xData-CogA Intern Report

By

Ignatius Tang

Supervisor

Ngoh Wei Jie

Rachel Tan

Duration

March 2024 – July 2024

Table of Contents

1 Brief Introduction

During my tenure at HTX, I actively participated in several projects: WordSmith, EPC-Chatbot and GenAI. My responsibilities for WordSmith were centred around exploring and benchmarking Natural Language Processing Models, implementing NLP techniques for language identification and summary evaluation using Prompt Flow. For the EPC Chatbot, I crafted an automatic evaluation pipeline with GPT models on police reports based on categories. In the GenAI team, I focused on experimenting with Vision Large Language Models(VLLMs) for Retrieval Augmented Generation (RAG) to expand the app capabilities in computing visual input. As well as text summarisation datasets and metrics research. In the subsequent sections, I will provide a comprehensive overview of my contributions to WordSmith, EPC-Chatbot and GenAI and their outcomes.

2 WordSmith

2.1 Description of Work

WordSmith is an automatic Notes of Meetings summarisation and transcription tool that accurately detects and diarises speakers according to their language. My supervisor, Rachel Tan, delegated the main task of coming up with a pipeline for summary-generation, with GPT-models as the standard, using the Azure ML Prompt Flow tool. Other mini-assignments were to explore the capabilities of the language identifier model Voxlingua107 and benchmarking newly-released code-switch NLP models.

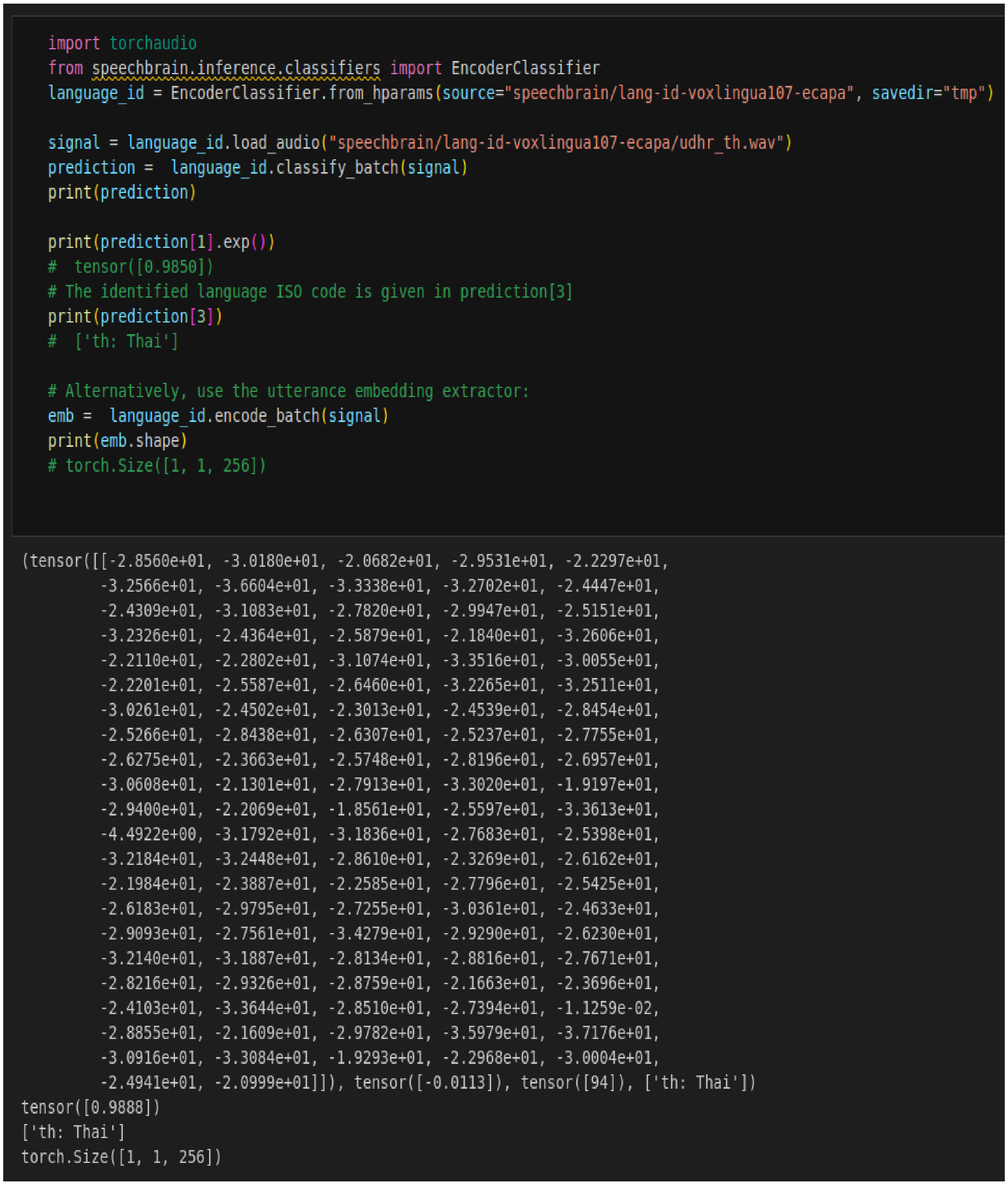
2.1.1 Voxlingua107 Language Identifier Model

The Voxlingua107-ecapa language identifier model is a based on the ECAPA-TDNN architecture and trained on the VoxLingua107 dataset using SpeechBrain, classifies speech utterances across **107 languages** with improved embedding performance for downstream tasks. VoxLingua107 is a diverse speech dataset comprising short segments from YouTube videos labelled by language, totaling **6628 hours** in the training set, with an average of **62 hours per language**, though actual amounts vary widely, and includes a separate development set validated for language accuracy.

Goal: Curate multilingual datasets for Voxlingua107 model evaluation.

The following is the step-by-step explanation of the language identifier workflow:

1. Load and test run the model.
    a. Voxlingua107 is loaded with a test input audio clip of a Thai-speaker. The output correctly identifies the language of the speech, the figure inside the "*Tensor*" parameter being the confidence level. Of which, figure 1 shows that the model is predicting **Thai language at 98.88% confidence level**.

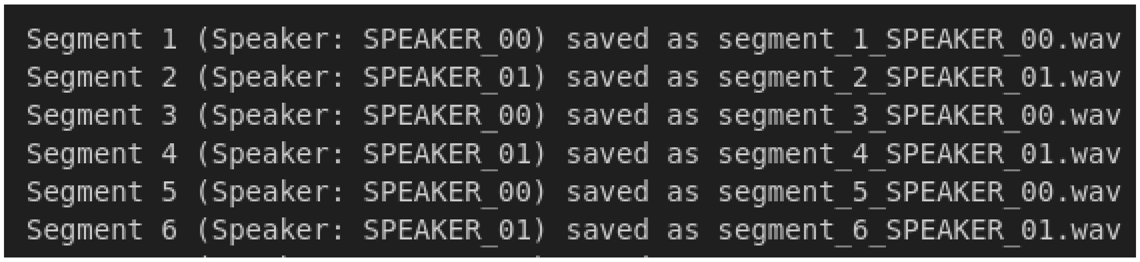


```python
import torchaudio
from speechbrain.inference.classifiers import EncoderClassifier
language_id = EncoderClassifier.from_hparams(source="speechbrain/lang-id-voxlingua107-ecapa", savedir="tmp")

signal = language_id.load_audio("speechbrain/lang-id-voxlingua107-ecapa/udhr_th.wav")
prediction =  language_id.classify_batch(signal)
print(prediction)

print(prediction[1].exp())
#  tensor([0.9850])
# The identified language ISO code is given in prediction[3]
print(prediction[3])
#  ['th: Thai']

# Alternatively, use the utterance embedding extractor:
emb =  language_id.encode_batch(signal)
print(emb.shape)
# torch.Size([1, 1, 256])
```

```
(tensor([[-2.8560e+01, -3.0180e+01, -2.0682e+01, -2.9531e+01, -2.2297e+01,
         -3.2566e+01, -3.6604e+01, -3.3338e+01, -3.2702e+01, -2.4447e+01,
         -2.4309e+01, -3.1083e+01, -2.7820e+01, -2.9947e+01, -2.5151e+01,
         -3.2326e+01, -2.4364e+01, -2.5879e+01, -2.1840e+01, -3.2606e+01,
         -2.2110e+01, -2.2802e+01, -3.1074e+01, -3.3516e+01, -3.0055e+01,
         -2.2201e+01, -2.5587e+01, -2.6460e+01, -3.2265e+01, -3.2511e+01,
         -3.0261e+01, -2.4502e+01, -2.3013e+01, -2.4539e+01, -2.8454e+01,
         -2.5266e+01, -2.8438e+01, -2.6307e+01, -2.5237e+01, -2.7755e+01,
         -2.6275e+01, -2.3663e+01, -2.5748e+01, -2.8196e+01, -2.6957e+01,
         -3.0608e+01, -2.1301e+01, -2.7913e+01, -3.3020e+01, -1.9197e+01,
         -2.9400e+01, -2.2069e+01, -1.8561e+01, -2.5597e+01, -3.3613e+01,
         -4.4922e+00, -3.1792e+01, -3.1836e+01, -2.7683e+01, -2.5398e+01,
         -3.2184e+01, -3.2448e+01, -2.8610e+01, -2.3269e+01, -2.6162e+01,
         -2.1984e+01, -2.3887e+01, -2.2585e+01, -2.7796e+01, -2.5425e+01,
         -2.6183e+01, -2.9795e+01, -2.7255e+01, -3.0361e+01, -2.4633e+01,
         -2.9093e+01, -2.7561e+01, -3.4279e+01, -2.9290e+01, -2.6230e+01,
         -3.2140e+01, -3.1887e+01, -2.8134e+01, -2.8816e+01, -2.7671e+01,
         -2.8216e+01, -2.9326e+01, -2.8759e+01, -2.1663e+01, -2.3696e+01,
         -2.4103e+01, -3.3644e+01, -2.8510e+01, -2.7394e+01, -1.1259e-02,
         -2.8855e+01, -2.1609e+01, -2.9782e+01, -3.5979e+01, -3.7176e+01,
         -3.0916e+01, -3.3084e+01, -1.9293e+01, -2.2968e+01, -3.0004e+01,
         -2.4941e+01, -2.0999e+01]]), tensor([-0.0113]), tensor([94]), ['th: Thai'])
tensor([0.9888])
['th: Thai']
torch.Size([1, 1, 256])
```

Figure 1: Loading the Voxlingua107 model and example language test.

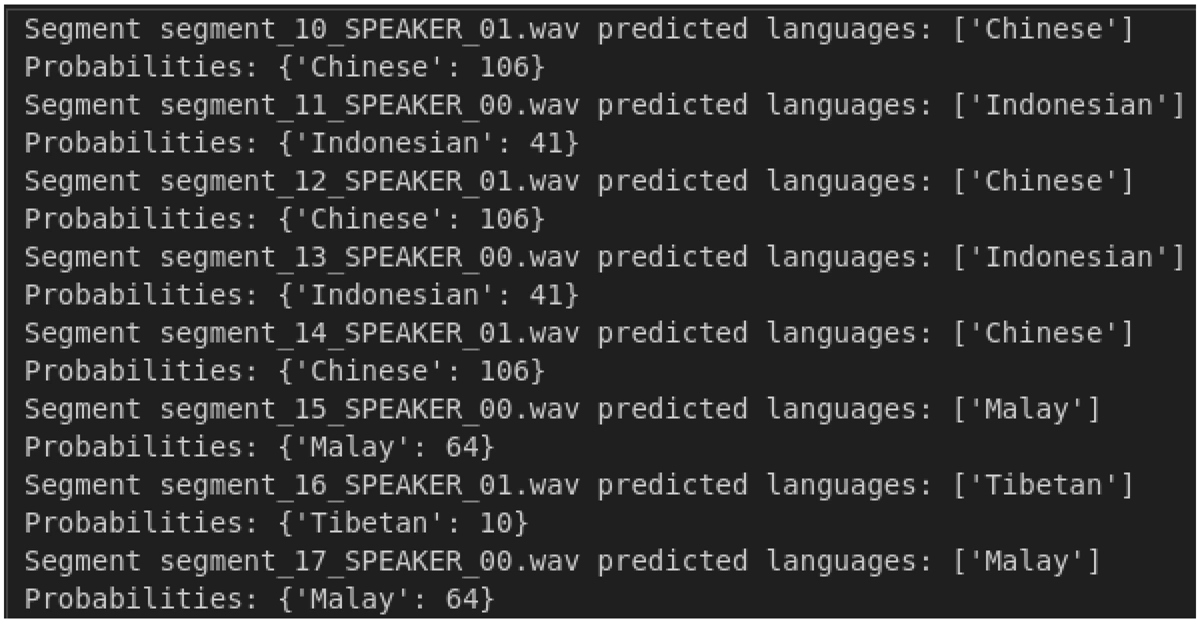
2. Dataset preparation
    a. Diarisation refers to the process of identifying and distinguishing different speakers in an audio recording.
    b. Since the goal of the assignment is to correctly distinguish languages of an input speech, I carefully selected an extract of a speech with an interpreter. This is because there will be two distinct voices and each speaker will only speak in their assigned language. Hence, I processed an Indonesian-Chinese speech with an interpreter for a clear diarisation effect.
    c. Since my focus is on the diarisation effect, I only needed a snippet of the original audio clip, which I processed into 120 seconds of test audio.
    d. The result is a segmentation of the input audio clip into its individual speaker segment .wav files as shown in figure 2. This is for me to visually tag **"SPEAKER_00" to Indonesian and "SPEAKER_01" to Chinese**.



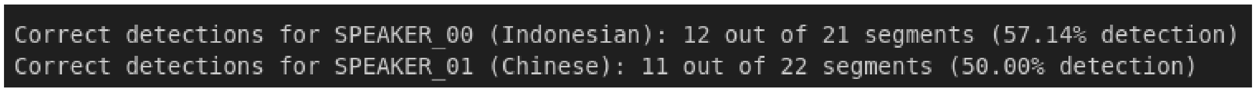
```
Segment 1 (Speaker: SPEAKER_00) saved as segment_1_SPEAKER_00.wav
Segment 2 (Speaker: SPEAKER_01) saved as segment_2_SPEAKER_01.wav
Segment 3 (Speaker: SPEAKER_00) saved as segment_3_SPEAKER_00.wav
Segment 4 (Speaker: SPEAKER_01) saved as segment_4_SPEAKER_01.wav
Segment 5 (Speaker: SPEAKER_00) saved as segment_5_SPEAKER_00.wav
Segment 6 (Speaker: SPEAKER_01) saved as segment_6_SPEAKER_01.wav
```

Figure 2: Diarisation of speakers into segments

3. Identifying Segmented Languages
    a. We iterate through every segmented file to determine correct language detection.
    b. The overlap of speech causes the diarisation to be inaccurate for some segments and hence they have poor quality or their time frames are too short to run language detection.
    c. As shown in figure 3, the correctly predicted languages have a higher probability score. "Tibetan" is wrongly predicted for Chinese while "Malay" for Indonesian, though realistically both languages are very similar in nature and tone.
    d. The result is a **57% accuracy for Indonesian and 50% accuracy for Chinese**.



```
Segment segment_10_SPEAKER_01.wav predicted languages: ['Chinese']
Probabilities: {'Chinese': 106}
Segment segment_11_SPEAKER_00.wav predicted languages: ['Indonesian']
Probabilities: {'Indonesian': 41}
Segment segment_12_SPEAKER_01.wav predicted languages: ['Chinese']
Probabilities: {'Chinese': 106}
Segment segment_13_SPEAKER_00.wav predicted languages: ['Indonesian']
Probabilities: {'Indonesian': 41}
Segment segment_14_SPEAKER_01.wav predicted languages: ['Chinese']
Probabilities: {'Chinese': 106}
Segment segment_15_SPEAKER_00.wav predicted languages: ['Malay']
Probabilities: {'Malay': 64}
Segment segment_16_SPEAKER_01.wav predicted languages: ['Tibetan']
Probabilities: {'Tibetan': 10}
Segment segment_17_SPEAKER_00.wav predicted languages: ['Malay']
Probabilities: {'Malay': 64}
```



```
Correct detections for SPEAKER_00 (Indonesian): 12 out of 21 segments (57.14% detection)
Correct detections for SPEAKER_01 (Chinese): 11 out of 22 segments (50.00% detection)
```

Figure 3: Language prediction with probability scores

4. Findings and Conclusion
    a. Model only predicts one language with a probability with each audio file, so it is difficult to fine-tune the model to sieve out languages of interest to improve the model prediction accuracy.
    b. Model does not perform well for languages that are similar in tone and nature. (Accents of the language may not be considered)
    c. Model is a comparable candidate for future benchmarking works for WordSmith.

2.1.2 STT Benchmarking of Code-Switch models

This mini-assignment was to evaluate the performance of a English-Chinese code-switch NVIDIA language model released in September 2023, STT En Zh Multilingual Code-Switched FastConformer Transducer L. The advantage of this model is that it is able to detect and transcribe both English and Mandarin Chinese language in the same sentence, with some trained dataset from South-East Asian accents.

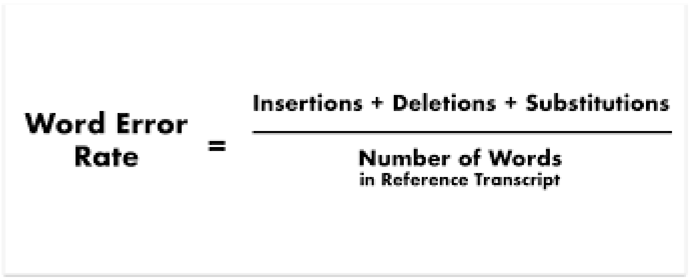
Goal: Craft a model-loading speech-to-text pipeline for local HTX datasets and evaluate the En-Zh code-switched model.

1. Data Preparation
    a. The benchmarking datasets used for evaluation provided by HTX are several hours of audio *.wav* files including police interrogations, radio podcasts and domestically-produced pseudo meetings.
    b. To improve the audio definition quality and remove silence gaps from the audio files, I implemented **Silero voice activity voice detector(VAD)** to first chunk the audio files.
    c. Since the model has limitations in processing larger volumes of audio files, I further chunked the VAD-chunked clips into smaller predefined chunks.
    d. The groundtruth of the audio file datasets are also provided in *.stm* time-stamped format, which will be used in the evaluation.

2. Audio Transcription
    a. I then cycle through and transcribed all the chunked audio files using the En-Zh model into *.txt* files.
    b. To concatenate these pieces of *.txt* files, I combined the transcripts into one whole combined transcript according to the original audio file.

3. Evaluation
    a. First, I converted the *.stm* groundtruth files into *.txt* files.
    b. We used the **Word Error Rate**[1] calculation metrics to benchmark the language models as shown in figure 4.

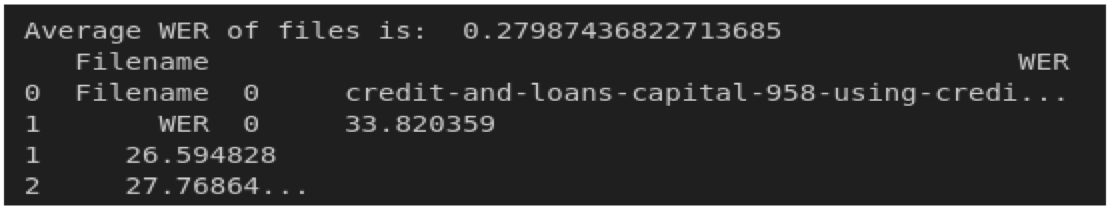
[1] WER is a common metric used to evaluate the accuracy of automatic speech recognition or transcription systems. It measures the difference between the transcribed text generated by the system and the reference/ groundtruth text.



$$\text{Word Error Rate} = \frac{\text{Insertions} + \text{Deletions} + \text{Substitutions}}{\text{Number of Words in Reference Transcript}}$$

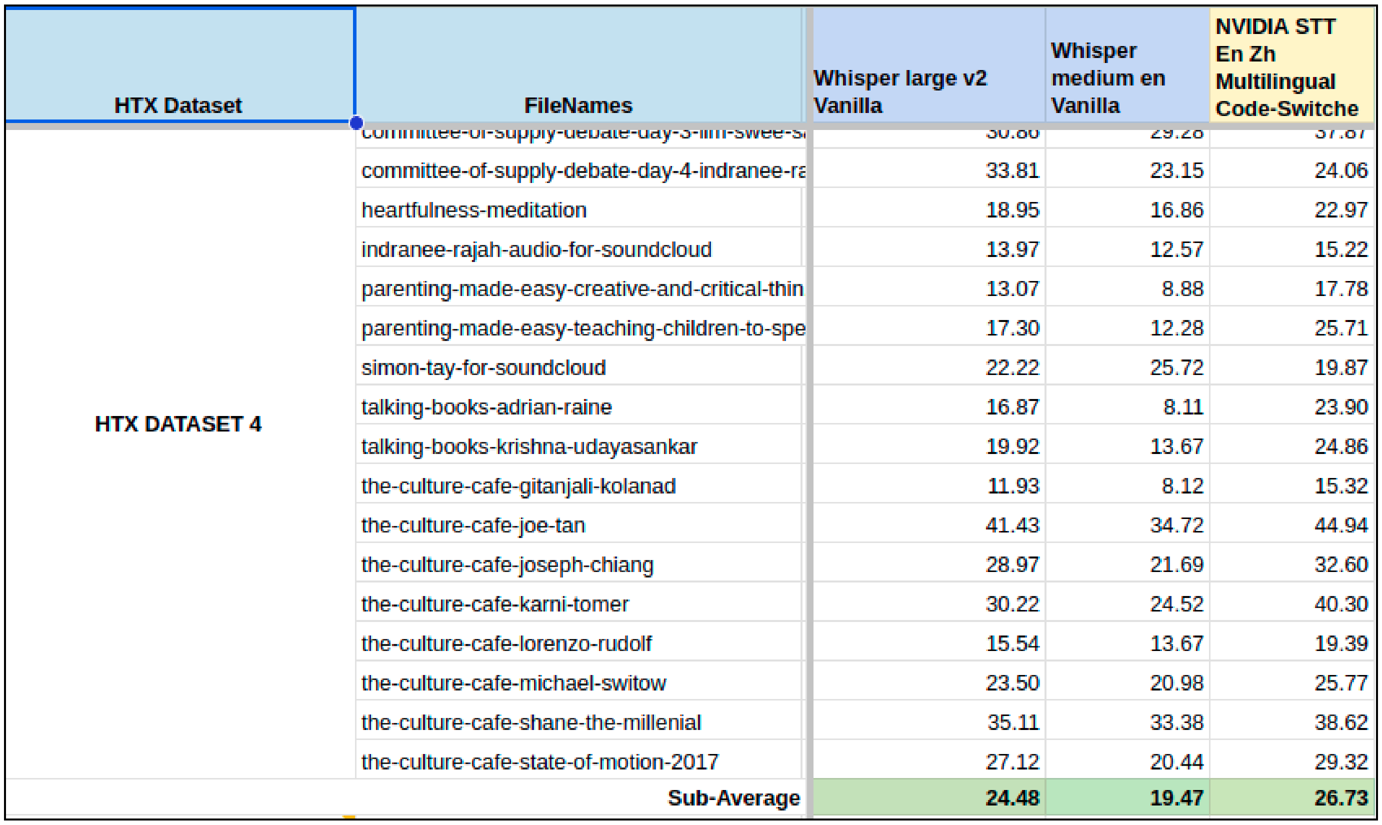
Figure 4: WER formula

c. For each transcribed dataset audio sample directory, I ran the WER script to and tabulated the results on the shared benchmarking sheets as shown in figures 5-7.



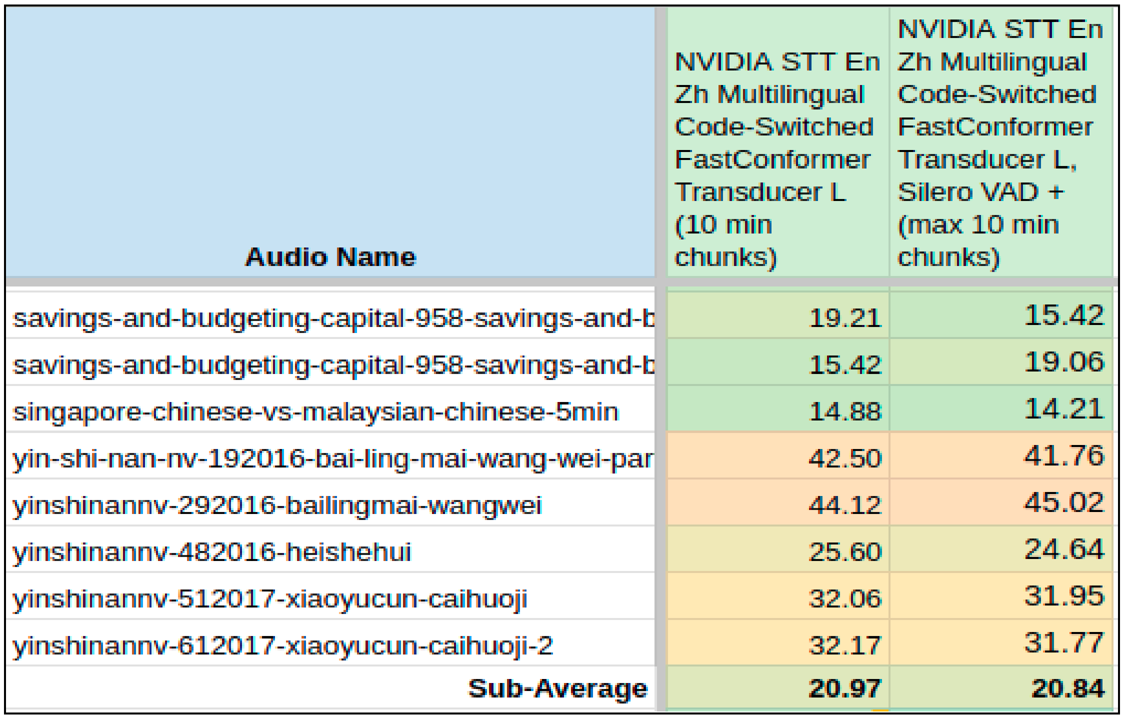
```
Average WER of files is:  0.27987436822713685
   Filename                                              WER
0  Filename  0      credit-and-loans-capital-958-using-credi...
1       WER  0      33.820359
1     26.594828
2     27.76864...
```

Figure 5: Example evaluation output of WER



HTX Dataset	FileNames	Whisper large v2 Vanilla	Whisper medium en Vanilla	NVIDIA STT En Zh Multilingual Code-Switche
HTX DATASET 4	committee-of-supply-debate-day-3-lim-swee-s	30.86	29.28	37.87
	committee-of-supply-debate-day-4-indranee-ra	33.81	23.15	24.06
	heartfulness-meditation	18.95	16.86	22.97
	indranee-rajah-audio-for-soundcloud	13.97	12.57	15.22
	parenting-made-easy-creative-and-critical-thin	13.07	8.88	17.78
	parenting-made-easy-teaching-children-to-spe	17.30	12.28	25.71
	simon-tay-for-soundcloud	22.22	25.72	19.87
	talking-books-adrian-raine	16.87	8.11	23.90
	talking-books-krishna-udayasankar	19.92	13.67	24.86
	the-culture-cafe-gitanjali-kolanad	11.93	8.12	15.32
	the-culture-cafe-joe-tan	41.43	34.72	44.94
	the-culture-cafe-joseph-chiang	28.97	21.69	32.60
	the-culture-cafe-karni-tomer	30.22	24.52	40.30
	the-culture-cafe-lorenzo-rudolf	15.54	13.67	19.39
	the-culture-cafe-michael-switow	23.50	20.98	25.77
	the-culture-cafe-shane-the-millenial	35.11	33.38	38.62
	the-culture-cafe-state-of-motion-2017	27.12	20.44	29.32
	Sub-Average	**24.48**	**19.47**	**26.73**

Figure 6: English Dataset benchmarking



Audio Name	NVIDIA STT En Zh Multilingual Code-Switched FastConformer Transducer L (10 min chunks)	NVIDIA STT En Zh Multilingual Code-Switched FastConformer Transducer L, Silero VAD + (max 10 min chunks)
savings-and-budgeting-capital-958-savings-and-b	19.21	15.42
savings-and-budgeting-capital-958-savings-and-b	15.42	19.06
singapore-chinese-vs-malaysian-chinese-5min	14.88	14.21
yin-shi-nan-nv-192016-bai-ling-mai-wang-wei-par	42.50	41.76
yinshinannv-292016-bailingmai-wangwei	44.12	45.02
yinshinannv-482016-heishehui	25.60	24.64
yinshinannv-512017-xiaoyucun-caihuoji	32.06	31.95
yinshinannv-612017-xiaoyucun-caihuoji-2	32.17	31.77
Sub-Average	**20.97**	**20.84**

Figure 7: Chinese Dataset benchmarking

4. Findings and Conclusion
    a. The En-Zh Model generally performed comparably well against other open-source language models like Whisper.
    b. The En-Zh Model only scored fairly for HTX datasets with strong accented dual-lingual speech, due to the higher level of hallucinations.
    c. Chinese language detection and transcription is rather accurate across the board, with a sub-average of WER~20 for all the Chinese datasets.

2.1.3 Azure Prompt Flow for Notes of Meetings(NoM) Summarisation

The NoM summarisation and evaluation using Prompt Flow is my main task for WordSmith. Prompt Flow empowers developers and data scientists to efficiently build, train, and deploy conversational AI models with minimal coding effort. It leverages a technique called **"prompt-based learning"** to simplify the creation of conversational AI models, where developers provide example prompts along with corresponding desired responses, and the model learns to generate appropriate responses based on these examples.

Goal: Streamline WordSmith Data Pipelining - Generate prompts for 3 types of summarisation notes; *Key points, Action points, Summary,* and evaluate the Prompt Flow.

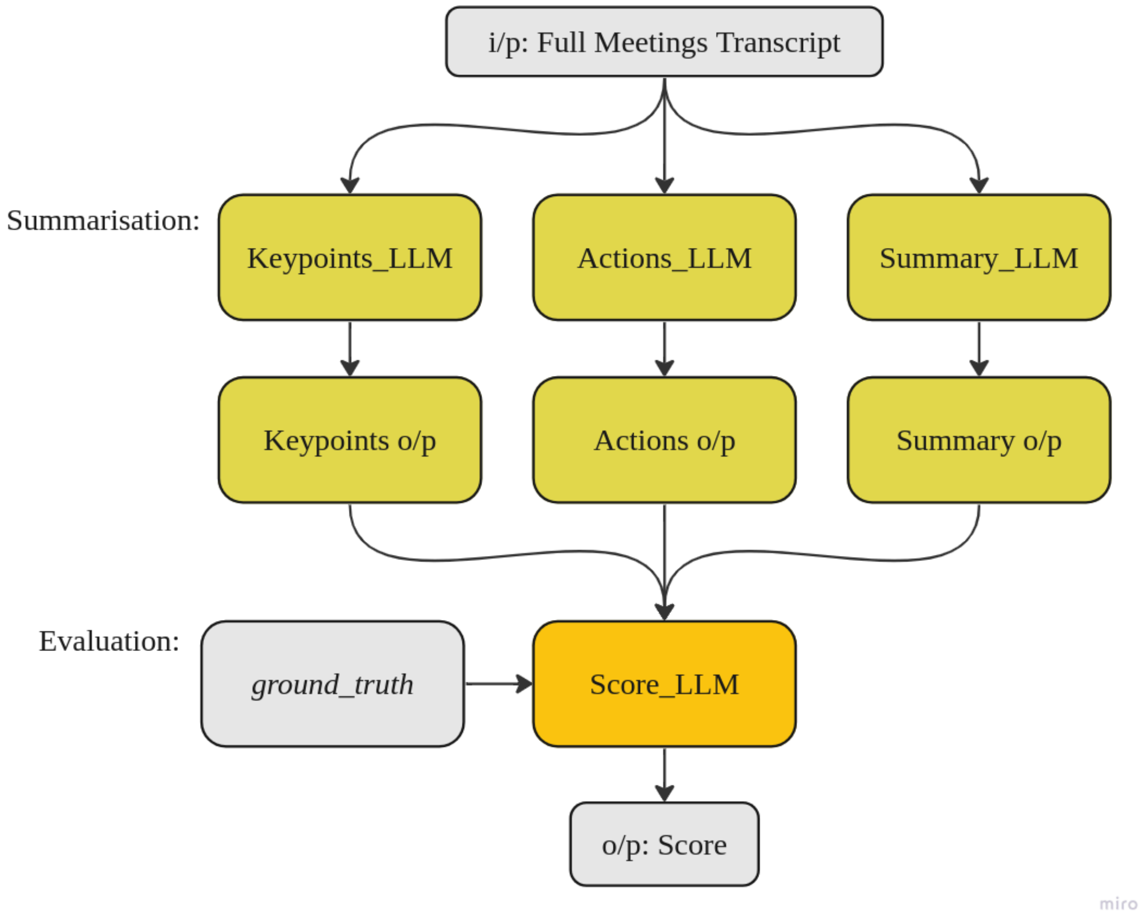


Figure 8: Overall flow of NoM Prompt Flow

<u>Prompt Flow</u>

Input dataset: HTX domestic pseudo meeting *.txt* transcriptions
Outputs: Key points summary, Action points summary , Continuous summary

LLM specifications:

a. **API = "chat"** *(Although completions are useful for summarisation tasks, chat allows the LLM to return more readable and accurate summary responses, probably due to how the LLM understands the prompts.)*
b. **LLM model = "gpt-3.5-turbo-16k"**
c. **Temperature[2] = 0** *(To ensure that the model's outputs are deterministic, meaning the results are consistent as I want the summaries for the different datasets to have a standardised format)*
d. No stop commands or maximum tokens

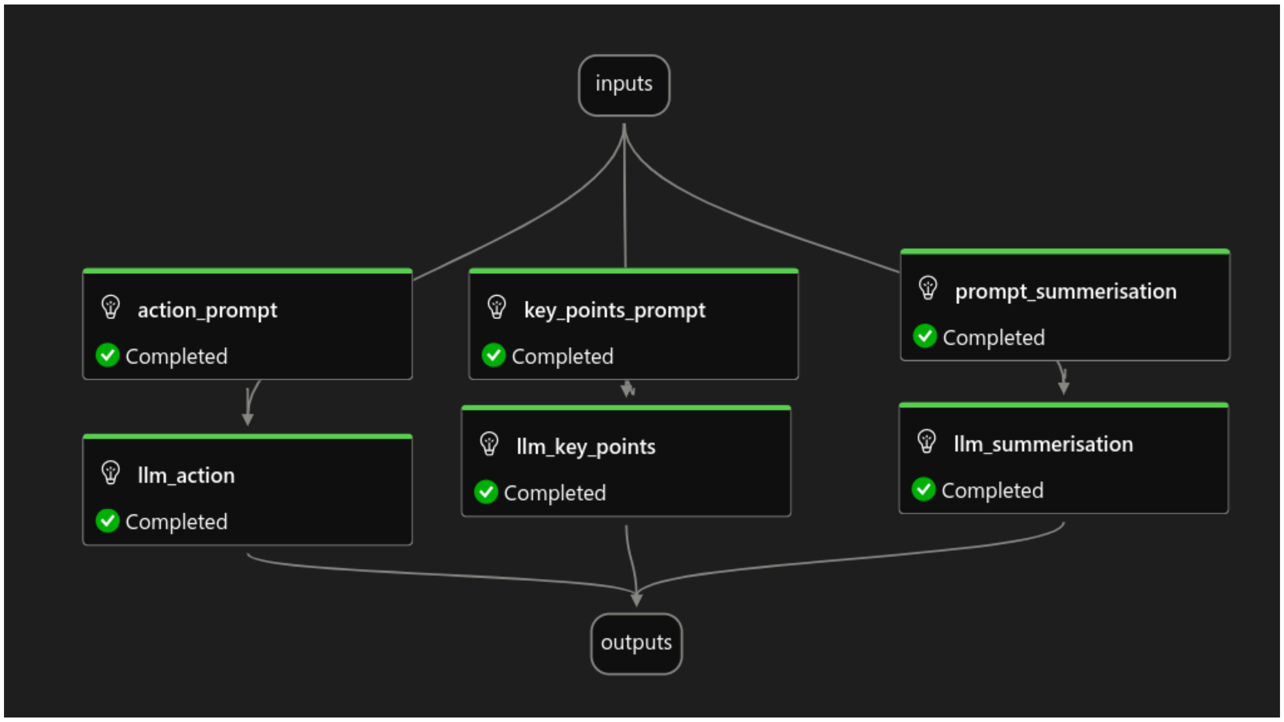


Figure 9: Full Prompt Flow sequence

[2] Controls how creative or random the responses from an LLM will be.

Action_prompt:

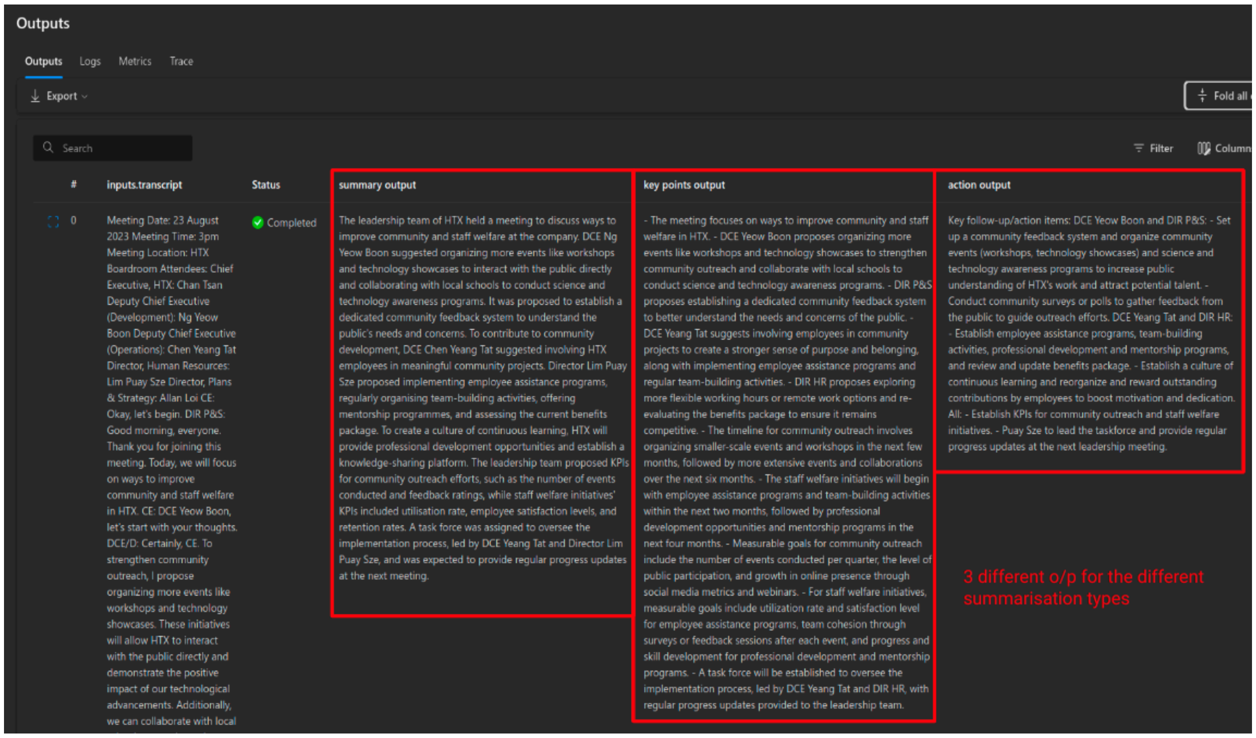
Summarise the following meeting transcript into key points using bullet format. Do not provide a verbatim summary; instead, present the information in third person, past tense, and in concise bullet points. Reflect the details accurately from the transcript, attributing each point to the respective speaker using designations (e.g., CE, DCE, Dir HR). Maintain clarity, and structure each bullet point as a statement of reasonable length. Do not add new information or details beyond the transcript.

Key_points_prompt:

The following is a meeting transcript. You are writing notes of the meeting. Summarise the transcript in third person point of view and past tense in proper paragraph form. Refer to each speaker by their short-form designation (e.g CE, DCE, Dir HR) followed by a comma. Do not make up any information.

Prompt_summerisation:

The following is a meeting transcript. Do not summarise the text. Extract the key followup or action items from the meeting and state the parties responsible for the follow up. Relay the information as accurately as possible from the transcript and do not make up any information or add additional information. Present all information in bullet point form, attributing the content to the individual responsible.



#	inputs.transcript	Status	summary output	key points output	action output
0	Meeting Date: 23 August 2023 Meeting Time: 3pm Meeting Location: HTX Boardroom Attendees: Chief Executive, HTX: Chan Tsan Deputy Chief Executive (Development): Ng Yeow Boon Deputy Chief Executive (Operations): Chen Yeang Tat Director, Human Resources: Lim Puay Sze Director, Plans & Strategy: Allan Loi CE: Okay, let's begin. DIR P&S: Good morning, everyone. Thank you for joining this meeting. Today, we will focus on ways to improve community and staff welfare in HTX. CE: DCE Yeow Boon, let's start with your thoughts. DCE/D: Certainly, CE. To strengthen community outreach, I propose organizing more events like workshops and technology showcases. These initiatives will allow HTX to interact with the public directly and demonstrate the positive impact of our technological advancements. Additionally, we can collaborate with local	✓ Completed	The leadership team of HTX held a meeting to discuss ways to improve community and staff welfare at the company. DCE Ng Yeow Boon suggested organizing more events like workshops and technology showcases to interact with the public directly and collaborating with local schools to conduct science and technology awareness programs. It was proposed to establish a dedicated community feedback system to understand the public's needs and concerns. To contribute to community development, DCE Chen Yeang Tat suggested involving HTX employees in meaningful community projects. Director Lim Puay Sze proposed implementing employee assistance programs, regularly organising team-building activities, offering mentorship programmes, and assessing the current benefits package. To create a culture of continuous learning, HTX will provide professional development opportunities and establish a knowledge-sharing platform. The leadership team proposed KPIs for community outreach efforts, such as the number of events conducted and feedback ratings, while staff welfare initiatives' KPIs included utilisation rate, employee satisfaction levels, and retention rates. A task force was assigned to oversee the implementation process, led by DCE Yeang Tat and Director Lim Puay Sze, and was expected to provide regular progress updates at the next meeting.	- The meeting focuses on ways to improve community and staff welfare in HTX. - DCE Yeow Boon proposes organizing more events like workshops and technology showcases to strengthen community outreach and collaborate with local schools to conduct science and technology awareness programs. - DIR P&S proposes establishing a dedicated community feedback system to better understand the needs and concerns of the public. - DCE Yeang Tat suggests involving employees in community projects to create a stronger sense of purpose and belonging, along with implementing employee assistance programs and regular team-building activities. - DIR HR proposes exploring more flexible working hours or remote work options and re-evaluating the benefits package to ensure it remains competitive. - The timeline for community outreach involves organizing smaller-scale events and workshops in the next few months, followed by more extensive events and collaborations over the next six months. - The staff welfare initiatives will begin with employee assistance programs and team-building activities within the next two months, followed by professional development opportunities and mentorship programs in the next four months. - Measurable goals for community outreach include the number of events conducted per quarter, the level of public participation, and growth in online presence through social media metrics and webinars. - For staff welfare initiatives, measurable goals include utilization rate and satisfaction level for employee assistance programs, team cohesion through surveys or feedback sessions after each event, and progress and skill development for professional development and mentorship programs. - A task force will be established to oversee the implementation process, led by DCE Yeang Tat and DIR HR, with regular progress updates provided to the leadership team.	Key follow-up/action items: DCE Yeow Boon and DIR P&S: - Set up a community feedback system and organize community events (workshops, technology showcases) and science and technology awareness programs to increase public understanding of HTX's work and attract potential talent. - Conduct community surveys or polls to gather feedback from the public to guide outreach efforts. DCE Yeang Tat and DIR HR: - Establish employee assistance programs, team-building activities, professional development and mentorship programs, and review and update benefits package. - Establish a culture of continuous learning and reorganize and reward outstanding contributions by employees to boost motivation and dedication. All: - Establish KPIs for community outreach and staff welfare initiatives. - Puay Sze to lead the taskforce and provide regular progress updates at the next leadership meeting.

Figure 10: Example 3 summary outputs

Evaluation Flow

For summary evaluation, I first created a *key_points.txt* file to compile ALL the groundtruth points of each dataset in one single *.txt* file. I also created the *key_points_dataset.py* script to extract the particular dataset segment of the *.txt* file for evaluation. I then used an LLM-assisted evaluation prompt to return TRUE/FALSE for each line in the specified groundtruth. Finally, I calculated and corresponded the total number of TRUE for the dataset using *calculate_score.py.*

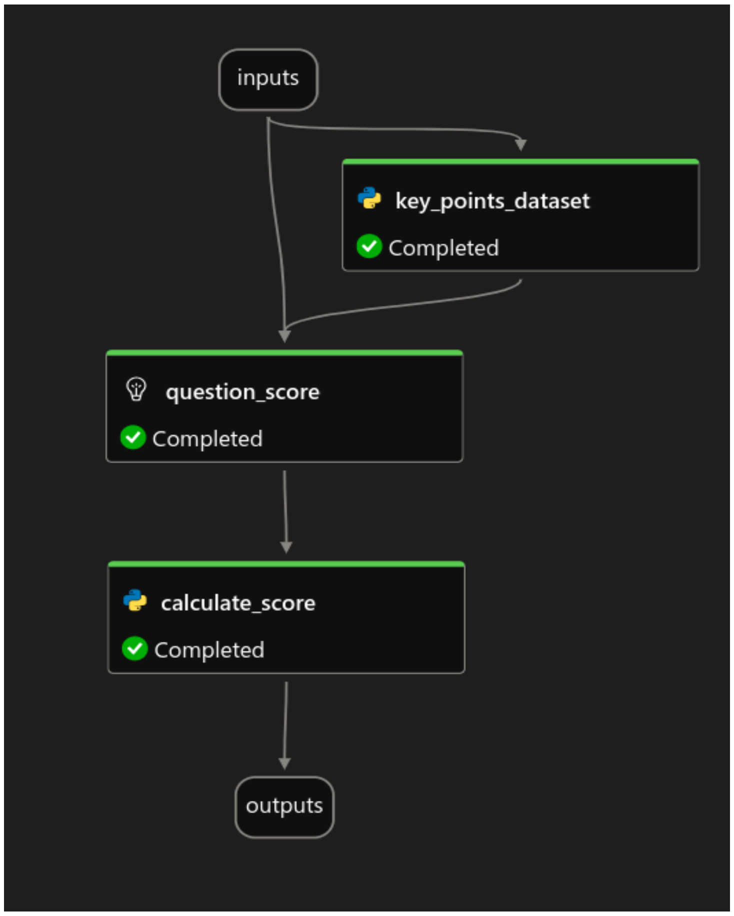


Figure 11: Full Evaluation Flow sequence

Inputs: 1. Full <*summary*> from previous Prompt Flow
2. Dataset <*transcript_name*>

Evaluation prompt: *(has the same specifications as Prompt Flow LLM)*

Question_score:

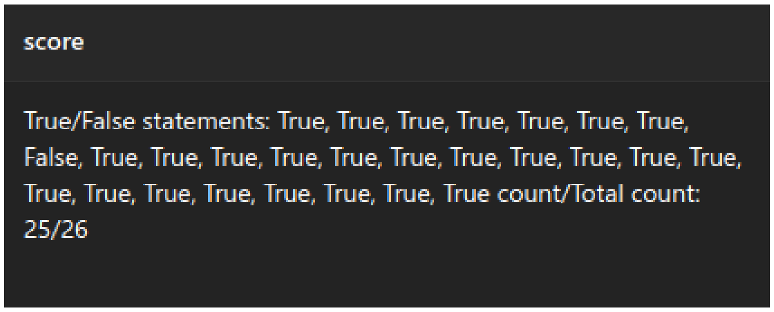
System:
You are a professional personal assistant. You will be given the definition of an evaluation metric for assessing the quality of a summary based on the given set of ground truth key points. Your job is to compute an accurate evaluation score using the provided evaluation metric.

User:
Given the key points and a summary below, carefully identify if details to the key points can be found in the summary. For each point, only return True or False if the information to the point can be found in the summary below. Do not respond with more than one word.

key_points: {{key_points}}
summary: {{summary}}

Question_score output:
"- Interns shared a presentation on proposed ideas: True\n- Implement mentorship programs: True\n- Organise joint workshops: True\n…"

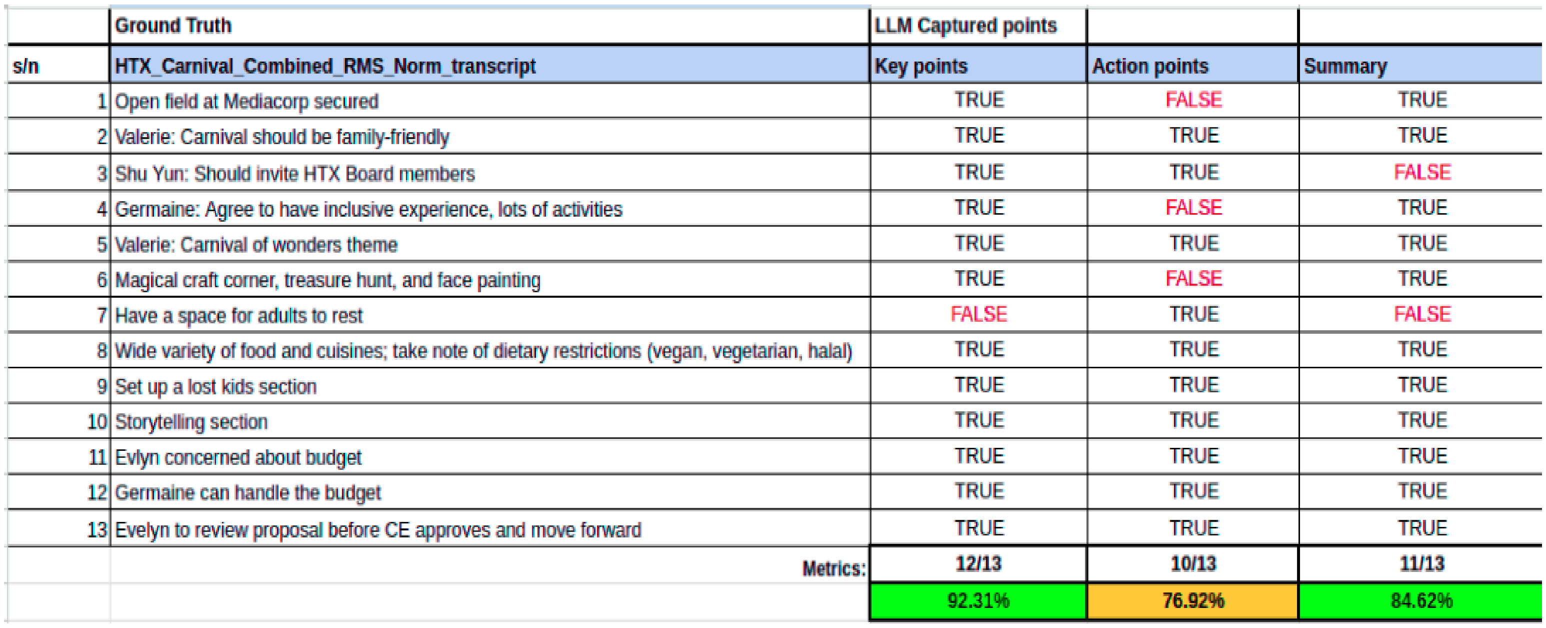
Scoring output:



```
score

True/False statements: True, True, True, True, True, True, True,
False, True, True, True, True, True, True, True, True, True, True, True,
True, True, True, True, True, True, True, True count/Total count:
25/26
```

Figure 12: Evaluation Flow scoring output

I repeated this for every summary type for each dataset and tabulated the results as follows:



	Ground Truth	LLM Captured points		
s/n	HTX_Carnival_Combined_RMS_Norm_transcript	Key points	Action points	Summary
1	Open field at Mediacorp secured	TRUE	FALSE	TRUE
2	Valerie: Carnival should be family-friendly	TRUE	TRUE	TRUE
3	Shu Yun: Should invite HTX Board members	TRUE	TRUE	FALSE
4	Germaine: Agree to have inclusive experience, lots of activities	TRUE	FALSE	TRUE
5	Valerie: Carnival of wonders theme	TRUE	TRUE	TRUE
6	Magical craft corner, treasure hunt, and face painting	TRUE	FALSE	TRUE
7	Have a space for adults to rest	FALSE	TRUE	FALSE
8	Wide variety of food and cuisines; take note of dietary restrictions (vegan, vegetarian, halal)	TRUE	TRUE	TRUE
9	Set up a lost kids section	TRUE	TRUE	TRUE
10	Storytelling section	TRUE	TRUE	TRUE
11	Evlyn concerned about budget	TRUE	TRUE	TRUE
12	Germaine can handle the budget	TRUE	TRUE	TRUE
13	Evelyn to review proposal before CE approves and move forward	TRUE	TRUE	TRUE
	Metrics:	12/13	10/13	11/13
		92.31%	76.92%	84.62%

Figure 13: Tabulated sheet for summary evaluation

<u>Findings and Conclusion</u>

- ➔ Key points and continuous summary seem to perform better than action points across the board, this is likely due to how the LLM interprets "*actionable steps*" in the initial prompt.
- ➔ Since evaluation for all 3 types of summaries are done according to the same groundtruth, evaluating the 3 types of summaries with the same ground truth may not be reliable.
- ➔ LLM might still return *FALSE* although the summary does capture that ground truth point.
- ➔ Craft 3 separate summary ground truths catered to each of the summary "formats".
- ➔ Limitation on maximum tokens the LLM can handle, transcripts over a certain length have to be first segmented, and then individually summarised and finally concatenated for each summary type for evaluation.

3 Electronic Police Centre-Chatbot (EPC)

3.1 Description of Work

The EPC-Chatbot is an additional but interesting project tasked by my subsequent supervisor, Wei Jie. It aims to integrate an automated AI chat-bot for better police report-processing and eventual document categorisation and profiling. The model automatically determines which pre-set questions have been answered by the initial report submitted by the user, and then clarifies the unanswered questions with the user in the chat.

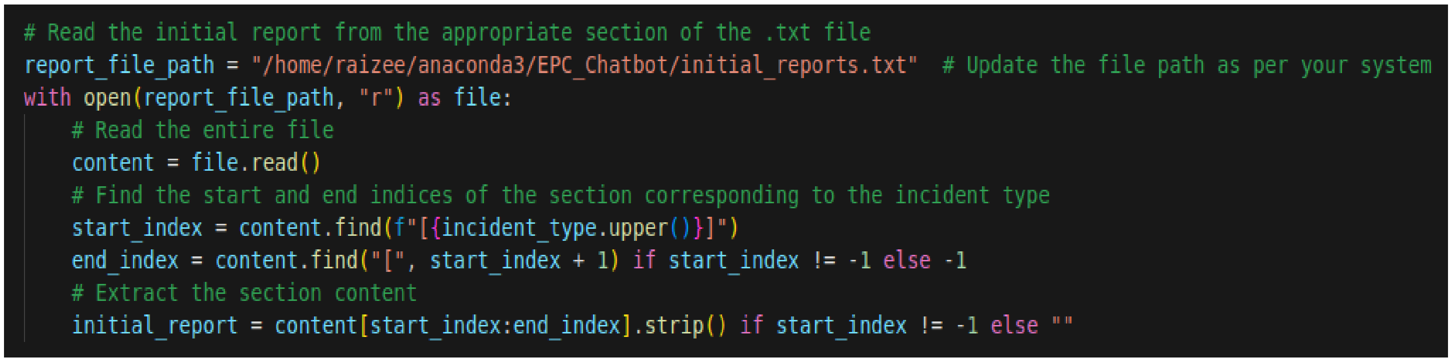
3.1.1 Question-Answering Categories

My task for this project is to come up with a testing pipeline for some initial report datasets. The test runs through all pre-set questions which are required for initial report questioning before writing it into an official police report. These pre-set questions vary for each incidence category of police reports; ***Violence, Theft, Cheating, Harassment, Lost-and-found***.

Goal: Refine the pre-set questions and engineer the prompts to ask to better aid the LLM accurately capture the information in the initial reports.

1. Consolidate pre-set questions
   First, I created an original_questions.py script to get the relevant pre-set questions for the specified incidence type. I can also easily edit these questions and add more incidence types for scalability.

2. Initial_reports.txt
   I then formatted the initial reports into a combined .txt file separated by their *[incidence_type]*, which is referenced in the main script to extract only that specific portion of the initial reports file.



```python
# Read the initial report from the appropriate section of the .txt file
report_file_path = "/home/raizee/anaconda3/EPC_Chatbot/initial_reports.txt"  # Update the file path as per your system
with open(report_file_path, "r") as file:
    # Read the entire file
    content = file.read()
    # Find the start and end indices of the section corresponding to the incident type
    start_index = content.find(f"[{incident_type.upper()}]")
    end_index = content.find("[", start_index + 1) if start_index != -1 else -1
    # Extract the section content
    initial_report = content[start_index:end_index].strip() if start_index != -1 else ""
```

Figure 14: Function to extract incidence-specific initial reports

3. LLM prompt

   Using OpenAI's api key, I used the gpt-4 model to generate TRUE/FALSE responses based on the pre-set questions. *(Similar to the WordSmith evaluation method)*

   The LLM prompt is split up into user and system parts to better specify prompts for targeted outputs.

   User_prompt:

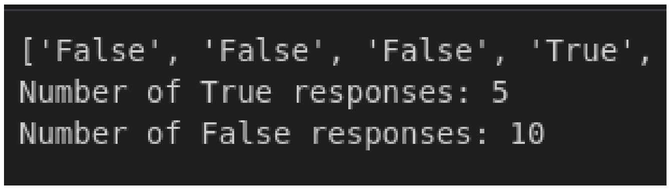
   "You are a police investigation officer. Given a question and a report below, carefully identify if details to the question can be found in the report. Return True or False if the information to the question can be found in the report below. Do not respond with more than one word.

   Question: **{question}**

   Report: **{initial_report} "**

   System_prompt

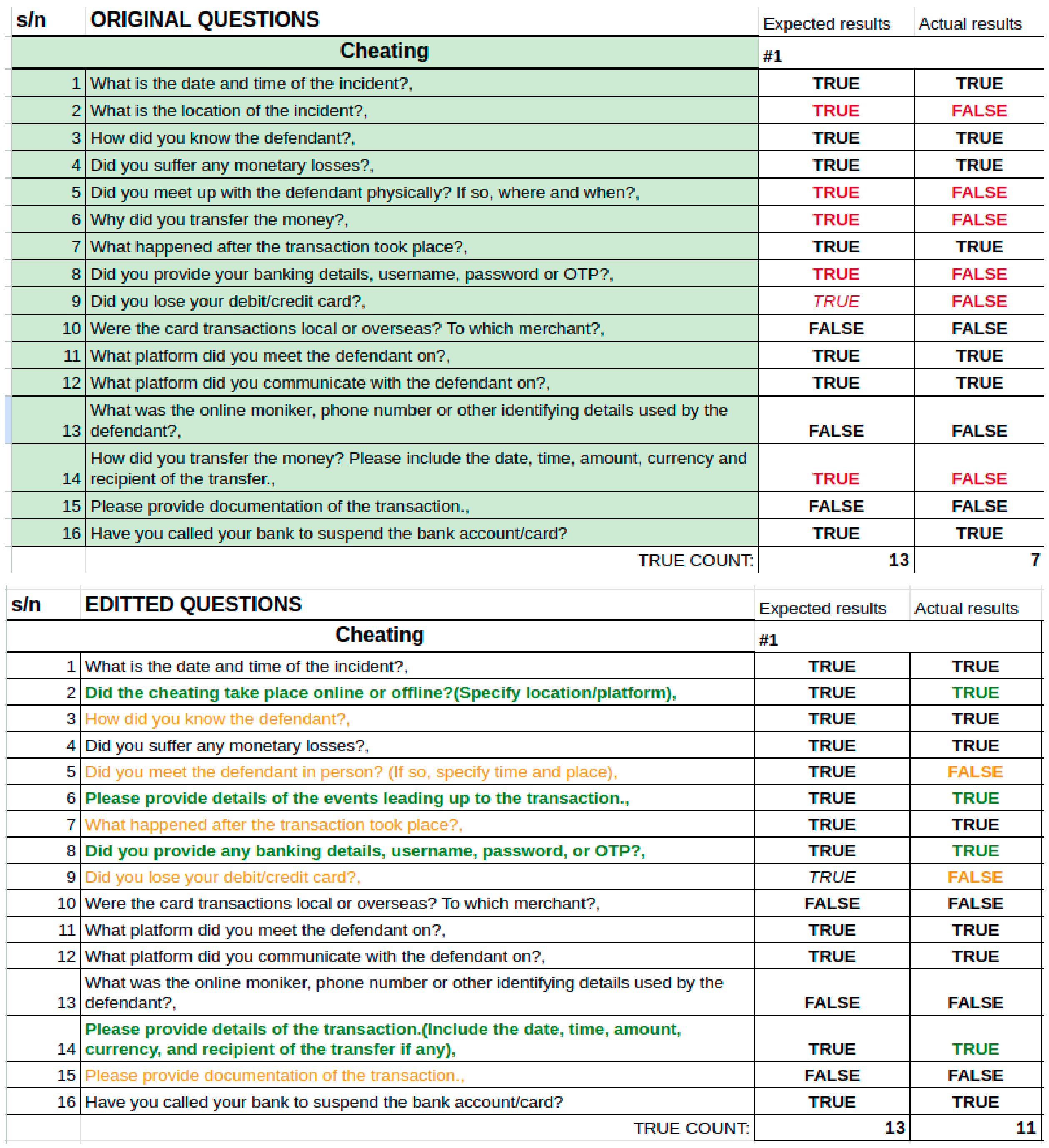
   "You are a professional personal assistant, helping your users answer queries based on the information you know. Assist the user in answering the query below to the best of your abilities.If the user asks you a question you do not know the answer to, please acknowledge that you do not have an answer. Please ensure your answers are clear and concise. Do not make up any information."



```
['False', 'False', 'False', 'True',
Number of True responses: 5
Number of False responses: 10
```

Figure 15: Example output of EPC-test script

4. <u>Tabulation of results</u>
   After recording the actual results of the original questions, I manually checked if the questions were expected to return another result from the initial reports. If the expected results and the actual results do not match (*marked in red*), I try to tweak the questions to return a more accurate result. The outcome of this approach shown an improvement in actual results as shown in figure 16.



s/n	ORIGINAL QUESTIONS	Expected results	Actual results
	Cheating	**#1**	
1	What is the date and time of the incident?,	TRUE	TRUE
2	What is the location of the incident?,	TRUE	FALSE
3	How did you know the defendant?,	TRUE	TRUE
4	Did you suffer any monetary losses?,	TRUE	TRUE
5	Did you meet up with the defendant physically? If so, where and when?,	TRUE	FALSE
6	Why did you transfer the money?,	TRUE	FALSE
7	What happened after the transaction took place?,	TRUE	TRUE
8	Did you provide your banking details, username, password or OTP?,	TRUE	FALSE
9	Did you lose your debit/credit card?,	*TRUE*	FALSE
10	Were the card transactions local or overseas? To which merchant?,	FALSE	FALSE
11	What platform did you meet the defendant on?,	TRUE	TRUE
12	What platform did you communicate with the defendant on?,	TRUE	TRUE
13	What was the online moniker, phone number or other identifying details used by the defendant?,	FALSE	FALSE
14	How did you transfer the money? Please include the date, time, amount, currency and recipient of the transfer.,	TRUE	FALSE
15	Please provide documentation of the transaction.,	FALSE	FALSE
16	Have you called your bank to suspend the bank account/card?	TRUE	TRUE
	TRUE COUNT:	13	7

s/n	EDITTED QUESTIONS	Expected results	Actual results
	Cheating	**#1**	
1	What is the date and time of the incident?,	TRUE	TRUE
2	**Did the cheating take place online or offline?(Specify location/platform),**	TRUE	TRUE
3	How did you know the defendant?,	TRUE	TRUE
4	Did you suffer any monetary losses?,	TRUE	TRUE
5	Did you meet the defendant in person? (If so, specify time and place),	TRUE	FALSE
6	**Please provide details of the events leading up to the transaction.,**	TRUE	TRUE
7	What happened after the transaction took place?,	TRUE	TRUE
8	**Did you provide any banking details, username, password, or OTP?,**	TRUE	TRUE
9	Did you lose your debit/credit card?,	*TRUE*	FALSE
10	Were the card transactions local or overseas? To which merchant?,	FALSE	FALSE
11	What platform did you meet the defendant on?,	TRUE	TRUE
12	What platform did you communicate with the defendant on?,	TRUE	TRUE
13	What was the online moniker, phone number or other identifying details used by the defendant?,	FALSE	FALSE
14	**Please provide details of the transaction.(Include the date, time, amount, currency, and recipient of the transfer if any),**	TRUE	TRUE
15	Please provide documentation of the transaction.,	FALSE	FALSE
16	Have you called your bank to suspend the bank account/card?	TRUE	TRUE
	TRUE COUNT:	13	11

Figure 16: Question tally for original vs edited questions

5. Findings and Conclusion
    - Some information in the initial report is ambiguous, causing the LLM output to be inconsistent for those questions. (*Marked in orange)*
    - There is a limit to which prompt engineering and question fabricating can improve the accuracy score. Eventually, these unanswered/ “ambiguous” questions have to be clarified by the chat-bot again.
    - LLM-based question-answering evaluation method is similar to semantics evaluation where the focus is on the meaning of the texts rather than the words itself.
    - This evaluation approach inspired the WordSmith summarisation evaluation in the previous chapter.

4 Generative Artificial Intelligence (GenAI)

4.1 Description of Work

Generative Artificial Intelligence is a subset of AI that focuses on creating new content or information. It utilises generative models, such as Generative Adversarial Networks (GANs) or autoregressive models, to generate new content, such as text, images, audio, and other media. It is used in automating the generation of reports and summaries from large datasets which can enhance decision-making processes.

The supervisors for this project are Jason and Wei Jie and they tasked me to work on Multimodal Large Language Models (LLMs) exploration. First, I had to understand the data pipeline for the GenAI app, including its Retrieval Augmented Generation (RAG) capabilities. The application is run on Docker which defines the services, their configurations, dependencies, and how they should be built and run.

<u>The following is a quick overview of the GenAI Workflow:</u>

1. User Prompt: User initiates a query.
2. API Gateway: Routes the query to appropriate services.
3. LLM Semantic Caching and Conversational Memory: Checks for cached responses and maintains conversation context.
4. Document Reader: Processes documents for relevant information.
5. Retriever: Retrieves data from the vector database.
6. Chat Model: Processes and generates a response.
7. Supporting Modules: Utilises prompt engineering, embedding models, and guardrails for optimal performance and safety.
8. Response Delivery: Sends the final response back to the user.

End Goal: Design a pipeline that loads a VLLM, reads visual document(s) and accurately comprehends visual data.
(The project is split up into multiple sub-tasks per week, amounting to around 6 weeks of work.)

4.1.1 Open-Source Vision Large Language Models(VLLMs)

The first task was to search for some open-source Vision LLM to benchmark against the GPT-4o model which we have used as the standard.

The following is the list of models to explore:

1. Microsoft/Phi-3-vision-128k-instruct
2. CogVLM2
3. XModelVLM
4. Vision Mamba[3]
5. TAT-LLM[4]

My research work was mainly centred around Microsoft's Phi-3 Vision model, newly-released in May 2024. The Phi-3-Vision-128K-Instruct is a lightweight, state-of-the-art multimodal model with a 128K token context length, built on high-quality datasets, and enhanced through fine-tuning and optimisation for precise instructions and robust safety.

4.1.2 Evaluation Metrics for VLLMs

Evaluation metrics for VLLMs are crucial for assessing their performance, accuracy, and reliability in tasks that combine visual and textual data, ensuring these models effectively understand and generate meaningful multimodal content. We considered both *semantic* and *non-semantic* evaluation techniques for a robust coverage of the VLLM's performance.

Semantic Metrics

We imported the *SimilarityFunction* class from the Hugging Face *Sentence Transformer* module, which has 4 different methods of semantic evaluation;

1. **Negative Manhattan Distance -** returns a *negative score*. (A smaller negative score indicates that the vectors are more similar, while a larger negative score indicates that they are more dissimilar.)
2. **Cosine similarity** - returns a score *between 0 and 1*. (A score of 1 indicates that the two vectors are identical, while a score of 0 indicates that they are completely dissimilar.)
3. **Negative Euclidean Distance** - returns a negative score. (A smaller negative score indicates that the vectors are more similar, while a larger negative score indicates that they are more dissimilar.)
4. **Dot Product Similarity -** similar to the cosine similarity. (A score of 1 indicates that the two vectors are identical, while a score of 0 indicates that they are completely dissimilar.)

[3] Required a large dataset(ImageNet1k) to be downloaded to pre-train the model. Dataset is too large for vm, compatibility issues on vm when running .py scripts. Weights of model architecture can be found on Hugging Face.

[4] Input data is mainly text/tabular data formatted in text/markdown. Require an intermediate step to convert tables in pdf/jpg into textual format.

Demo:

```
model = SentenceTransformer("all-MiniLM-L6-v2")
sentence1 = "I love playing soccer with my friends."
sentence2 = "I hate playing soccer with my friends."
```

Output:

```
negative manhattan ==> -9.9298
cosine ==> 0.7913
negative euclidean ==> -0.6461
dot_product ==> 0.7913
```

Even though the meaning of the sentence is the total opposite, the idea of “playing soccer with my friends” is the same, causing the cosine similarity to have a high similarity value above.
In the end, we chose to use *cosine similarity* as the semantic metrics for the task for ease of implementation.

Non-semantic Metrics

There were several candidates metrics for comparing the *edit-distance*[5] of the texts. We eventually narrowed down to using **ANLS (Average Normalised Levenshtein Similarity)** to evaluate the answers generated by the model using the **TableVQA** or **DocVQA**[6] method, depending on the dataset type.

ANLS is a metric used to evaluate the accuracy of the answers generated by machines, measuring how closely the machine-generated answers match the correct answers by considering the similarity of the text.

- ➔ Compares the LLM-generated answer against ground truth.
- ➔ *Levenshtein Distance* - calculates the number of edits (insertions, deletions, substitutions) needed to transform one text into the other. [*something like WER*]
- ➔ Normalises the distance by the length of the ground truth answer to account for different answer lengths.
- ➔ Converts the normalised distance into a **similarity score between 0 and 1**(where 1 indicates a perfect match)
- ➔ **Averages** the similarity scores across ALL evaluated answers to produce the final ANLS score.

E.g. No. of single-character edits for *"Hello World"* and "*Hello Wrld*" = **1**.
Levenshtein distance normalised by dividing it by the length of the longer string >> **1/11≈0.0909**.
ANLS=1−normalised Levenshtein distance >> **ANLS=1−0.0909≈0.9091**.
Averages the similarity scores across ALL evaluated answers to produce the final ANLS score.

[5] The number of changes (insertions, deletions, or substitutions) needed to transform one sentence into another.
[6] DocVQA aims to advance Document Analysis and Recognition by organising challenges and releasing datasets to enable machines to understand document images and answer related questions.

Combined Metrics*

Retrieval Augmented Generation Assessment (RAGAS) evaluates the quality and relevance of generated answers by comparing it against retrieved information from external sources to improve the reliability of the evaluation scores. For this task, we are using *answer_similarity* and *answer_correctness* from the **ragas.metrics** module class as shown below:

```
score_similarity = evaluate(dataset, metrics=[answer_similarity])
score_correctness = evaluate(dataset, metrics=[answer_correctness])
```

The *evaluate( )* function is also inferred from the ragas module and uses OpenAI's gpt-models as part of the evaluation method. Similarity and correctness scores between 0 and 1, where 1 indicates a perfect score.

RAGAS module:

Answer Correctness - assesses whether the generated answer accurately addresses the question based on retrieved information. Metrics used for this evaluation include:

- **Exact Match (EM)**: Checks if the generated answer matches the reference answer exactly.
- **F1 Score**: Measures the harmonic mean of precision and recall between the generated answer and the reference answer.

Answer Similarity - evaluates how similar the generated answer is to the reference answers, focusing on the semantic content rather than exact wording. Metrics used for this evaluation include:

- **Cosine Similarity**: Measures the cosine of the angle between the vector representations (e.g., TF-IDF, embeddings) of the generated and reference answers.
- **BERTScore:** Uses embeddings from pre-trained BERT models to calculate similarity scores, capturing semantic similarity more effectively than traditional methods.
- **ROUGE Scores**: Includes ROUGE-N (e.g., ROUGE-1, ROUGE-2) and ROUGE-L, which measure n-gram overlap and longest common subsequence, respectively, between the generated and reference answers.

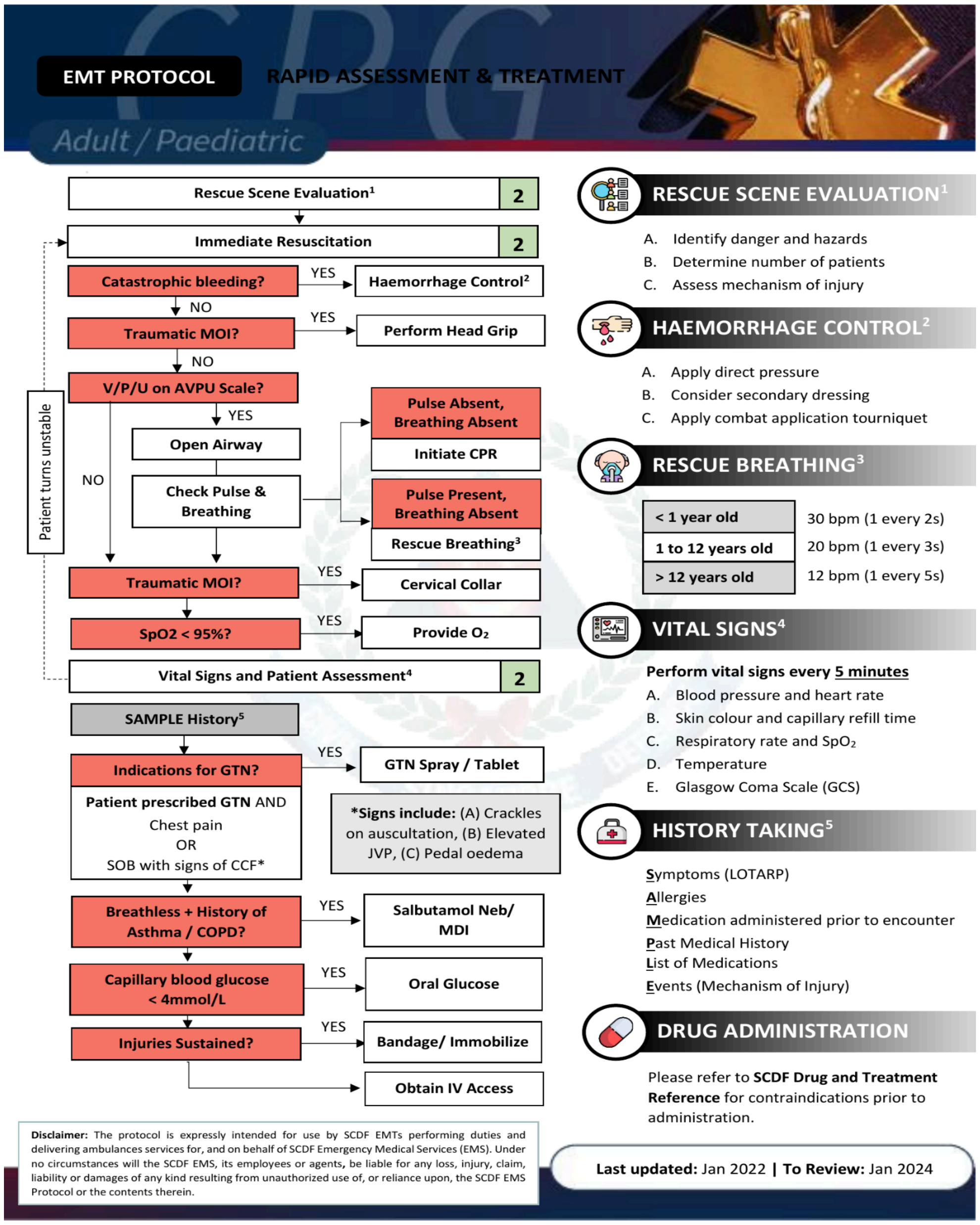
By combining these metrics, RAGAS provides a comprehensive assessment of both the correctness and similarity of generated answers, ensuring that they are accurate and semantically aligned with the reference information.

4.1.3 Dataset Preparation

We wanted to use datasets with hierarchical flowcharts and tables where information retrieval can be tricky. Each dataset has a ground truth *.json* file with ***[question_Id]: [question]: [answers]*** format headers. The ground truth question-answer pairs are generated by the *GPT-4o* benchmark model and then manually checked and edited for correctness.

SCDF_Protocol

- Single-document flowchart dictating the appropriate emergency response procedures.
- Small dataset with 11 VQA pairs. *(For quick eval.)*



RESCUE SCENE EVALUATION[1]

A. Identify danger and hazards
B. Determine number of patients
C. Assess mechanism of injury

HAEMORRHAGE CONTROL[2]

A. Apply direct pressure
B. Consider secondary dressing
C. Apply combat application tourniquet

RESCUE BREATHING[3]

< 1 year old	30 bpm (1 every 2s)
1 to 12 years old	20 bpm (1 every 3s)
> 12 years old	12 bpm (1 every 5s)

VITAL SIGNS[4]

Perform vital signs every 5 minutes

A. Blood pressure and heart rate
B. Skin colour and capillary refill time
C. Respiratory rate and SpO_2
D. Temperature
E. Glasgow Coma Scale (GCS)

HISTORY TAKING[5]

Symptoms (LOTARP)
Allergies
Medication administered prior to encounter
Past Medical History
List of Medications
Events (Mechanism of Injury)

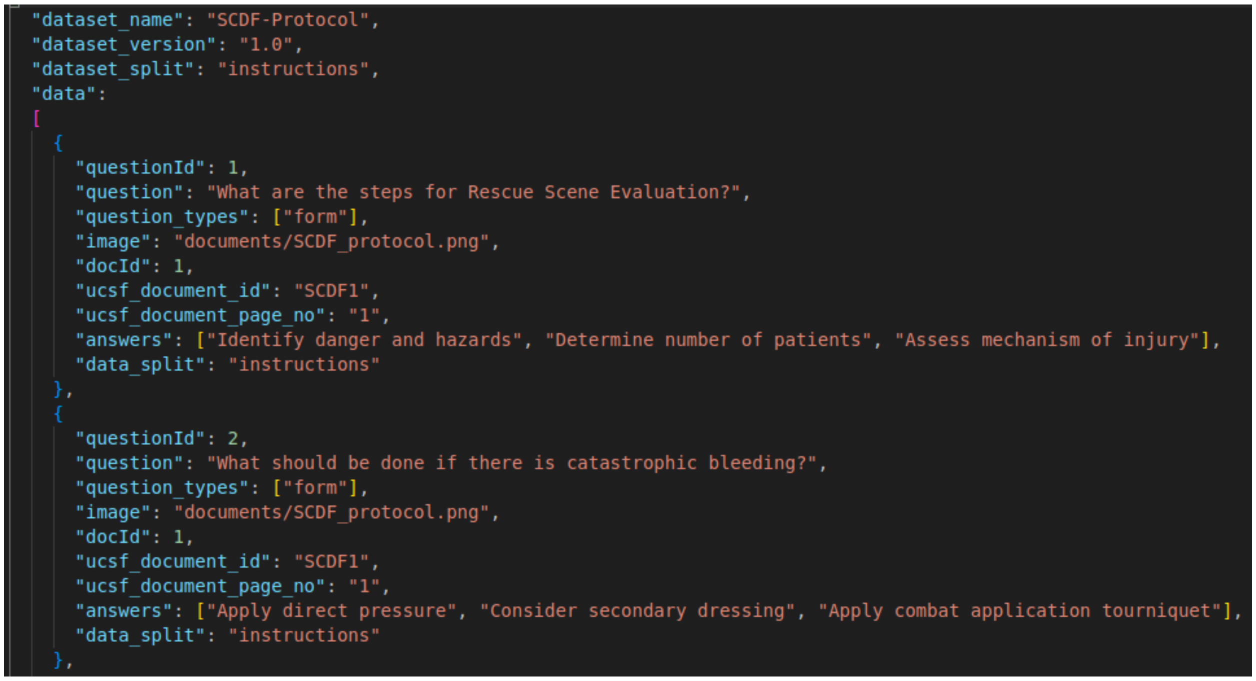
DRUG ADMINISTRATION

Please refer to **SCDF Drug and Treatment Reference** for contraindications prior to administration.

Disclaimer: The protocol is expressly intended for use by SCDF EMTs performing duties and delivering ambulances services for, and on behalf of SCDF Emergency Medical Services (EMS). Under no circumstances will the SCDF EMS, its employees or agents, be liable for any loss, injury, claim, liability or damages of any kind resulting from unauthorized use of, or reliance upon, the SCDF EMS Protocol or the contents therein.

Last updated: Jan 2022 **|** **To Review:** Jan 2024

Figure 17: SCDF_protocol.png



```
"dataset_name": "SCDF-Protocol",
"dataset_version": "1.0",
"dataset_split": "instructions",
"data":
[
  {
    "questionId": 1,
    "question": "What are the steps for Rescue Scene Evaluation?",
    "question_types": ["form"],
    "image": "documents/SCDF_protocol.png",
    "docId": 1,
    "ucsf_document_id": "SCDF1",
    "ucsf_document_page_no": "1",
    "answers": ["Identify danger and hazards", "Determine number of patients", "Assess mechanism of injury"],
    "data_split": "instructions"
  },
  {
    "questionId": 2,
    "question": "What should be done if there is catastrophic bleeding?",
    "question_types": ["form"],
    "image": "documents/SCDF_protocol.png",
    "docId": 1,
    "ucsf_document_id": "SCDF1",
    "ucsf_document_page_no": "1",
    "answers": ["Apply direct pressure", "Consider secondary dressing", "Apply combat application tourniquet"],
    "data_split": "instructions"
  },
```

Figure 18: Snippet of SCDF_protocol.json ground truth

<u>Scotland SOP</u>

- Multi-document police SOP directory with many flowcharts and tables.
- Large dataset with 80 VQA pairs.

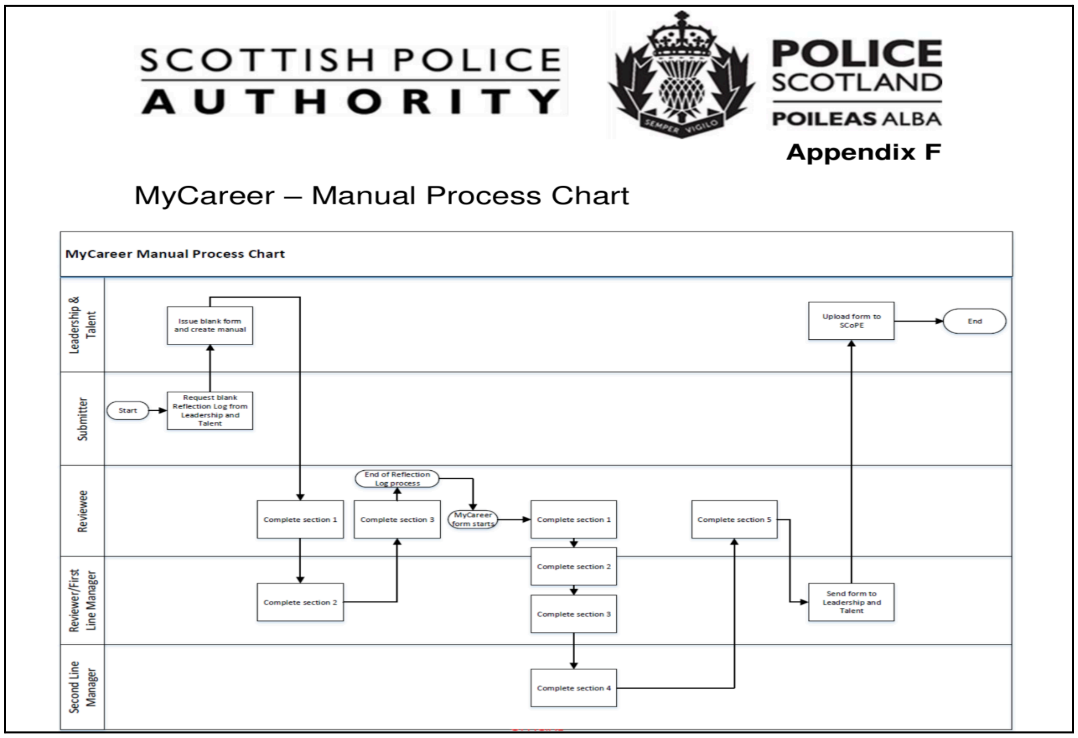
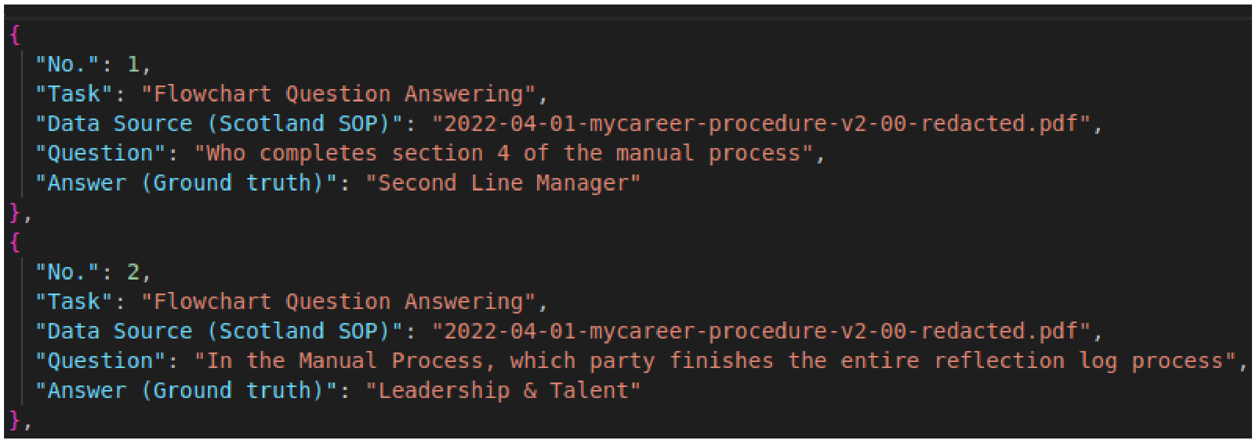


Figure 19: Abstract 2022-04-01-mycareer-procedure-v2-00-redacted.png



```
{
  "No.": 1,
  "Task": "Flowchart Question Answering",
  "Data Source (Scotland SOP)": "2022-04-01-mycareer-procedure-v2-00-redacted.pdf",
  "Question": "Who completes section 4 of the manual process",
  "Answer (Ground truth)": "Second Line Manager"
},
{
  "No.": 2,
  "Task": "Flowchart Question Answering",
  "Data Source (Scotland SOP)": "2022-04-01-mycareer-procedure-v2-00-redacted.pdf",
  "Question": "In the Manual Process, which party finishes the entire reflection log process",
  "Answer (Ground truth)": "Leadership & Talent"
},
```

Figure 20: Snippet of Scotland_SOP.json ground truth

4.1.4 Automated Evaluation Pipeline and Results

Inputs for the answer generation of the image are the ground truth questions and the image to be examined. The following is how I define the chat_completions function for the answer generation:

```
# Function to generate chat completions
def get_chat_completion(image_base64, question):
   payload = { "model": "gpt-4o",
       "messages": [{
           "role": "system",
        "content": "You are a professional
assistant that inspects images for information
and extracts them into concise summaries."},
          {"role": "user",
        "content": [{"type": "text",
           "text": f"Here is the image content.
Answer the following question fully with only
information from the image content in summarised
point form. Give the actionable steps only. Do
not use any abbreviations or
introductions.\nQuestion: {question}"},
                   {"type": "image_url",
"image_url": {"url":
f"data:image/jpeg;base64,{image_base64}"}}
]}], "max_tokens": 2000}
```

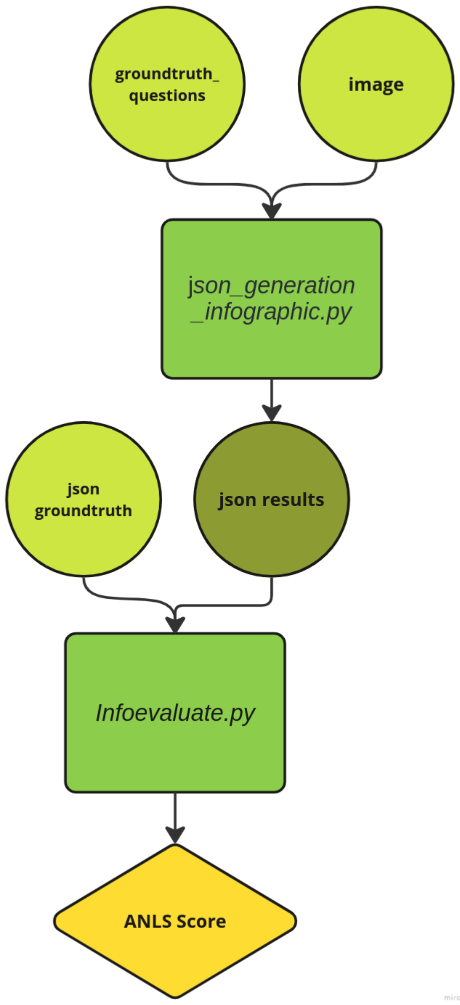


Figure 21: Full evaluation pipeline flow

The prompt for the answer generation is fixed as this, but the output of the *.json* format is slightly different depending on the evaluation method:

ANLS* Method (non-semantics)

```
{"questionId": entry['questionId'],
"question": question,
"answer": answer}
```

Cosine Similarity Method (semantics)

```
{"qa_id":entry['questionId'],
 "question": question,
 "gt": [entry['answers']],
 "pred": answer}
```

➔ The ANLS* metrics compares 2 *.json* files; ground truth and the model-generated results to generate the ANLS score.
➔ Cosine similarity directly takes in 1 *.json* file with the groundtruth and prediction answers and compares them to generate a similarity score.
➔ RAGAS evaluation method takes in the same *.json* as cosine similarity to generate correctness + similarity scores.*

The RAGAS evaluation is done be defining the helper *correctness( )* and *similarity( )* functions in *ragas_eval_helper.py*:

```
@traceable
def ragas_evaluate_similarity(dataset: Dataset, result_dir: str,
result_file: str) -> None:
   try:
       result = evaluate(dataset, metrics=[answer_similarity])

       df = result.to_pandas()
       original_data = dataset.to_pandas()
       df['question'] = original_data['question']
       df['answer'] = original_data['answer']
       df['contexts'] = original_data['contexts'].apply(lambda x:
'|'.join(x))
       df['ground_truth'] = original_data['ground_truth']

       output_file = os.path.join(result_dir, result_file)
       df = df[['question', 'answer', 'contexts', 'ground_truth',
'answer_similarity']]
       df.to_csv(output_file, index=False)
       print(f"Similarity evaluation results saved to {output_file}")
   except Exception as e:
       print(f"Error during RAGAS similarity evaluation: {e}")
       traceback.print_exc()
```

```
@traceable
def ragas_evaluate_correctness(dataset: Dataset, result_dir: str,
result_file: str) ...
```

These helper functions are then imported in the *ragas_eval.py* script where the RAG pipeline is initialised:

```
if __name__ == "__main__":
   qa_chain = init_rag()
   ragas_dataset = generate_answer(os.getenv("qa_data_in_dir"), qa_chain)

   # Evaluate similarity
   ragas_evaluate_similarity(ragas_dataset, os.getenv("ragas_result_dir"),
"similarity_results.csv")

   # Evaluate correctness
   ragas_evaluate_correctness(ragas_dataset,
os.getenv("ragas_result_dir"), "correctness_results.csv")
```

4.1.5 Benchmarking VLLMs

Hence, the results of all three evaluation methods are tabulated as shown in figure 22.

Dataset	**Dataset**	**ANLS**	**Semantic Scoring (Cosine)**	**RAGAS (Answer_correctness)**	**RAGAS (Answer_similarity)**	**Avg**
SCDF_Protocol Infographic	gpt-4o	**0.8655**	**0.8700**	**0.8509**	**0.9717**	**0.9113**
SCDF_Protocol Infographic	CogVLM	**0.5311**	**0.6794**	**0.6684**	**0.9086**	**0.7885**
SCDF_Protocol Infographic	Microsoft/Phi-3-vision-128k-instruct	**0.4098**	**0.6126**	**0.3911**	**0.8623**	**0.6267**
TableVQA Bench	gpt-4v	**-**	**0.8244**	**0.76715**	**0.947125**	**0.8571375**
TableVQA Bench	CogVLM2	**-**	**0.4789**	**0.5843**	**0.8763**	**0.7303**
Scotland SOP	gpt4o	**-**	**0.7872**	**0.6862**	**0.9461**	**0.81615**
Scotland SOP	CogVLM2	**-**	**0.5529**	**0.6218**	**0.9021**	**0.76195**
Scotland SOP	Microsoft/Phi-3-vision-128k-instruct	**-**	-	**0.5509**	**0.8784**	**0.71465**

Figure 22: VLLM Evaluation Results

Findings and Conclusion

➔ CogVLM2 seems to slightly outperform the Phi-3 model, and can be a good candidate for future Multi-modal GenAI projects.

➔ ANLS method is generally used for case-sensitive and formatting-specific contexts such as name and NRIC matching in documents, hence may not be efficient in capturing semantic meaning of image data.

➔ Semantic cosine similarity might risk false positive results where opposite meaning of the text still returns a high score.

➔ The team proceeded with RAGAS answer_correctness and answer_similarity evaluation method for future multi-modal works for a more rounded approach.

➔ Research is on other models such as Xmodel, are currently put on hold.

4.2 Text Summarisation

As a final task, my teammate Joel and I had new directions to explore and craft a summarisation evaluation pipeline for continuous and point-form summary generation. This task is similar to my experience working on WordSmith, but I wanted to expand the application scope of this task. This means covering larger datasets and common evaluation metrics. Eventually, we wanted the text summarisation feature to be integrated into the GenAI application, so as to generate quality summaries for a vast application of real-world data.

Goal: Design a summary evaluation pipeline for both continuous and point-form summarisation.

4.2.1 Summarisation Dataset Preparation

We curated a set of summarisation datasets for both long articles for continuous summaries and local HTX datasets for point-form summaries. (Dataset loading method is marked in green)

PubMed
The PubMed dataset for summarisation contains long documents, providing the body of papers as "article" and their summaries as "abstract." It has around 120,000 samples of long articles in the train set itself, hence we can randomly select 100 samples from here for summary generation.

"ccdv/pubmed-summarization": ("article", "abstract")

GovReport
The Govreport dataset is also for summarisation of long documents, with body of papers as “report” and their summaries as “summary”. It has 17,500 entries in the training set and we can impose the same random sampling method here.

"ccdv/govreport-summarization": ("report", "summary")

HTX Local Dataset
Created, with the courtesy of WordSmith, for speech-to-text summary analysis. Dataset has **9 different meeting *.txt* transcripts** with unique topics, along with predetermined ground truth points. This dataset is used specifically for the point-summary generation method.

4.2.2 Summarisation Metrics Exploration

When evaluating summary quality, it's essential to consider different metrics tailored to both point-form (bullet points) and continuous (paragraph) summaries to ensure comprehensive assessment. Generally, the main continuous summary metrics can also be performed on the point-form summaries. We selected a set of common metrics[7] that has a good coverage in both exact string-matching, semantics, correctness and relevance.

1. **Recall-Oriented Understudy for Gisting Evaluation (ROUGE)**
   ROUGE metrics compares generated summaries to reference summaries.
   There are 4 ROUGE types:
   - *ROUGE-1*: Measures unigram (single word) overlap, assessing basic word matching.
   - *ROUGE-2*: Measures bigram (two consecutive words) overlap, capturing more context.
   - *ROUGE-L*: Evaluates the longest common subsequence, reflecting the longest shared order of words.
   - *ROUGE-LSum*: Similar to ROUGE-L but splits the text using newline characters to handle multi-paragraph documents.

   For continuous summaries, *ROUGE-L* and *ROUGE-LSum* are generally the most effective metrics because they evaluate the longest common subsequence, reflecting the overall structure and coherence of the text.

   For point-form summaries, *ROUGE-1* and *ROUGE-2* are more suitable as they measure unigram and bigram overlaps, focusing on key terms and short phrases, which are typically more relevant in bullet points and list-based summaries.

2. **BERTScore**
   BERTScore is a metric for evaluating text summaries that leverages **pre-trained BERT embeddings** to measure the semantic similarity between the generated summary and the reference summary. Unlike traditional metrics that focus on exact word matches, BERTScore captures deeper contextual meaning by comparing the embeddings of words, making it a good metric for assessing the quality of summaries based on their semantic content and not just surface-level text similarity.

   - *Precision*: Measures how much of the generated summary is relevant and correct compared to the reference summary.

[7] Common evaluation metrics can be found in the Evaluate module: https://github.com/huggingface/evaluate/tree/main/metrics

- *Recall*: Measures how much of the reference summary is captured in the generated summary.
- *F1 Score*: The harmonic mean of precision and recall, providing a single metric that balances both.

3. **Bilingual Evaluation Understudy (BLEU)**
   This metric is used to evaluate the quality of text summaries by **comparing the n-grams** (contiguous sequences of n words) in the generated summary to those in the reference summary. It calculates precision for n-grams of various lengths, usually up to four, and includes a brevity penalty to handle length discrepancies.
   BLEU is effective because it considers both short and long phrase matches, making it useful for measuring how well the generated summary retains the key phrases and structure of the reference summary.

4. **Metric for Evaluation of Translation with Explicit Ordering (METEOR)**
   This is a text evaluation metric that improves upon BLEU by considering synonyms, stemming, and word order. It calculates precision and recall for unigrams and aligns them using a flexible matching approach that **accounts for synonyms and morphological variations**.
   METEOR is a good metric for summary evaluation because it **captures semantic meaning more effectively than BLEU**, offering a more comprehensive assessment of how well the generated summary reflects the content of the reference summary.

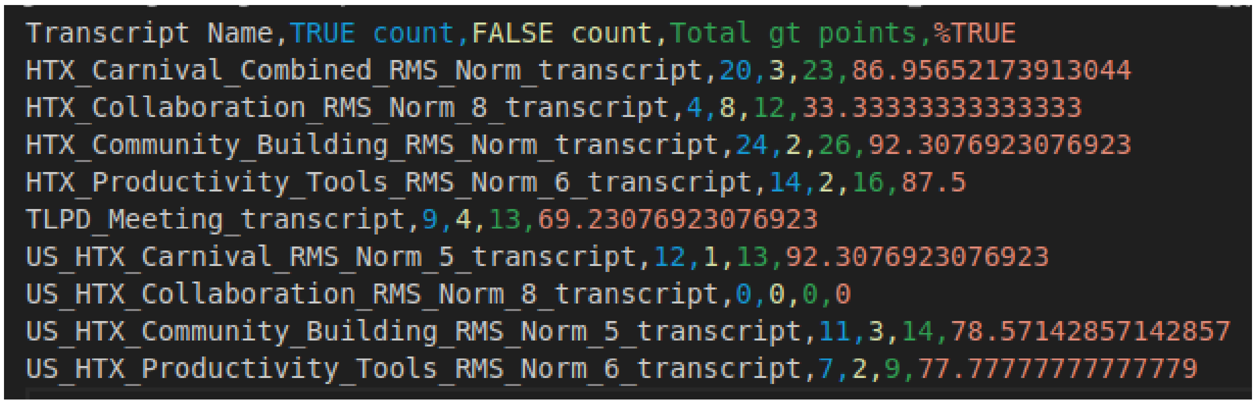
Combined Evaluation Metrics example:

Dataset	**Model**	**BERT-Precision_orig**	**BERT-Recall_orig**	**BERT-F1_orig**	**ROUGE-rouge1**	**ROUGE-rouge2**	**ROUGE-rougeL**	**ROUGE-rougeLsum**	**METEOR**	**BLEU**
pubmed-summarization	gpt-4o	0.7666571140289307	0.8710850477218628	0.8154071569442749	0.34518588791673943	0.11483578874762396	0.18447008974757617	0.32108817927775213	0.30586491953804806	0.038381371197231465

5. **<u>LLM-based evaluation</u>** *(for point-summaries)*
   This method is solely used for point-summary evaluation where we impose an LLM model to determine if a given ground truth can be found in the generated summary.

```
prompt = f"Compare the following generated summary with the ground
truth summary. For every point in the Ground Truth Summary, return
TRUE/FALSE if the information is captured in the Generated
Summary:\n\nGenerated Summary:\n{generated_summary}\n\nGround Truth
Summary:\n{ground_truth_summary}"

"model": "gpt-4o",  # Replace with the desired model
"messages": [ {
    "role": "system",
 "content": "You are a professional evaluation assistant that
assesses the quality of generated summaries." },
            {"role": "user",
          "content": prompt}
```

The above LLM-based evaluation script uses OpenAI's GPT model to assess the quality of generated summaries by comparing them to ground truth summaries. It reads summaries from *.csv* files, sends evaluation requests to the OpenAI API, and analyses the responses to determine if the information in the ground truth summaries is captured in the generated summaries. The results, including counts of *TRUE/FALSE* matches and percentage of correctly captured points, are saved in a new *.csv* file as shown in figure 23, providing an effective method to evaluate point-summary accuracy.



```
Transcript Name,TRUE count,FALSE count,Total gt points,%TRUE
HTX_Carnival_Combined_RMS_Norm_transcript,20,3,23,86.95652173913044
HTX_Collaboration_RMS_Norm_8_transcript,4,8,12,33.33333333333333
HTX_Community_Building_RMS_Norm_transcript,24,2,26,92.3076923076923
HTX_Productivity_Tools_RMS_Norm_6_transcript,14,2,16,87.5
TLPD_Meeting_transcript,9,4,13,69.23076923076923
US_HTX_Carnival_RMS_Norm_5_transcript,12,1,13,92.3076923076923
US_HTX_Collaboration_RMS_Norm_8_transcript,0,0,0,0
US_HTX_Community_Building_RMS_Norm_5_transcript,11,3,14,78.57142857142857
US_HTX_Productivity_Tools_RMS_Norm_6_transcript,7,2,9,77.77777777777779
```

Figure 23: HTX-dataset_gpt-4o_evaluation.csv results

4.2.3 Summary Evaluation Pipeline

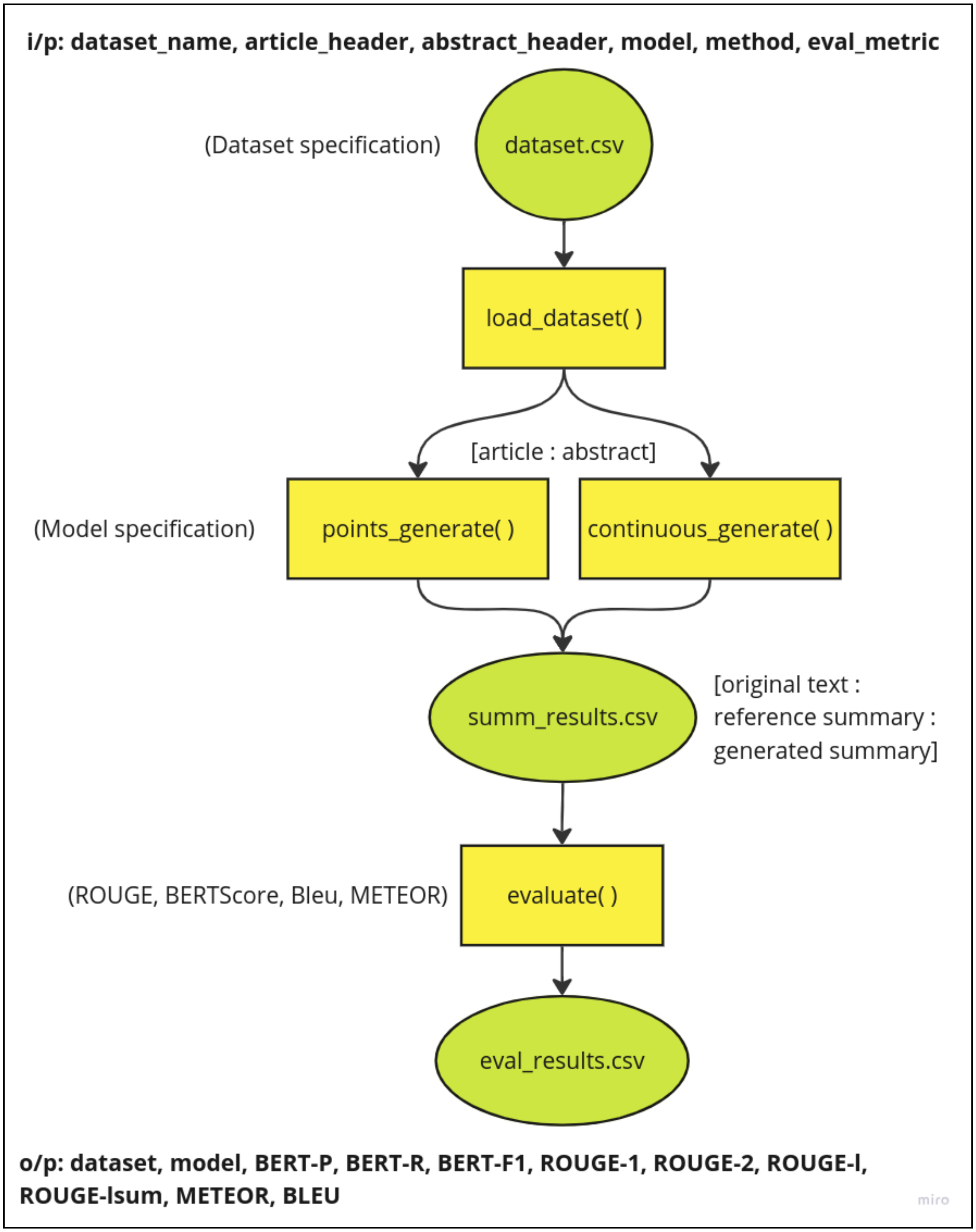

i/p: dataset_name, article_header, abstract_header, model, method, eval_metric
(Dataset specification)
dataset.csv
load_dataset( )
[article : abstract]
(Model specification)
points_generate( )
continuous_generate( )
summ_results.csv
[original text : reference summary : generated summary]
(ROUGE, BERTScore, Bleu, METEOR)
evaluate( )
eval_results.csv
o/p: dataset, model, BERT-P, BERT-R, BERT-F1, ROUGE-1, ROUGE-2, ROUGE-l, ROUGE-lsum, METEOR, BLEU
miro

Figure 24: Data Pipeline for summary evaluation

The following are python scripts that runs this evaluation pipeline which can be dockerised and pushed to the GenAI main application:

Summ_eval.py
Initialises the summary evaluation pipeline.

Summ_eval_helper.py
Helper functions:

1. *load_csv( )*: Pulls and loads dataset, converts its format into .csv with the appropriate headers.
2. *llm_evaluate( )*: LLM-specific evaluation function following metric #5
3. *summ_evaluate( )*: Combination metric evaluation function following metrics #1-4
4. *points_generate( )*: Generates point-summaries for given dataset
5. *continuous_generate( )*: Generates continuous paragraph summaries for given dataset

<u>**Findings and Conclusion**</u>

➔ A combination of evaluation metrics is essential for comprehensive assessment of the generated summary.
➔ Task-specific summary datasets should be considered for targeted outputs. (e.g. HTX local datasets)
➔ Phrasing of ground truth for point-summaries and the type of model used for evaluation and answer generation will affect the accuracy scores. Experiment with prompt engineering and data cleaning to improve this accuracy.

5 Reflections and Acknowledgements

During my time as a Data Scientist Intern at Home Team in the xData CogA team, I had the invaluable opportunity to work on pioneering projects such as WordSmith and GenAI. My primary focus was to research and develop data pipeline methods, ultimately packaging these projects into automated solutions for future implementation. This role allowed me to delve into Natural Language Processing (NLP) research, speech-to-text summarisation, generative AI techniques, and Vision Large Language Models.

One of the most significant aspects of my internship was the extensive learning and skill development I experienced. I gained proficiency in NLP techniques, particularly in data preparation and evaluation metrics for LLM-generated summaries. Additionally, I acquired knowledge on retrieval-augmented generation logic to enhance the reliability of evaluation scores. These skills have not only expanded my technical expertise but have also given me a deeper understanding of the practical applications of data science in real-world projects.
The journey was not without its challenges. Navigating technical issues with limited information available online was a recurring hurdle. However, with the experienced inputs of my supervisors, critical feedback from peers and the opportunity to attend external workshops, I was able to overcome these obstacles through a combination of trial-and-error and strategic thinking.

I am immensely grateful for the mentorship and support I received throughout my internship. I extend my heartfelt thanks to my WordSmith supervisor, Rachel Tan, for her insightful guidance, and to my GenAI supervisors, Ngoh Wei Jie and Jason Ng, for their warm supportive counsel. A special thanks to Huang ShiSheng for providing me with this incredible opportunity to explore, share, and grow.

The nature of the projects and the autonomy I was given to explore various aspects of data science made my internship thoroughly enjoyable. This experience has significantly boosted my confidence in my data science skills, particularly in leading projects, collaborating with like-minded professionals, and sharing knowledge to benefit the team. This internship has shaped me into a more proactive, dedicated data scientist who takes pride in their work and is eager to continue contributing to the field. Thank you to everyone who made this experience possible and so enriching.