# TalkMaster

An AI Assistant for speedy technical support

**Report by:**

Ruchita Rajkumar Bhadre

Sammed Jitendra Kamboj

Word count: 1758 (typed, including references, individual contributions, tables, and captions) + 231 (In images) = 1989 (total)

## 1. Introduction

The Learning Space Management (LSM) at the University of Toronto (UofT) is the department that manages the central or 'shared' space at University of Toronto. Tech2U is a program offered by LSM to humanize tech support across campus. The current workflow of a TalkMaster includes answering intercom calls from classrooms, resolving the issues remotely using various apps shown in Fig 1, and dispatching staff to room if necessary. This process is time-consuming and requires experienced personnel to provide support. To address this, we leveraged the capabilities of large language models and built an AI solution to aid in decision-making.

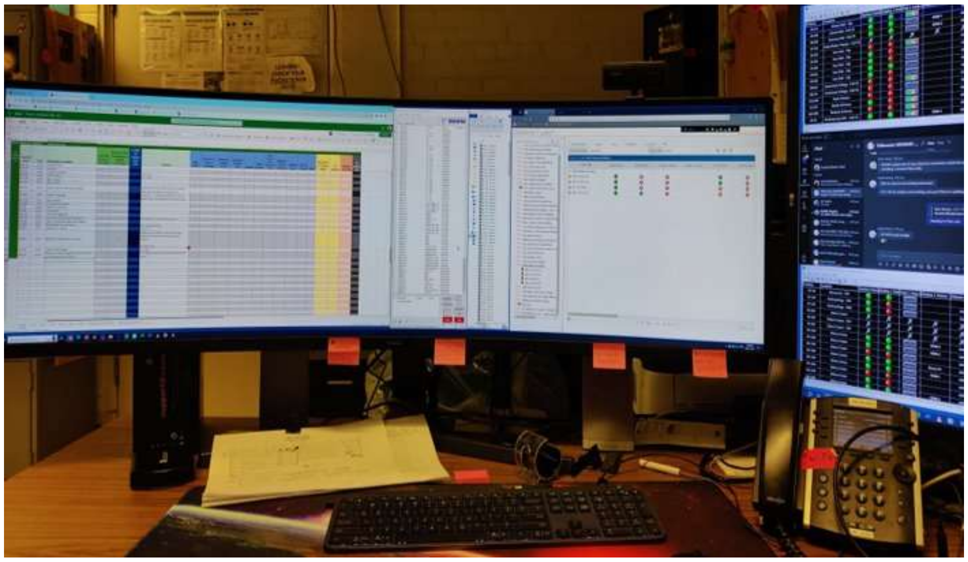


*Fig 1: TalkMaster station*

## 2. Background

Natural Language Processing (NLP) is a nuanced field and has been the most researched in the past few years. The classical NLP techniques relied on approaches such as the Bag of Words model, n-grams, and other traditional methods [1]. In the field of machine translation, there was a big step forward with the adoption of deep learning. Initially, researchers explored the use of Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, which demonstrated superior performance compared to traditional approaches [2]. Subsequently, attention mechanisms were introduced to further enhance the capabilities of RNNs and LSTMs [3]. The invention of Transformers was a game changer. This innovative architecture, designed specifically for language translation, had a profound impact on the field [4].

Transformers evolved into powerful language models, with models like GPT-2 highlighting the potential of large-scale pre-trained models [5]. This trend continued with the emergence of Large Language Models (LLMs) like GPT3 [6], Mistral 7B [15], and Llama2[14]. However, there are limitations on practical use due to hallucinations, the model's knowledge cutoff date, and privacy and ethical concerns related to API calls. In the realm of chatbots, the Retrieval Augmented Generation (RAG) approach has gained attention. RAG leverages both retrieval and generation mechanisms to enhance chatbot performance [7].

This project adopts a hybrid approach, combining classification techniques and RAG to address specific challenges. Classification helps in identifying the need to send a staff member to classroom while RAG improves response generation by incorporating information retrieved from a predefined knowledge base.

## 3. Solution Architecture

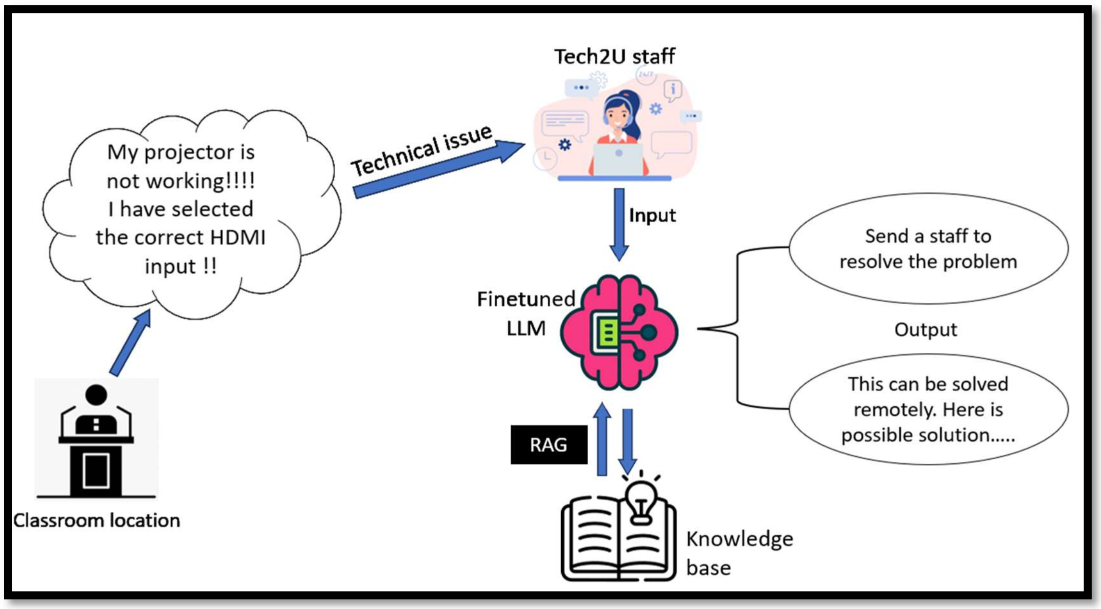

Tech2U staff
My projector is not working!!!! I have selected the correct HDMI input !!
Technical issue
Input
Finetuned LLM
Send a staff to resolve the problem
Output
This can be solved remotely. Here is possible solution.....
RAG
Classroom location
Knowledge base

*Fig 2: System Architecture*

The system has two tasks:

i. Classification (Should send staff to room or not?)
ii. RAG (If not, how to solve it remotely?)

## 4. Data Processing

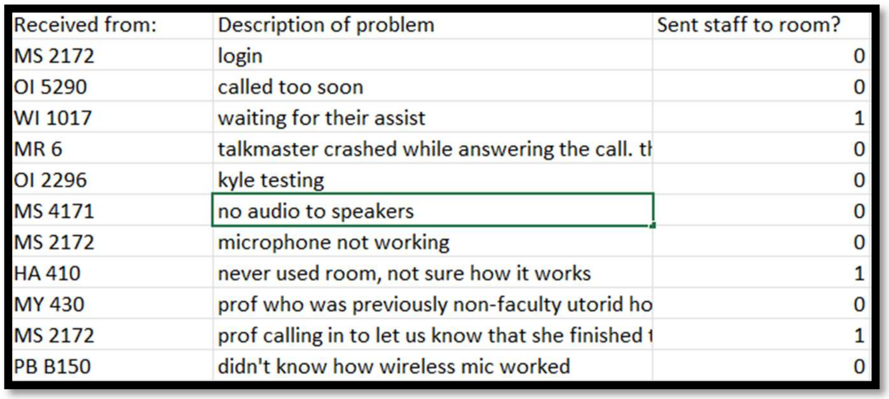
### 4.1 Classification.

The data used for classification task was the daily TalkMaster logs. It consisted of room number, time, description of problem, and various categories to identify the type of problem.

Uncleaned data had 5687 entries.

Data processing for the classification task involved:

- Data cleaning to remove false alarms, no response, accidental calls and other non-relevant information.
- Data labelling for the label- "Sent staff to room?".
- Removing punctuation and converting entire text to lowercase for better comparison.



| Received from: | Description of problem | Sent staff to room? |
|---|---|---|
| MS 2172 | login | 0 |
| OI 5290 | called too soon | 0 |
| WI 1017 | waiting for their assist | 1 |
| MR 6 | talkmaster crashed while answering the call. tł | 0 |
| OI 2296 | kyle testing | 0 |
| MS 4171 | no audio to speakers | 0 |
| MS 2172 | microphone not working | 0 |
| HA 410 | never used room, not sure how it works | 1 |
| MY 430 | prof who was previously non-faculty utorid ho | 0 |
| MS 2172 | prof calling in to let us know that she finished t | 1 |
| PB B150 | didn't know how wireless mic worked | 0 |

*Fig 3: Processed Data for classification*

After preprocessing the data-

| | |
|---|---|
| No. of entries | 4442 |
| No. of positive examples | 1326 |
| No. of negative examples | 3116 |
| Top (Description on problem) | login |

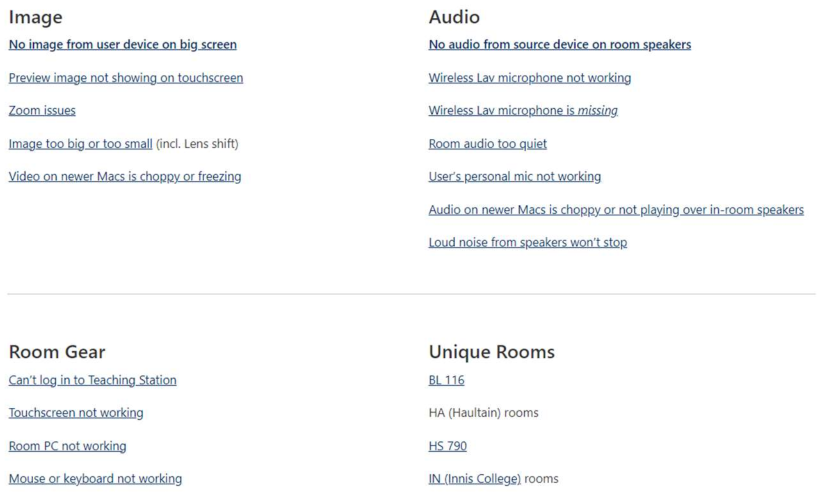
*Table 1. Description of talkmaster daily log data*

Finally for the input to model, we combined the room number and description of problem into a single prompt.

e.g. "MS 4171 microphone not working"

## 4.2 Retrieval Augmented Generation (RAG)

The second source of data that was available was talkmaster knowledgebase. The data was hosted on Microsoft SharePoint. The knowledgebase consists of troubleshooting steps for many issues, and it is continuously evolving.



**Image**

No image from user device on big screen

Preview image not showing on touchscreen

Zoom issues

Image too big or too small (incl. Lens shift)

Video on newer Macs is choppy or freezing

**Audio**

No audio from source device on room speakers

Wireless Lav microphone not working

Wireless Lav microphone is *missing*

Room audio too quiet

User's personal mic not working

Audio on newer Macs is choppy or not playing over in-room speakers

Loud noise from speakers won't stop

**Room Gear**

Can't log in to Teaching Station

Touchscreen not working

Room PC not working

Mouse or keyboard not working

**Unique Rooms**

BL 116

HA (Haultain) rooms

HS 790

IN (Innis College) rooms

*Fig 4: Talkmaster knowledgebase*

We converted all the files available (39 documents) in knowledge base into pdfs files. The data ingestion pipeline for RAG system can be illustrated in Fig 5.

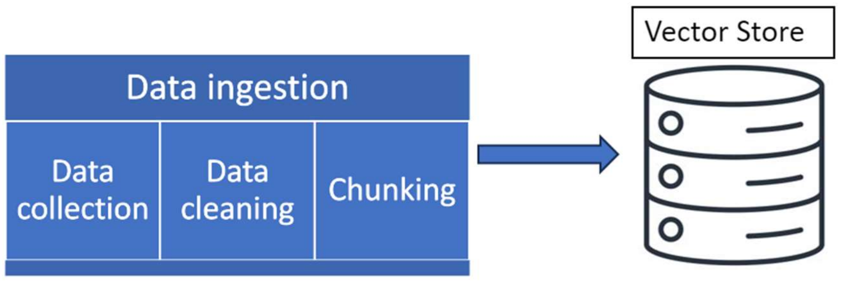


*Fig 5: Data ingestion pipeline for RAG system.*

The data collection involves collecting all the data from knowledge base. The next step involved cleaning the data which was read from pdfs. Each document consisted of some redundant features like “troubleshooting guide” as header, URLs, page numbers and dates on footer, and at the end of each document there was a table containing the editing history of the document consisting of names of individuals who worked on it. This table was systematically removed. . Following this, we standardized the text to lowercase, expunged non-ASCII characters, and partitioned the documents into nodes of chunk size of 200 with a 10% overlap. Finally, these nodes are saved in a vector database.

## 5. Software and Model

### 5.1 Classification

A tiny GPT-2 model was trained on custom classification dataset with the combined prompt (Description of problem) as input and sent staff to room as the output label from the model.

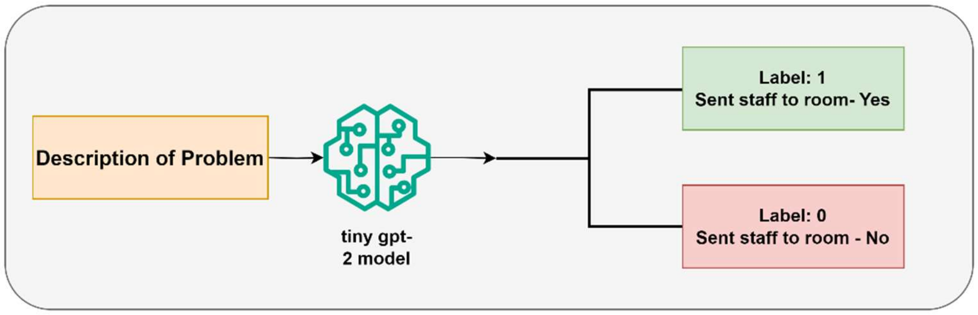


*Fig 6: Classification task*

### 5.2 RAG

The RAG system's architecture revolves around a series of key components: Vector store, Node postprocessor, LLM (Large Language Model), and response synthesizer. Initially, nodes are stored in a vector store and upon user query, relevant nodes are retrieved and post-processed. The processed nodes, along with the query, go through a response synthesizer, where LLM generates responses, forming the core framework depicted in Fig 7. The implementation of the RAG system is realized using the Python programming language, and the LLM orchestration tool utilized is 'llama-index.'

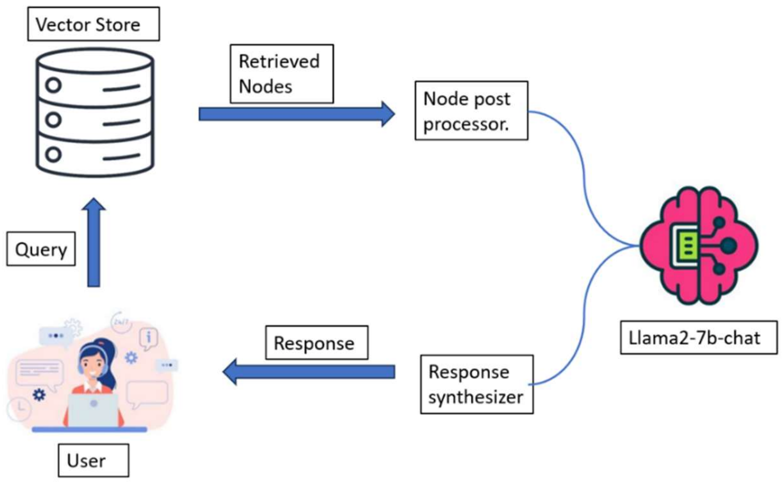


*Fig 7. RAG system architecture*

### 5.2.1 Vector Store and embedding model.

We used ChromaDB as our vector store. ChromaDB is an open-source AI based embedding database. The chosen embedding model for this project is the 'bge-base-en-v1.5' model from the BAAI [8]. This selection is based on its performance, as demonstrated on the MTEB leaderboard [9], and considerations related to system memory.

### 5.2.2 Node Postprocessor.

Post-processing of retrieved nodes is crucial for refining responses. Within the framework of the llama-index library [10], we deployed two distinct post-processing techniques: SentenceEmbeddingOptimizer and SentenceTransformerRerank.

The former enhances token usage by eliminating irrelevant sentences based on query similarity, retaining only the top relevant ones. This results in a refined set of nodes. The latter, SentenceTransformerRerank, utilizes 'cross-encoder/ms-marco-MiniLM-L-12-v2' from the 'sentence-transformer' package [11] [12], reordering nodes for maximum relevance. These yield processed nodes with removed non-relevant sentences and reordered for optimal relevance.

### 5.2.3 LLM and response synthesizer.

After the nodes are postprocessed, the query and nodes are then passed to the response synthesizer. A response synthesizer is what generates a response from an LLM, using a user query and a given set of text chunks/nodes. For the task of generating responses, our chosen LLM is the llama2-7b-chat model [14]. We employed the 'tree-summarize' option from available methods [13].

## 6. Quantitative and qualitative results

### 6.1 Classification

The tiny gpt-2 model performed well despite its size and gave training accuracy of around 85% and validation accuracy around 73% as seen in Fig 8.

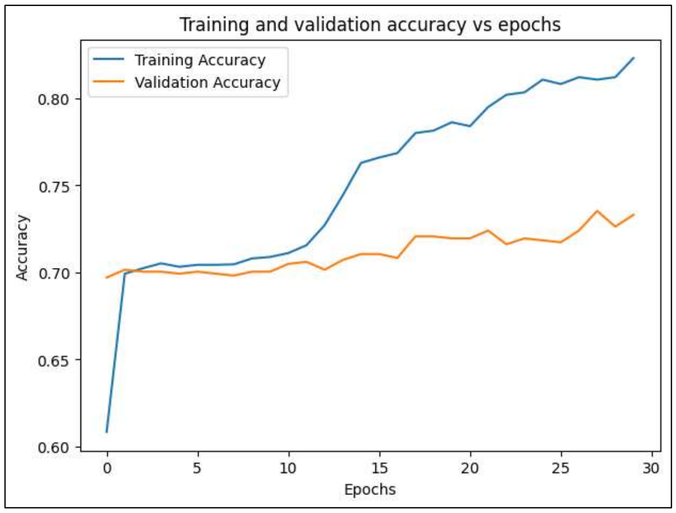


*Fig 8: Training and Validation accuracy curves*

The training loss exhibits a steady decrease over epochs, indicating that our model is effectively learning from the training data. The model performs well on unseen data as well as seen in validation loss curve shown in Fig 9.

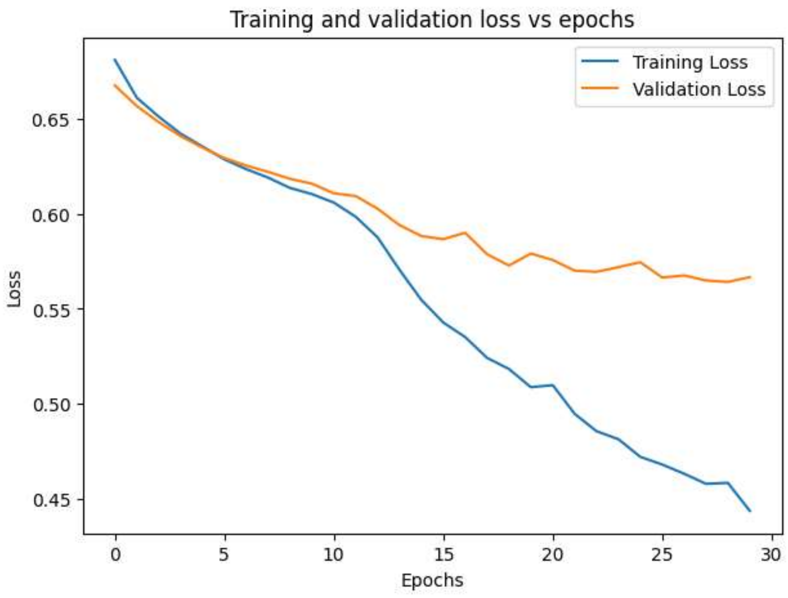
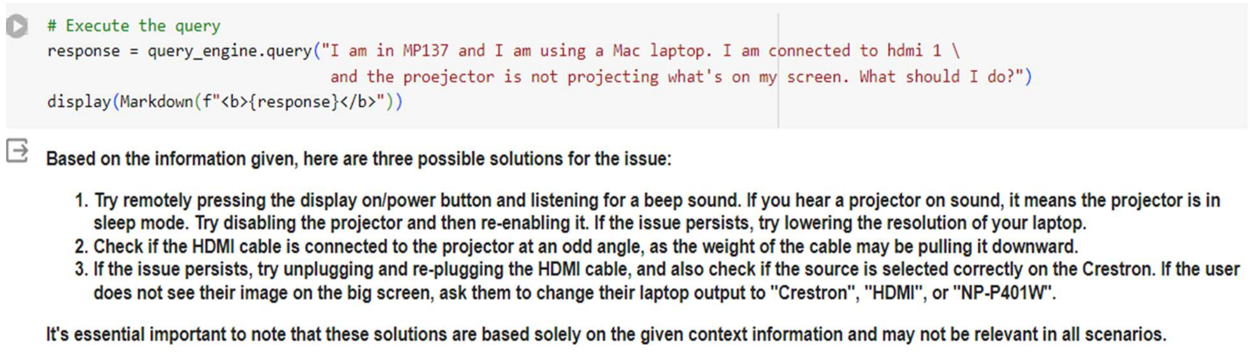


*Fig 9: Training and Validation loss curves*

### 6.2 RAG

For the quantitative evaluation of RAG system, we decided to use human judgement on relevancy. We ran a set of 20 different queries which gave classification results as zero (sending staff to room not needed) and we independently decided if the answers are relevant or not. We achieved an inter-rater agreement of 85%. Fig 10 is an example of retrieved answer from the RAG system:



```
# Execute the query
response = query_engine.query("I am in MP137 and I am using a Mac laptop. I am connected to hdmi 1 \
                              and the proejector is not projecting what's on my screen. What should I do?")
display(Markdown(f"<b>{response}</b>"))
```

**Based on the information given, here are three possible solutions for the issue:**

1. **Try remotely pressing the display on/power button and listening for a beep sound. If you hear a projector on sound, it means the projector is in sleep mode. Try disabling the projector and then re-enabling it. If the issue persists, try lowering the resolution of your laptop.**
2. **Check if the HDMI cable is connected to the projector at an odd angle, as the weight of the cable may be pulling it downward.**
3. **If the issue persists, try unplugging and re-plugging the HDMI cable, and also check if the source is selected correctly on the Crestron. If the user does not see their image on the big screen, ask them to change their laptop output to "Crestron", "HDMI", or "NP-P401W".**

**It's essential important to note that these solutions are based solely on the given context information and may not be relevant in all scenarios.**

*Fig 10. A sample output from RAG system.*

## 7. Discussion and Learnings

Classification task gave intuitive results indicating that model learned well on the training data. If there is a mention of "batteries" in the prompt the label sent staff to room is always 1, which makes sense as it is clearly a delivery that will require in-person assistance. Also, if the prompt contains the word "remotely" the output label is 0.

Our model also learned and gained knowledge about the TalkMaster specific applications such as Crestron Fusion and X-Panel which was observed from the generated output.

## 8. Reflect

Attempting to use advanced RAG to include the classification task. This can be achieved by using multiple indexes. Moreover, we can consider using more capabilities of an LLM in RAG such as LLM re-rank, query transformation, and more post-processing on nodes.

## 9. Acknowledgement

We extend our sincere appreciation to Ian Swain and Natalie Wallace for graciously granting permission to use Tech2U data, enhancing the depth of this project. A heartfelt thank you to Professor Jonathan Rose for his invaluable guidance and support. We are also grateful to the TAs Mohamed Abdelwahab and Jiading Zhu for their valuable insights and timely feedback.

## References:

[1] Jurafsky, D., & Martin, J. H. (2019). Speech and Language Processing (3rd ed.).

[2] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. arXiv preprint arXiv:1406.1078.

[3] Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In International Conference on Learning Representations (ICLR).

[4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is All You Need. In Advances in Neural Information Processing Systems (NeurIPS).

[5] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," 2019.

[6] T. B. Brown et al., "Language Models are Few-Shot Learners," 2020, doi: 10.48550/arxiv.2005.14165.

[7] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," 2021, doi: 10.48550/arxiv.2005.11401.

[8] S. Xiao, Z. Liu, P. Zhang, and N. Muennighof, "C-Pack: Packaged Resources To Advance General Chinese Embedding," 2023, doi: 10.48550/arxiv.2309.07597.

[9] N. Muennighoff, N. Tazi, L. Magne, and N. Reimers, "MTEB: Massive Text Embedding Benchmark," arXiv (Cornell University), 2023, doi: 10.48550/arxiv.2210.07316.

[10] https://docs.llamaindex.ai/en/stable/module_guides/querying/node_postprocessors/node_postprocessors.html#similaritypostprocessor

[11] N. Reimers and I. Gurevych, "The Curse of Dense Low-Dimensional Information Retrieval for Large Index Sizes," arXiv (Cornell University), 2021, doi: 10.48550/arxiv.2012.14210.

[12] https://www.sbert.net/docs/pretrained-models/ce-msmarco.html

[13] https://docs.llamaindex.ai/en/stable/module_guides/querying/response_synthesizers/root.html

[14] H. Touvron et al., "Llama 2: Open Foundation and Fine-Tuned Chat Models," 2023, doi: 10.48550/arxiv.2307.09288.

[15] A. Q. Jiang et al., "Mistral 7B," 2023, doi: 10.48550/arxiv.2310.06825.

## Individual Contributions

| | |
|---|---|
| Ruchita | • Collection of daily TalkMaster data<br>• Cleaning data to retain required details only- room number, description of problem.<br>• Labelling the 4442 entries in the data for label – sent staff to room<br>• Creating custom dataset for training with description of problem as input and sent staff to room label as output<br>• Training the tiny-gpt2 model on the custom dataset.<br>• Analyzing classification results |
| Sammed | • Converting knowledgebase into pdf files (39 documents).<br>• Cleaning and chunking files and storing them in vector database.<br>• RAG system models selection<br>• Coding the RAG system.<br>• Analyzing the RAG system results. |

## Appendix :

GitHub repository: https://github.com/ece1786-2023/TalkMaster

## Permissions:

| Team Member | Post Source code | Post video | Post final report |
|---|---|---|---|
| Ruchita | Yes | Yes | Yes |
| Sammed | Yes | Yes | Yes |

**Note: The data used for this project is private data of University of Toronto's Learning space management department. The data must not be posted publicly, so we will delete the data files from repo, thus, data will not be displayed on GitHub.**