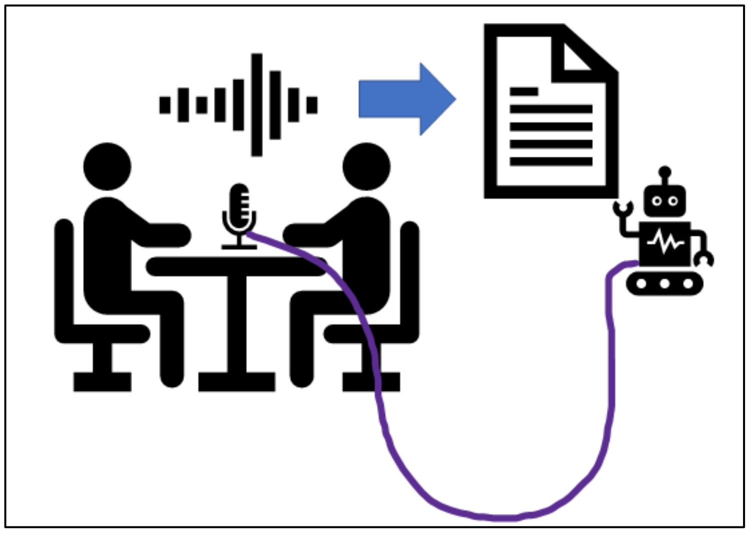
Smart Meeting Minutes

- **Due:** *11:59 pm 23/5/2022 (Monday of Week 12)*
- **Contributes** 50% of your final result
- **Group Assignment – Group of 3-4 students**

Summary

Speech recognition technology and the use of digital assistants are rapidly moving from our mobile phones to our homes, and their application in sectors such as business, banking, marketing and healthcare is rapidly becoming evident. Some of the most popular digital assistants include: Amazon's Alexa, Apple's Siri, Google's Google Assistant and Microsoft's Cortana. Smart meeting minutes has become one of the most explored and used tools as it facilitates detailed note-taking for meetings, through automatic and in-context meeting transcription, using the AI-driven speech-to-text transcription feature. At the end of the meeting, the transcript with speaker attribution can be easily retrieved via a platform. Besides, the user's notes, actions and decisions are easy to manage and their information is always available. Above all, it eliminates unnecessary overhead, allowing users and their teams to focus on their core business. In this project, you are required to develop an easy-to-use, voice enabled meeting minutes solution. This solution includes audio recognition algorithms and a platform for generating meeting minutes.



Constraints:

Real-world speech and audio recognition is complex. You are therefore not expected to build a sophisticated and advanced automatic speech recognition model. However, you should at least be able to reimplement the simple audio recognition model to recognize some respective words. Be sure to follow this tutorial on the official TensorFlow website to learn how to train a basic automatic speech recognition model to recognize ten different words. This tutorial uses a portion of the Speech Commands datasets, which contain short (one second or less) audio clips of commands, such as "down", "go", "left", "no", "right", "stop", "up" and "yes". After this tutorial, you should understand the methodology of speech recognition and know how to train an audio recognition model using deep learning (DL). You should adapt the code to work with smart meeting minutes. Note that, you should take advantage of existing libraries such as PyTorch, Tensorflow, Keras, Theano, etc.

- To begin, you will need to collect your own dataset to train the automatic speech recognition model. You are encouraged to use the Speech Commands datasets and combine them with your own collected datasets to increase the amount and variety of audio data.
- For the automatic speech recognition module, in addition to the technique presented in this tutorial, you will have to implement at least one other network or training strategies to improve the performance of the approach presented in this tutorial. You will have to justify the reason for proposing this new network or training strategies.
- As for the graphical user interface (GUI), it should at least allow users to:
  - Enter the time and date of the meeting,
  - Start a new conversation and convert the audio to text,

    - Download meeting notes
- You may refer to DeepTalk application to get a general idea on the GUI.

System requirements

- One of the key features of smart meeting minutes is to provide a service that automatically recognises speech and converts it into text to allow the user to process, store and manage their meeting updates.
- **Basic version:** The basic version of this system consists of first developing an automatic speech recognition model based on this tutorial, then train and test it on a larger dataset that you have collected. Another machine learning model, which may be formed by a new network architecture or a new training strategy, has to be implemented and compared to the aforementioned baseline technique. Next, a GUI will be available for the user to perform those actions mentioned in the previous section.
- **Extension 1**: A robust system must be able to recognize speech in real time. Therefore, the first extension of this project is to allow a user to start a new live conversation, and observe the conversion of speech to text in real time. You also need to improve the GUI to allow the user to get more information. For example, users can generate a report based on the keyword he or she enters, etc.
- **Extension 2**: After successfully implementing extension 1 which analyses live speech, you need to improve the AI model to be able to differentiate between different speakers. This involves the aggregation of more than one machine learning model and also an annotated dataset with an additional target output indicating the different speakers.

Project requirements

- Source code maintained on Git based VCS (Github/Bitbucket/GitLab/…). You must provide read-only access to the tutor/lecturer
- Running illustrative demo of a working prototype (please refer to **Marking Scheme** for details on functionality that needs to be implemented)
- Project report (8-10 pages) that includes the following sections
    - Cover Page (with team details) and a Table of Contents (TOC)
    - Introduction
    - Overall system architecture
    - Data collection and annotation,
    - Implemented machine learning techniques for automatic speech recognition,
    - Scenarios/examples to demonstrate how the system works,
    - Some critical analysis of the implementation,
    - Practical application description and
    - Summary/Conclusion.
- Presentation (video) and oral Q&A afterwards. Questions will be regarding the project as well as the unit content of the lectures (fundamentals). All group members are required to participate and contribute to the presentation, and should be able to answer questions related to the project and detail own contributions

Marking Scheme

Requirements	***Mark***
Task 1: Data collection. Implement the suggested baseline technique with the new dataset.	30
Task 2: Implement another machine learning technique, which may be a new network architecture or a new training strategy, and compare its performance with the baseline technique.	20
GUI: For a user to perform basic tasks such as 1) enter the time and date of the meeting, 2) start a new conversation and convert the audio to text, and 3) download meeting notes	10
Project Report	10
Project Presentation (Video)	10
	80
Research Component (can be done by the whole team, a sub-team, or an individual) Work on the **Extensions** and get your tutor's approval then complete it very well. You can get up to 40% for this component and your Assignment will get up to 120%	Up to 40
	120/100
You need to follow good programming practice (e.g., well-designed, well-structured codes with clear and helpful comments). Failure to do so get penalty.	**Up to -20**
You need to demonstrate the progress you make every week to your tutor. That is, if your tutor approach you and ask for the progress, you have to be able to show the tutor the progress you have made in comparison to the previous week. Failure to do so will get a penalty.	**Up to -50**

NOTE:

– Individual marks will be proportionally adjusted based on each team member's overall contribution to the project as indicated in the '*Who did what*' declaration.
– You must also provide to shlee@swinburne.edu.my read only access to your git repository within 1 week of forming teams.

Submission

- You must upload your work to Canvas by 11:59pm on 23/5/2022(Monday). Create a single zip file with your code and a working version of your system. Standard late penalties apply - 10% for each day late, more than 5 days late is 0%.
- Note that, the pre-recorded video link, can be youtube or any external file storage **(10-minute duration)** should be stated in the project report.