**Intelligent Speech-to-Text Systems: Enhancing Documentation with NLP-Driven Transcription and Summarization**

Zi-Yang Chang1, a), Su-Cheng Haw1, 2, b), Palanichamy Naveen1, 2, c) and Jayapradha J3, d)

*1Faculty of Computing and Informatics, Multimedia University, Jalan Multimedia, 63100 Cyberjaya, Malaysia.*

*2Center for Digital Innovations, CoE for Immersive Experience, Multimedia University, Persiaran Multimedia, 63100 Cyberjaya, Malaysia.*

*3Department of Computing Technologies, SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu 6030203, India.*

*b) Corresponding author: sucheng@mmu.edu.my*

*a) 1211200628@student.mmu.edu.my*

*c) p.naveen@mmu.edu.my*

*d) jayapraj@srmist.edu.in*

**Abstract.** In this highly digitized world, the need for efficient and accurate speech-to-text solutions is becoming increasingly important. Manual transcription is a lengthy process often accompanied by numerous mistakes, thus creating considerable hurdles for individuals and organizations managing large amounts of spoken content. This paper offers an integrated advanced speech recognition solution along with Natural Language Processing (NLP) methodologies to achieve automated transcription, producing structured meaningful documentation instead. The system in question not only renders the unstructured audio into precise text but also employs summarization algorithms to condense the content into brief and coherent summaries. To truly test its viability in real-life scenarios, the experiments described herein focus on assessing both transcription accuracy and semantic quality in relation to the provided summaries. The findings reveal intelligent speech-to-text systems' capabilities in transforming audio data into usable information indeed actionable while further facilitating automated documentation smart content management broad area.

# INTRODUCTION

Artificial Intelligence (AI) and Natural Language Processing (NLP) have totally revolutionized how digital platforms handle, process and pass on information. With advancements in speech recognition technologies and their integration into Internet-connected devices and services, automated transcription systems have become an important tool to enhance productivity. These systems simplify the task of converting spoken language into written text, streamlining both documentation and real-time transcription processes.

For the text summarization, Rahul [1] reviewed various NLP-based machine learning methods for text summarization and distinguishing the difference between extractive and abstractive methods. They found that neural networks, reinforcement learning, and sequence-to-sequence models significantly enhance summarization accuracy. Besides, Zhang [2] proposed PEGASUS, which is a transformer-based model pre-trained for abstractive summarization using the Gap-Sentences Generation (GSG) objective, and this model has achieved state-of-the-art performance on multiple benchmarks.

For transcription, the speech recognition system for the Estonian language developed by O. Lev [3] has reported a Word Error Rate (WER) of 6.9% in transcribing speech. The system could not handle user-submitted audio recordings. WhisperX is a fully time-aligned speech recognition system, specifically designed for long-form audio transcription, incorporating Voice Activity Detection along with phoneme alignment [4]. The audio-to-text conversion in WhisperX relies on OpenAI's Whisper [5] model but does not handle low-resource languages or domain-specific content appropriately. An audio-to-text conversion model using Google's Speech Recognition is presented by Singh et al., who reports an accuracy of 85.15% [6]. This model requires further development for noisy conditions and different accents.

## Automatic Speech Recognition (ASR)

Automatic Speech Recognition (ASR) is part of AI that converts speech to text using machine learning and NLP. Using advanced algorithms and neural networks, ASR can process audio inputs, detect spoken languages, and generate precise text transcription, and other tasks. Voice assistants such as Siri and Alexa, chatbots, and video subtitles are just some of the applications that use this technology. ASR is almost becoming a necessity; it enables everyone to access audio content easily while making the transcription process simple [7][8].

ASR comprises five steps: preprocessing, acoustic modeling, language modeling, decoding, and post-processing. The first step or preprocessing cleans the raw audio input and makes it suitable for further processing. Feature extraction is the second step where the audio signal is transformed into meaningful representations. Acoustic modeling determines the relationship between acoustic signals and phonemes using statistical approaches or deep learning. Language modeling provides linguistic and contextual information to the recognized phonemes to form coherent words into meaningful sentences. Decoding ensures that what text has been generated corresponds to what has been spoken in the audio. Post-processing addresses errors related to punctuation, capitalization, grammar, etc., hence making transcription more readable and usable. The output may also be enriched by summarization, emotion detection, or translation as an application of post-editing. In the end, ASR makes sure that the final output is polished and professional enough to be used in various contexts. ASR has been used in various fields, such as the health sector, Telecommunication industry, Robotic industry and so on [9]. Here, we will discuss the techniques of ASR.

### Traditional Method (HMM-based ASR)

Acoustic modelling is the core component of ASR systems. This part maps the audio signal to the phonemes that represent the spoken language. Traditional statistical methods such as Hidden Markov Models (HMMs), which are a statistical tool that is widely used in pattern matching, are used for speech recognition because of their ability to model effectively [10].

HMMs work by creating a probabilistic model of sequences, where each state represents a phoneme or acoustic unit. This allows the system to recognize speech based on statistical correlations between sound patterns and phoneme sequences. In short, HMMs create a chain from the sequence of phonemes to model and transcribe spoken words [11]. HMM can be improved by employing statistical sequences of the occurrence of these states. It does this by automatically calculating the probability of moving from one network state to another [12].

### End-to-end model (E2E)

The term “end-to-end” is means including all the stages of a process, so for the end-to-end model is means that model that enables joint training from scratch [13]. Basically, it can handle almost all the phases and some specific tasks to customize or enhance the result. It simplifies the process of ASR, which folds the acoustic, pronunciation and language models into one neural network with a much smaller number of parameters than a conventional ASR system [14], making the process more efficient and scalable. The popular end-to-end models that are currently used by the public are recurrent neural network transducer (RNN-T), RNN attention-based encoder-decoder (AED) and Transformer-AED [15]. These models directly map audio inputs to text, RNN-T focuses on efficient streaming transcription, RNN-AED uses attention mechanisms for sequence alignment, and Transformer-AED utilises self-attention for superior parallel processing and context capture. [16] aimed to transcribe multi-talker in a conversation or meeting without affecting the recognition accuracy. Besides that, [17] proposed a system that can real-time transcription by using E2E and achieve the result of 2-4% increase in WER but reduced delay by 1.5-2 seconds.

## Text Summarization

In an age where vast amounts of information are generated daily, the ability to extract key information quickly has become important. Text summarization is a technique that increases productivity and decision-making because it provides concise representations of complicated texts to let people understand the lengthy content in a short description. Despite its potential, text summarization still has lots of limitations that limit its capabilities. According to [18], one of the significant challenges is abstractive summarization, which requires deeper understanding to generate a new sentence, and it might generate a summary that did not relevant with the original text. Additionally, dealing with special domains such as medical is another challenge because the summarization models may not understand and represent specific vocabulary accurately. In addition, the summarization process will become difficult when processing data that has grammatical mistakes or irrelevant content. Furthermore, when handling lengthy documents, it is hard to let the model generate summaries that are coherent and logical. Advancements in deep learning have led to the development of state-of-the-art models such as PEGASUS, T5, and BART, which these models are using transformer architectures to enhance summarization performance.

The workflow of text summarization is separated into four phases, which are preprocessing, sentence scoring or selection, summarization generation, and post-processing to get the final summaries. In the preprocessing, the unwanted characters, symbols, and stop words were removed; words or sub-words were broken down using tokenization [19]; words were reduced to their base form via lemmatization. Next, the sentence scoring model determines which sentence is important; scores are given based on relevance, keyword frequency, and contribution to the main topic. Subsequently, summarization generation focuses on creating a concise summary that captures the essence of the original content. Some advanced models, such as Transformers or seq2seq architecture, are often used in this phase to generate fluent, coherent summaries. After the summary is generated, the post-processing is to ensure that the output is polished and readable. This phase includes steps such as correct grammar and punctuation, ensuring proper sentence structures, and making any necessary adjustments to improve the clarity and flow. Here will be discussion for the techniques of text summarization, which are mainly separated into two categories which are Abstractive Summarization (ABS) and Extractive Summarization (EXT).

### Abstractive Summarization (ABS)

Abstractive summarization uses techniques such as sequence-to-sequence model or Transformer-based architecture to generate the summaries. These models are trained to understand the meaning of the input text and generate a shorter version that remains the core ideas of the original text. In addition, ABS involves encoding the entire document into a fixed-length vector, letting the machine process it numerically, then decode into a shorter, coherent summary that understandable by humans [18]. [20] using the ABS to summary the text for specific language by using the model called Bidirectional encoder representations from transformers (BERT). Other applications, such as news, scientific papers, and reports, show that the ABS can provide greater flexibility and coherence than EXT. Still, it needs a large-scale of training data to get a better output [21].

### Extractive Summarization (EXT)

EXT is the selection of important phrases, sentences, or paragraphs directly from the text to generate the summary. [22] states that it is a summary that consists entirely of the extracted content, so summary sentences are sentences or words obtained from the original text, some kind of copy-paste approach. This method is to identify the most informative parts of the text without generating new content. Techniques like Term Frequency-Inverse Document Frequency (TF-IDF) or more advanced methods such as TextRank are often used to rank the sentence by their relevance to the overall text, and finally the highest-ranked sentences are then selected and combined to form the summary. [23] used EXT to design a model that for long documents like scientific paper, which have more than 500 sentences. Through its characteristics, it is widely used in cases that need an accurate and quick summary, but it has a weakness in terms of coherence between the sentences in the summary.

## Proposed Framework

The workflow of the end-to-end speech processing system is divided into four phases, preprocessing, speech recognition, NLP processing, and formatting, to get a transcription aside from the text summarization. Figure 1 shows an overview of the workflow of the proposed speech-to-text system, which included preprocessing, speech recognition, NLP processing and formatting.

**FIGURE 1.** Phases in the ASR process

Preprocessing is the initial phase of a speech recognition system, preparing audio input for the model. Techniques like filtering, voice activity detection, and spectrogram generation are used to improve the quality of the audio signal. Speech recognition is the most crucial phase, converting the audio into text using Acoustic Modeling and Language Modeling. Statistical methods like HMMs and GMMs are used to predict the text accurately. NLP techniques are applied to enhance the structure and coherence of the text. The formatting phase refines the output to meet user expectations, ensuring better readability. The dataset used in this paper is the Audio dataset on Kaggle, which is called “TED Talks Web-scraped dataset”, a comprehensive collection that includes around 5k audio files about the TED talks. This dataset contains descriptive information for each audio file, making it suitable for transcription and summarization because it has the ground truth to compare and evaluate the model.

Before evaluating model performance, the dataset must be cleaned and preprocessed to ensure accuracy. First, missing or null values are identified using isna().sum() and msno, and rows without ground truth or YouTube video codes are dropped. Next, duplicate values are checked with duplicated().sum(), confirming none exists. Numerical conversions are applied to transform “K” and “M” into thousands and millions, respectively, for better visualization. The audio duration is capped at 300 seconds to standardize length. YouTube codes are then combined with the base URL for automated audio downloads. To improve transcription evaluation, parentheses, punctuation, and special symbols are removed, and rows containing only "(Music)" are dropped. Finally, after downloading the audio, private or unavailable videos are removed, finalizing the cleaned dataset in CSV format. After capping the duration, there are about 925 audio files that need to be downloaded before passed to the Whisper model. This is the automated YouTube video downloaded through the youtube\_code\_link column using the Python code as script. Using yt\_dlp, this library can download videos from YouTube. After doing the data preprocessing (filter of the video and cleansing of the excel for the comparison), we fed the video into Whisper and PEGASUS and evaluated the performance by comparing the generated output and the ground truth that was provided in the dataset.

### Whisper – Transcription

Whisper is a transformer-based automatic speech recognition (ASR) model which is developed by OpenAI. It is trained on a large multilingual and multitask dataset and make it able to handle a variety of speech styles and background conditions. In this study, the Whisper medium model is used to transcribe English TED Talk audio files.

After downloading the videos from YouTube, each video is preprocessed before being passed to the Whisper model for transcription. The preprocessing involves resampling the audio stream to 16 kHz using Ffmpeg and audio is saved in WAV format to ensure compatibility with the model’s input requirements and consistency. Once resampled, the video is processed using the Whisper medium model to generate transcriptions. This process is repeated for each video in the dataset, and the resulting transcriptions are collected for further analysis.

### PEGASUS – Summarization

PEGASUS is a transformer-based abstractive summarization model developed by Google Research. It is pre-trained by using a novel self-supervised objective called Gap-Sentences Generation (GSG), where important sentences are masked and the model learns to generate them from the remaining content. In this study, PEGASUS model is used to generate summaries from transcribed TED Talk content, focus on convert lengthy speech transcription into concise and readable summaries.

After getting the result that generated from Whisper model, the next step is to summarize the text by using PEGASUS. Before that, each text file is preprocessed by removing the irrelevant symbols, all lower-case words, correcting the length of the transcription to make the length of the ground truth, and ensuring that the text is properly segmented into coherent paragraphs. These cleaned texts are then passed to PEGASUS model for the abstractive summarization. This process is repeated for each transcription file in the transcription folder, and the resulting transcription will be analysis in the future.

# EVALUATION METRICS

## Word Error Rate (WER)

In speech recognition and transcription, one of the metrics to evaluate the accuracy of transcription is Word Error Rate (WER). WER is to measure the difference between the transcribed text and the reference (ground truth) text, providing a measurement of the transcription quality. However, WER should be used with cautions in some conditions because this metric treats all errors equally and does not consider the context of words or their importance within a sentence. For example, High WER but the same meaning: “The cat sits on the mat” and “the cat is sitting on the mat”, and Low WER but lost meaning: “I love rainy day” and “I love sunny day”.

For two sequences of words, the transcribed output and the reference ground truth, the WER is calculated as Equation (1), where: S: The number of words that need to be replaced to match the reference; D: The number of words that are missing from the transcribed output; I: The number of extra words that appear in the transcribed output; N: The total number of words in the reference text.

(1)

In conclusion, WER is only concerned with word-level errors, and therefore, the situations mentioned above cannot give a complete picture of how a human understands transcription. By combining human evaluation and other semantic metrics, we could get a better understanding of the performance of systems.

## BERTScore

In text summarization, one of the key metrics for evaluating the similarities between generated summary and the reference summary (ground truth) is BERTScore. BERTScore uses deep contextual embedding from transformer-based models to provide a comprehensive evaluation of text quality compared to the traditional metrics like ROUGE or BLEU. Traditional metrics which focus on word level, BERTScore evaluate summaries based on semantic similarity and fluency, making it a more effective metric for evaluating the quality of abstractive summarization systems. BERTScore provides a semantic-oriented evaluation of text generation, making it useful for tasks such as summarization, translation, and paraphrasing. However, there are some situations that need to be used with caution like high BERTScore but different factual accuracy, which means that “The capital of France is Paris” and “The capital of France is Rome”, this kind of situations might get a high BERTScore, but it is factually wrong.

In conclusion, BERTScore is a powerful metric to evaluate the quality of text summarization, but it is not a perfect metric on its own. Human assessment or factual correctness check is needed to get a complete picture of the generated summary’s quality.

# RESULTS AND DISCUSSION

This section discusses the performance of the model and how they work in this dataset through two evaluation metrics: WER and BERTScore. Once we get the result from Whisper and PEGASUS, we combine the result that generated from the model and the ground truth that provided in the dataset into a csv file to better comparison process and checked that any mismatch file that is in the dataset but not generated by the model. After combining into a csv file, then we start to run the evaluation metrics through csv file. Table 1 shows the performance of the transcription and summarization models based on two evaluation metrics: Word Error Rate and BERTScore for transcription and summarization.

The results are obtained by comparing each of the videos with their corresponding ground truth, and list them in the table above to have a better understanding of the result and to analyze why some of the results are worse. The dataset includes a diverse range of content, such as conversational videos, interviews, and music clips like beat box and autotuned performances.

|  |  |  |
| --- | --- | --- |
| **TABLE 1.** Evaluation result | | |
|  | **WER** | **BERTScore** |
| Transcription – Whisper | 0.0578 | - |
| Summarization – PEGASUS | 1.9987 | 0.8157 |

Table 1 shows that Word Error Rate (WER) for each video-transcription pair. The average WER result of the Whisper is around 0.0578, which means that the accuracy of Whisper is about 94%, showing that the strong performance of Whisper for most video of the database. However, the result is affected by some music videos such as Beatbox, autotuned that are sound not like a person speaking, where it struggles to detect the words.

Besides that, this table also shows both WER and BERTScore (F1) side-by-side for a comparative analysis. The average WER result of PEGASUS, which can be said to be totally different from the summary that is given by the dataset at the word level. However, Fig. 1also shows that there are around 0.8157 of the average BERTScore for the model, which means despite the high WER, the generated summaries are semantically like reference summaries.

The reason is that BERTScore is focused on the semantic level, but WER is more focused on the word level. Since the Abstractive Summary (ABS) extracts the key information and regenerates or restructures the sentence, if checked word by word, the error rate will be very high. These findings suggest that while WER is useful for evaluating extractive or speech recognition tasks, BERTScore is more suitable for assessing the quality of abstractive summaries. Therefore, both metrics should be considered together for a comprehensive evaluation. While Whisper demonstrated high accuracy under standard speech conditions, it struggled significantly with non-verbal audio elements such as beatboxing or autotuned due to its training data being biased towards clean, speech-heavy content. Similarly, PEGASUS sometimes produces summaries that contain hallucinated facts, particularly when the input transcripts are noisy.

After we got the results, we started to analysis why the results looked like this. Firstly, for the transcription, we noticed that the whisper cannot process very well for the audio that contains autotuned, or some special voices that are not like the proper speech, which means that it more tends to process speech audio instead of the music audio. For the summarization, we noticed that the result summarization well, but the result is not that high in marks, and it might be because the ground truth of summary in the dataset is the description of the audio, is not the proper summarization ground truth, and maybe it is the issue for this result. To overcome these issues, we plan to unify the process and deploy this model and let the public use it to have a better understanding of how the model works, and the result will be saved in google forms to further analysis. As a result, the comparison using BERTScore may not accurately reflect the true quality of the summaries generated, since it evaluates structurally and semantically different text forms.

# CONCLUSION

In this paper, we have successfully implemented transcription and summarization models, focusing on their accuracy and effectiveness. The transcription model was evaluated using Word Error Rate (WER) to measure how well the system can convert the speech into text, and the summarization model was accessed using BERTScore to ensure the generated summaries were coherent and relevant. Although the computational cost of the transcription model is relatively high, the accuracy that it achieved makes it a reasonable trade-off.

In our future work, we will focus on making the system functional and accessible to users who can easily upload audio or video files, process them, and download the results in Word or PDF format. To overcome current limitations, we aim to explore model fine tuning, especially PEGASUS to improve accuracy and coherence. Finally, the system will be surveyed by users through Google Forms to analysis any errors and improvement through real-world feedback.

# REFERENCES

1. Rahul, S. Adhikari, and Monika, “NLP based Machine Learning Approaches for Text Summarization,” in *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, (IEEE, Erode, India, 2020), pp. 535–538.
2. J. Zhang, Y. Zhao, M. Saleh, and P.J. Liu, “PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization,” (2020).
3. A. Olev, and T. Alumäe, “Estonian Speech Recognition and Transcription Editing Service,” BJMC **10**(3), (2022).
4. M. Bain, J. Huh, T. Han, and A. Zisserman, “WhisperX: Time-Accurate Speech Transcription of Long-Form Audio,” in *INTERSPEECH 2023*, (ISCA, 2023), pp. 4489–4493.
5. A.L. Haz, E.D. Fajrianti, N. Funabiki, and S. Sukaridhoto, “A Study of Audio-to-Text Conversion Software Using Whispers Model,” in *2023 Sixth International Conference on Vocational Education and Electrical Engineering (ICVEE)*, (IEEE, Surabaya, Indonesia, 2023), pp. 268–273.
6. A.R. Singh, D. Bhardwaj, M. Dixit, and L. Kumar, “An Integrated Model for Text to Text, Image to Text and Audio to Text Linguistic Conversion using Machine Learning Approach,” in *2023 6th International Conference on Information Systems and Computer Networks (ISCON)*, (IEEE, Mathura, India, 2023), pp. 1–7.
7. S. Alharbi, M. Alrazgan, A. Alrashed, T. Alnomasi, R. Almojel, R. Alharbi, S. Alharbi, S. Alturki, F. Alshehri, and M. Almojil, “Automatic Speech Recognition: Systematic Literature Review,” IEEE Access **9**, 131858–131876 (2021).
8. K.Q. Yip, P.Y. Goh, and L.Y. Chong, “Social Messaging Application with Translation and Speech-to-Text Transformation,” *Journal of Informatics and Web Engineering* **3**(2), 169–187 (2024).
9. H. Ibrahim, and A. Varol, “A Study on Automatic Speech Recognition Systems,” in *2020 8th International Symposium on Digital Forensics and Security (ISDFS)*, (IEEE, Beirut, Lebanon, 2020), pp. 1–5.
10. P. L, P. Kumar, and D. Singh, “Speech Recognition Using Hmm and Combinations: A Review,” SSRN Journal, (2024).
11. M. Maseri, and M. Mamat, “Performance Analysis of Implemented MFCC and HMM-based Speech Recognition System,” in *2020 IEEE 2nd International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, (IEEE, Kota Kinabalu, Malaysia, 2020), pp. 1–5.
12. J. Joshi, and D. Sharma, “Systematization in Asr Systems: A Review,” (2024).
13. R. Prabhavalkar, T. Hori, T.N. Sainath, R. Schlüter, and S. Watanabe, “End-to-End Speech Recognition: A Survey,” IEEE/ACM Trans. Audio Speech Lang. Process. **32**, 325–351 (2024).
14. B. Li, S. Chang, T.N. Sainath, R. Pang, Y. He, T. Strohman, and Y. Wu, “Towards Fast and Accurate Streaming End-To-End ASR,” in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (IEEE, Barcelona, Spain, 2020), pp. 6069–6073.
15. J. Li, Y. Wu, Y. Gaur, C. Wang, R. Zhao, and S. Liu, “On the Comparison of Popular End-to-End Models for Large Scale Speech Recognition,” in *Interspeech 2020*, (ISCA, 2020), pp. 1–5.
16. L. Lu, J. Li, and Y. Gong, “Endpoint Detection for Streaming End-to-End Multi-Talker ASR,” in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (IEEE, Singapore, Singapore, 2022), pp. 7312–7316.
17. C. Arriaga, A. Pozo, J. Conde, and A. Alonso, “Evaluation of real-time transcriptions using end-to-end ASR models,” (2024).
18. W.S. El-Kassas, C.R. Salama, A.A. Rafea, and H.K. Mohamed, “Automatic text summarization: A comprehensive survey,” Expert Systems with Applications **165**, 113679 (2021).
19. J. Jayapradha, Y. Kulkarni, L.V. G, P. Naveen, and E.A. Anaam, “Treatment Recommendation using BERT Personalization,” *Journal of Informatics and Web Engineering* **3**(3), 41–62 (2024).
20. M. Awais, and R. Muhammad Adeel Nawab, “Abstractive Text Summarization for the Urdu Language: Data and Methods,” IEEE Access **12**, 61198–61210 (2024).
21. A. Khan, K. Khan, W. Khan, S.N. Khan, and R. Haq, “Knowledge-based Word Tokenization System for Urdu,” *Journal of Informatics and Web Engineering* **3**(2), 86–97 (2024).
22. S. Govindapillai, L.K. Soon, and S.C. Haw, “Domain-Independent True Fact Identification from Knowledge Graph,” *JOIV: International Journal on Informatics Visualization* **9** (3), 893-913 (2025).
23. A.P. Widyassari, S. Rustad, G.F. Shidik, E. Noersasongko, A. Syukur, A. Affandy, and D.R.I.M. Setiadi, “Review of automatic text summarization techniques & methods,” Journal of King Saud University - Computer and Information Sciences **34**(4), 1029–1046 (2022).