adel Moumen, and Sanchit Gandhi. Open asr leaderboard: Towards reproducible and transparent multilingual and long-form speech recognition evaluation, 2025.

[tvo26] tvosch. Dutch-asr-leaderboard, 2026.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[VZWH+23] Daphne Van Zandvoort, Laura Wiersema, Tom Huibers, Sandra van Dulmen, and Sjaak Brinkkemper. Enhancing summarization performance through transformer-based prompt engineering in automated medical reporting. *arXiv preprint arXiv:2311.13274*, 2023.

[WKA+18] Ian Williams, Anjuli Kannan, Petar S Aleksic, David Rybach, and Tara N Sainath. Contextual speech recognition in end-to-end neural network systems using beam search. In *Interspeech*, pages 2227–2231, 2018.

[ZK12] Andrej Zgank and Zdravko Kacic. Predicting the acoustic confusability between words for a speech recognition system using levenshtein distance. *Elektronika ir Elektrotechnika*, 18(8):81–84, 2012.

[ZKW+19] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[ZLD+24] Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B. Hashimoto. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57, 2024.