REPORT

[Intelligent Reasoning-Based Trending and Recommendation Generation]

Group：[Group 12]

Due Date：[29th October]

EXECUTIVE SUMMARY

In the ever-evolving landscape of academia, staying connected with the latest trends and engaging with academic communities are ongoing challenges for students and faculties. This project aimed to address these issues by developing an intelligent mobile application for the National University of Singapore (NUS) community. The application, equipped with a "Hot Trending" and "Recommendation List" features powered by intelligent systems, has successfully revolutionized knowledge discovery and community engagement within the NUS academic environment.

Catalog

1. Introduction

1.1 Background

From both a student's perspective and a market standpoint, there is a significant opportunity to enhance the overall campus experience by leveraging intelligent systems technologies. Students often grapple with a broad spectrum of challenges across academic, social, and daily life aspects. They face the daunting task of keeping up with the vast amount of information and resources available in academic communities, while also dealing with the limitations of current social media platforms. These platforms are increasingly driven by generalized and data-centric recommendation systems that often inundate users with irrelevant content and connections.

Recognizing the diverse needs of students and the shortcomings of existing social media platforms, we envision the development of a revolutionary personal Campus AI assistant and a custom community-centric mobile application. These two innovations aim to address the specific pain points faced by students within the academic environment.

Our personal Campus AI assistant will offer custom data fetching and intelligent heat searching capabilities, providing students with tailored support and information retrieval. This will empower students to navigate academic challenges more effectively, streamline their daily tasks, and improve their overall campus experience.

Simultaneously, from a market perspective, there is a growing need for a more personalized and community-oriented digital platform. In the era of fast-paced digital interactions, the one-size-fits-all approach of major social media players often leaves users feeling disconnected and overwhelmed. By contrast, our community-focused mobile application will prioritize customized content and meaningful connections.

At its core, this application will introduce a "Hot Trending" feature powered by intelligent systems. This function will enable users to stay informed about trending topics, articles, and discussions across various academic domains, fostering a sense of belonging and engagement within the National University of Singapore (NUS) academic community.

1.2 Overview

We plan to develop a Campus AI assistant that provide answers and solutions for students based on their previous data, and our community-focused knowledge database. Our AI Assistant will be equipped with advanced algorithms that analyze students' previous interactions and preferences. In this way, it will provide increasingly personalized answers and solutions over time. Information previously shared by users will reside in a public database that will serve as a knowledge base, storing valuable knowledge and solutions for all users. We will use the collective wisdom of the university community to ensure that useful information is available for future use. We would also create a forum for students to share and seek information,

providing trending hashtags based on user interests. The forum will provide information for the chatbot to better interact with users, and the chatbot would gain user's interest and enable them to interact with the forum more conveniently.

The Hot Trending List feature employed recognition systems to scan various data sources such as academic journals, news articles, social media, and university forums to identify trending topics and discussions within the NUS community. This data was processed through natural language processing (NLP) algorithms to extract relevant keywords and sentiments. A reasoning system was then used to prioritize and rank these topics based on user engagement, relevance, and individual preferences. Users had access to a constantly updated list of trending subjects, making it easier for them to stay informed and engaged in their academic communities.

1.3 Aims & Objectives

At the core of this project is the development of a cutting-edge intelligent system with the capacity to not only identify but also prioritize the most relevant and trending topics, articles, and discussions within the National University of Singapore (NUS) academic community. Our vision is for this system to act as a guiding beacon, directing users towards the most compelling and pertinent academic content.

Our objectives can be summarized as follows:

User-Friendly Campus AI Assistant Platform:
We aim to create a user-friendly Campus AI Assistant platform that empowers students to easily submit queries and seek assistance. This platform also fosters a sense of community and enhances user engagement, enabling students and faculty to discover, interact with, and contribute to discussions on relevant academic topics.

Robust Question-Matching System: We strive to establish a robust question-matching system capable of automatically matching students' queries with existing answers or relevant posts and recommending solutions.

Intelligent Trends Identification: The primary aim of this project is to develop a sophisticated intelligent system capable of identifying and prioritizing trending topics, articles, and discussions within the NUS academic community. This system acts as a guiding light, leading users to the most compelling and pertinent academic content.

Meaningful Interactions: Our goal is to facilitate meaningful interactions and discussions among users, encouraging knowledge sharing, collaboration, and academic growth. We want to create an environment where knowledge sharing is not only encouraged but celebrated, and where collaboration between students and faculty members becomes second nature.

Enhanced Campus Experience: Our commitment is to enhance the overall campus experience for students by providing access to answers and sharing experiences, thereby elevating their sense of engagement and belonging.

Educational Platform: We intend to provide students with an educational platform that encourages active learning, communication, and collaboration, enabling them to acquire more knowledge and skills.

Advanced Intelligent Systems: We are committed to implementing advanced intelligent system technologies such as natural language processing and machine learning to deliver intelligent solutions to students' problems.

In essence, our project is more than just a platform; it is a digital space where the intellectual tapestry of NUS thrives and flourishes. Through our sophisticated intelligent system and the spirit of collaboration, we are dedicated to enabling a vibrant academic discourse, where ideas are shared, perspectives are enriched, and the entire academic community embarks on a journey of perpetual learning and growth.

1.3.1 Target Audience

Based on the objectives and vision outlined in the provided content, the target audience for this project primarily includes:

Students at the National University of Singapore (NUS): This project is designed to cater to undergraduate and graduate students from various academic disciplines. It aims to provide them with a platform where they can easily access relevant academic content, engage in discussions, and collaborate with peers and faculty members.

Faculty Members at NUS: The platform is also intended to serve the academic staff at NUS. It offers an avenue for professors, researchers, and other faculty members to discover trending topics, share their expertise, and engage in academic discussions. By facilitating their involvement, it enriches the academic community.

Academic Enthusiasts: Beyond the immediate NUS community, academic enthusiasts from across the globe who are interested in NUS-related topics and discussions may also find value in this platform. They can gain insights into the latest developments and trends in various academic fields.

Administrators and Educational Institutions: The project might also be of interest to university administrators and educational institutions looking to adopt or learn from the innovative strategies and technologies employed in fostering a sense of community and enhancing user engagement within a university setting.

Researchers and Innovators: Those who are involved in research or the development of intelligent systems, especially in the fields of artificial intelligence, machine learning, and natural language processing, may find inspiration or insights in the technology and methodologies employed in this project.

2. Business Value

The development of the intelligent mobile application for knowledge discovery and community engagement within the National University of Singapore (NUS) academic environment offers several significant business values:

Enhanced User Engagement: By providing a personalized platform for students and faculty to discover, interact with, and contribute to academic discussions, the application fosters a strong sense of community. This higher engagement leads to increased user retention and loyalty, which can be valuable for any business focused on user-driven content.

Competitive Advantage: Offering an intelligent system that identifies and prioritizes trending topics, articles, and discussions within the NUS academic community provides a competitive advantage. This unique feature sets the application apart from generic academic resources, making it a go-to platform for users seeking the latest and most relevant information.

Improved Knowledge Discovery: The "Hot Trending" function powered by intelligent systems enhances knowledge discovery. Users can easily find and engage with trending topics, making it a valuable tool for students and faculty looking to stay informed and relevant in their respective academic domains.

User Feedback and Iteration: Incorporating feedback mechanisms and regularly maintaining the application ensures a continual improvement process. This commitment to enhancing user experience not only keeps existing users satisfied but also attracts new users, as it demonstrates a dedication to providing a high-quality service.

3. System Features

3.1 System Intelligence

Hot Trending Recommender: This feature is a core element of your system's intelligence. It leverages user preferences and their previous interactions with the application. Here's a breakdown of this feature:

Preferences: Users have the ability to customize their preferences on their personal profile. This personalization allows the system to recommend topics and content that align with the user's interests and needs.

Previous Data: The system stores data from user interactions, such as questions asked to the chatbot or posts shared by the user. These interactions and experiences are stored in a database and are used to refine and improve the recommendations made by the system. This data-driven approach enhances the relevance of content.

3.2 Ease of Access and Scalability

Hot Trending Recommendation System: The system is built around an application (APP) that offers easy access and scalability. Here's what this entails:

Ease of Access: Users can download the application to their mobile devices. The application provides a user-friendly interface that simplifies the registration process and makes it easy for users to log in and start using the system.

Scalability: The application is designed to accommodate a growing user base. As more users join, the system should be able to handle increased load without compromising performance. Scalability is essential to ensure a seamless experience as the user community expands.

3.3 User Management

User Management: To ensure that the system can deliver personalized recommendations, user accounts are a fundamental component of the system. Here's how user management works:

Login Accounts: Users are required to create login accounts to access and use the system. The front-end of the application, built using Flutter, likely includes a login page where users can register and log in.

User management is crucial for tailoring recommendations to each user's preferences and ensuring the security and privacy of user data.

4. System Design

4.1 System Architectures

We combined the flutter and flask in python as the main system infrastructure. Flask, as backend of system , include the user login, hot research result,cusomized recommendation information function. Flutter, as frontend of system, is responsible for the interaction between users and app and represent the result of all functions.

Figure 1. Overall Architecture Diagram

The following Entity Relationship Diagram shows how the data presented in each step.

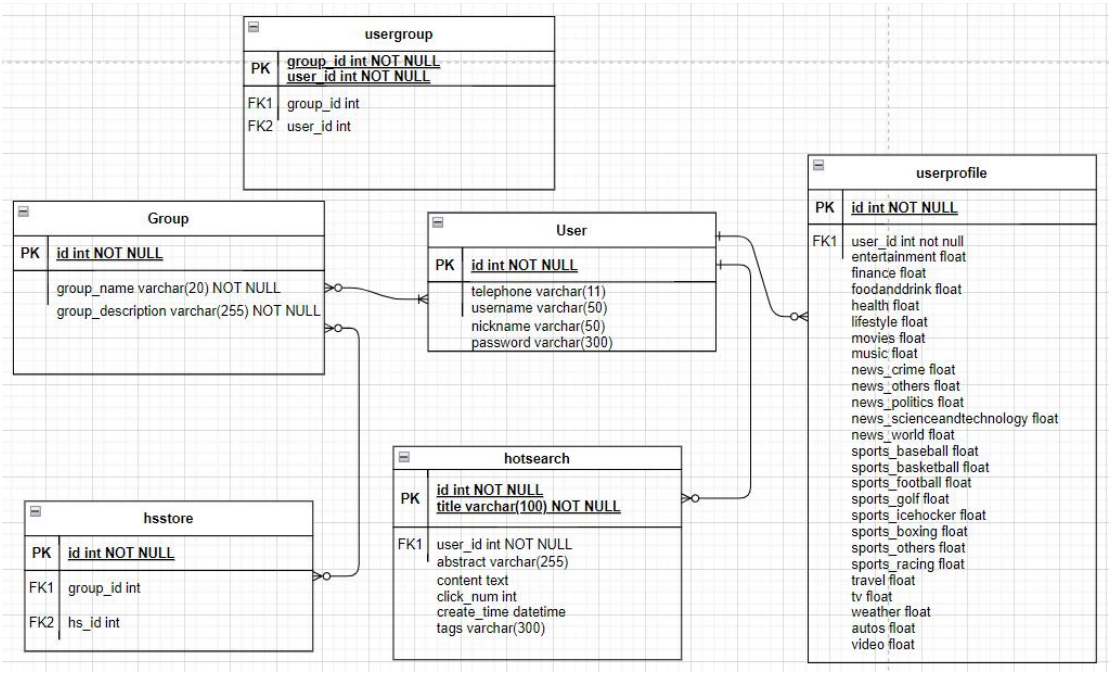

Figure 2. Entity Relationship Diagram

For Large language model, we are currently using openAI's API, specifically GPT3.5 turbo-0613. We can easily replace it with other available models, but considering cost and effects, we found it the most suitable. This also allow users to implement their own API keys so that our server cost can be minimized.

For safety purposes, we specifically designed the system to implement API calls purely from the local APP, and request details are not going to be uploaded to the server.

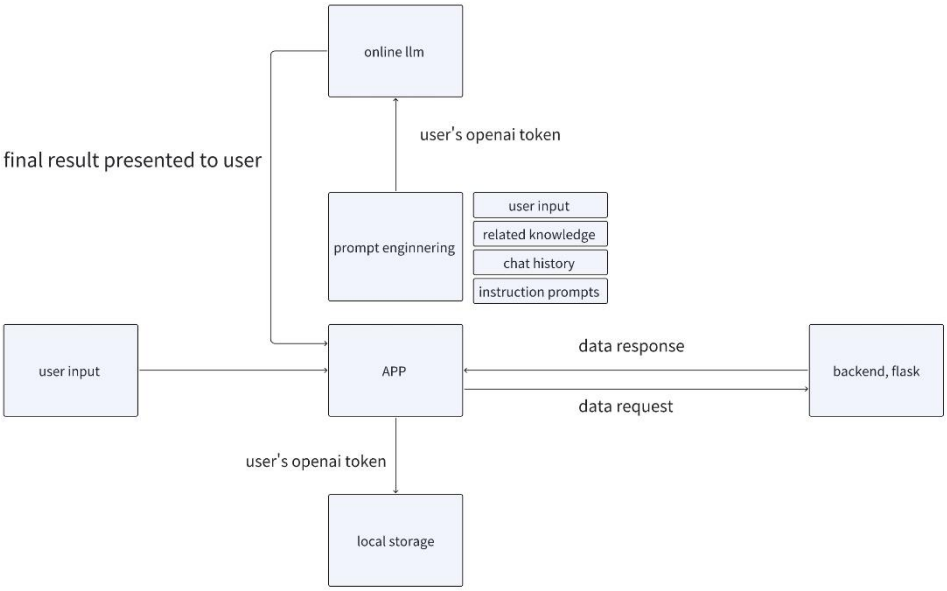

Figure 3. Demonstration of the flow of question answering

From a Product perspective our system can be separate into these following parts:
post sending:
it is easily accessed for user to send messages, and interesting stories or information.

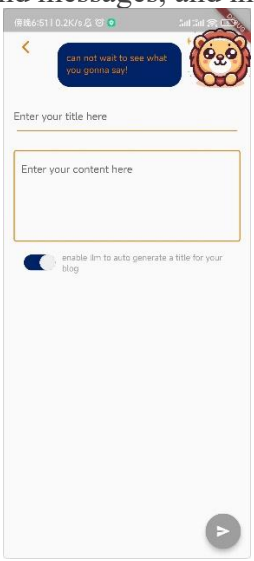

Figure 4. Post Page

Trending and Recommendation list:
It present users previous blogs that they might be interested based on different way of predictions.

Figure 5. Trending Page

AI chat bot:

It consolidates information retrieved from previous post of users in groups and present user a understandable and suitable answer based on information retrieved. It act like a semi-public database for group uses.

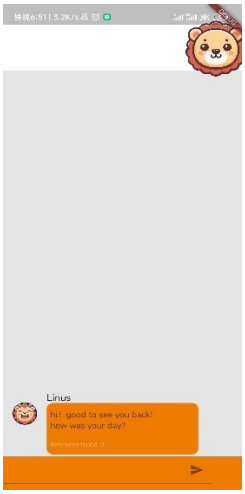

Figure 6. Chatbot Page

Group system:

It classify users and select information related to them. Meanwhile, it restricts databases user chatbots can access. This increase the possibility of finding relative data and it prevent potential privacy risks.

Figure 7. User Profile Page

The idea if that our user can found this product useful in question answering or blog posting, and the post they send is going to be benefit others and make our product more suitable for them. In the future we are going to create more modules to aid the formation of this positive cycle.

4.2 System Implementation

4.2.1 Recommendation List, Trending List and Latest List

The App's recommendation list function is powered by an intelligent recommendation system which, based on the user's preference, can generate a list of titles linking to the original article pages, while the trending list is created objectively with regression algorithm depending on whether the topics are within the hottest in real-time. See Figure 8.

After downloading the App and getting registered, a user can join different groups where users can post blogs and discuss topics of common interests, such as hiking, basketball, AI, etc. All user activities will be stored in the database, from which the blogs posted in the groups the user joined will be extracted and fed as text input into a fine-tuned BERT classifier. Each blog can be classified as 25 possible labels, so the classifier with a softmax function will output a 25-dimensional vector corresponding to the probabilities of 25 classes. Then the vector will be passed to a task-specific scoring system to match with the user feature vector which is also a 25-dimensional vector defined dynamically by the user's activities like clicks and views. The scoring system outputs the matching scores. Finally, a recommendation list will be generated taking blogs with top matching scores.

The database stores the posting time and the number of clicks of blogs posted within a time frame, which are passed to a regression algorithm to generate a global trending list. The trending list can facilitate users to be informed of the trending topics, articles, and discussions within the NUS academic community.

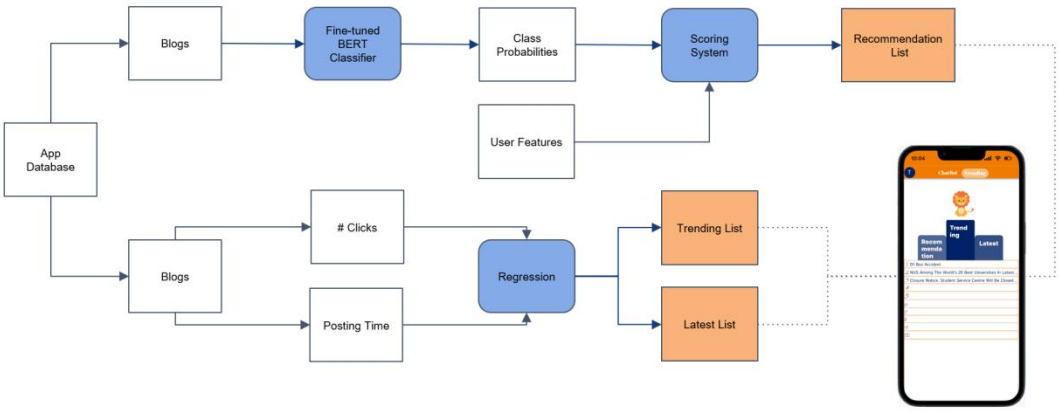

Figure 8. Design of recommendation list, trending list and latest list

4.2.2 AI ChatBot and Blog Posting

The ChatBot is a campus AI assistant capable of automatically matching students' queries with existing answers or relevant posts and recommending solutions. The ChatBot's type-in window also allows students to post blogs related to academics, campus life, extracurricular activities, and more. See Figure 9.

When user types in a question, it will be passed to an ensemble model composed of the TF-IDF algorithm and the BOW algorithm, which transforms the question text into embeddings for the down-stream matching. Meanwhile, the question will be fed into a fine-tuned BERT classifier which outputs a 25-dimensional vector corresponding to the probabilities of 25 class labels. Then the labels with probability value greater than a threshold will be used to select relevant blogs from the groups the user joined in the database. Similarly, each of the selected blogs will be passed to the same ensemble model to get blog embeddings. After that, the question embedding will be used to calculate matching scores with blog embedding in the weighted matching system. The answer to the question exists in the blogs with high matching scores. Finally, those blogs are combined and sent to a large language model by a designed prompt to extract a refined answer. However, if all the class probabilities output from the fine-tuned BERT classifier are smaller than a threshold – meaning there is no possible answer in the database, the question will be passed directly to the large language model to generate an answer.

Beside the question-answering function, users can also post blogs to pages of the groups they joined, which can facilitate the sharing of knowledge and experiences, thereby fostering a collaborative campus community.

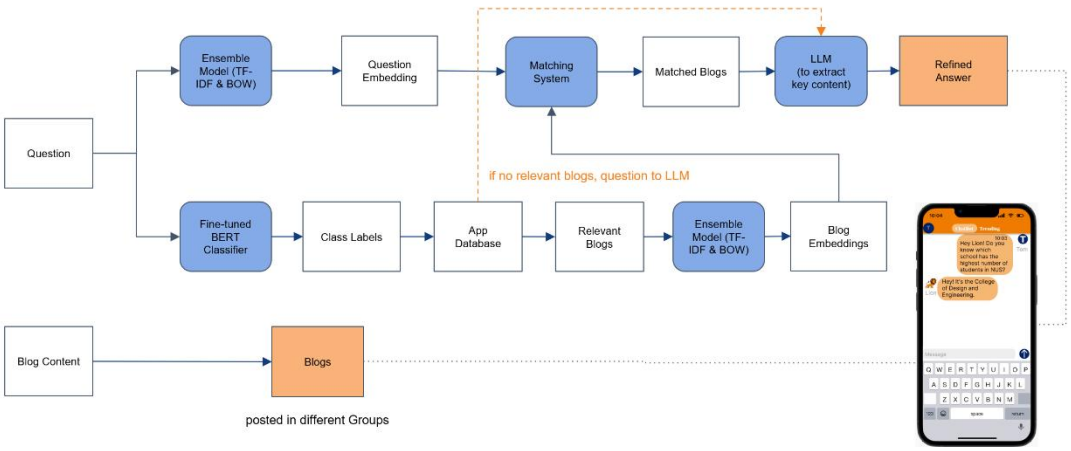

Figure 9. Design of AI ChatBot and Blog Posting

5. Data

5.1 Data Preparing

5.1.1 Dataset Searching

Searching for an appropriate dataset before training a model for downstream tasks is crucial for the success of a machine learning project. Based on our project protectives, we searched and compared the relevance, diversity, size, quality and licensing of various datasets and decided on the Microsoft News Dataset (MIND)[1].

The MIND is a large-scale dataset for news recommendation research. It was collected from anonymized behavior logs of Microsoft News website. MIND contains about 160k English news articles and more than 15 million impression logs generated by 1 million users. Every news article contains rich textual content including title, abstract, body, category, and entities. Each impression log contains the click events, non-clicked events, and historical news click behaviors of this user before this impression.

Compared with the data sets of Google and kaggle, we choose MIND because its data set contains not only the title, introduction and category of the corresponding article, but also the corresponding number of clicks, which is convenient for us to calculate the popularity. In addition, since this data set is carefully designed, it is more convenient for subsequent processing, and the large amount of data also ensures that its data distribution can be adjusted according to our needs.

[1] https://msnews.github.io/assets/doc/ACL2020_MIND.pdf

5.2 Data Understanding

To further explore and gain insights into the dataset, we employed various data visualization techniques using Python libraries, which allowed us to form a comprehensive understanding of the dataset. By identifying data distributions, patterns, and relationships, we gained insights that informed our subsequent data preprocessing and model selection. It's important to note that data visualization is not a one-time task but an iterative process, continually helping us refine our understanding of the data as the project progresses.

Firstly, the dataset structure printed shows that the data is organized as a tabular dataset, where each column represents a specific attribute or piece of information related to the news articles.
As shown in Figure 10, each row corresponds to a single news article and provides information about the article's category, subcategory, title, abstract, URL, and named entities found within the title and abstract.

	News ID	Category	SubCategory	Title	Abstract	URL	Title Entities	Abstract Entities
0	N88753	lifestyle	lifestyleroyals	The Brands Queen Elizabeth, Prince Charles, and Prince Philip Swear By	Shop the notebooks, jackets, and more that the royals can't live without.	https://assets.msn.com/labs/mind/AAGH0ET.html	[{"Label": "Prince Philip, Duke of Edinburgh", "Type": "P", "WikidataId": "Q80976", "Confidence": 1.0, "OccurrenceOffsets": [48], "SurfaceForms": ["Prince Philip"]}, {"Label": "Charles, Prince of Wales", "Type": "P", "WikidataId": "Q43274", "Confidence": 1.0, "OccurrenceOffsets": [28], "SurfaceForms": ["Prince Charles"]}, {"Label": "Elizabeth II", "Type": "P", "WikidataId": "Q9682", "Confidence": 0.97, "OccurrenceOffsets": [11], "SurfaceForms": ["Queen Elizabeth"]}]	[]
1	N45436	news	newsscienceandtechnology	Walmart Slashes Prices on Last-Generation iPads	Apple's new iPad releases bring big deals on last year's models.	https://assets.msn.com/labs/mind/AABmf2I.html	[{"Label": "IPad", "Type": "J", "WikidataId": "Q2796", "Confidence": 0.999, "OccurrenceOffsets": [42], "SurfaceForms": ["iPads"]}, {"Label": "Walmart", "Type": "O", "WikidataId": "Q483551", "Confidence": 1.0, "OccurrenceOffsets": [0], "SurfaceForms": ["Walmart"]}]	[{"Label": "IPad", "Type": "J", "WikidataId": "Q2796", "Confidence": 0.999, "OccurrenceOffsets": [12], "SurfaceForms": ["iPad"]}, {"Label": "Apple Inc.", "Type": "O", "WikidataId": "Q312", "Confidence": 0.999, "OccurrenceOffsets": [0], "SurfaceForms": ["Apple"]}]
2	N23144	health	weightloss	50 Worst Habits For Belly Fat	These seemingly harmless habits are holding you back and keeping you from shedding that unwanted belly fat for good.	https://assets.msn.com/labs/mind/AAB19MK.html	[{"Label": "Adipose tissue", "Type": "C", "WikidataId": "Q193583", "Confidence": 1.0, "OccurrenceOffsets": [20], "SurfaceForms": ["Belly Fat"]}]	[{"Label": "Adipose tissue", "Type": "C", "WikidataId": "Q193583", "Confidence": 1.0, "OccurrenceOffsets": [97], "SurfaceForms": ["belly fat"]}]
3	N86255	health	medical	Dispose of unwanted prescription drugs during the DEA's Take Back Day	NaN	https://assets.msn.com/labs/mind/AAISxPN.html	[{"Label": "Drug Enforcement Administration", "Type": "O", "WikidataId": "Q622899", "Confidence": 0.992, "OccurrenceOffsets": [50], "SurfaceForms": ["DEA"]}]	[]

Figure 10. Data samples

Secondly, visualizing the number of samples of each category in the same histogram allows for understanding of the category distribution and the distribution of its subcategories. From Figure 11, we can clearly see that the most of the samples was on the category of sports and news with newsus and football_nfl as the top subcategory respectively.

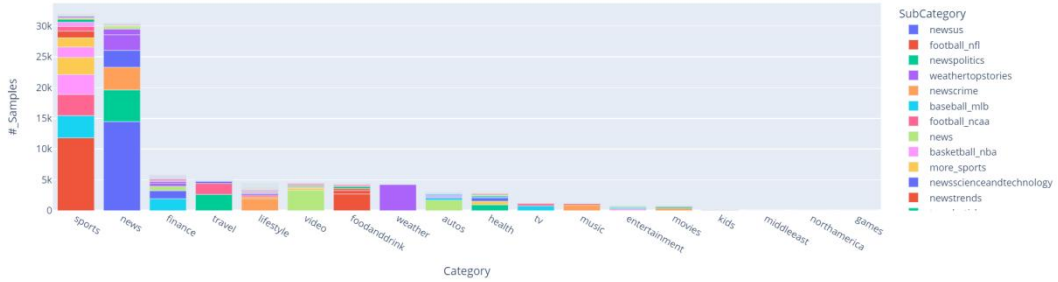

Figure 11. Category and subcategory distribution

In addition, we can dive deeper into the distribution of each category separately to see how many samples there are in each subcategory. An example of category news is shown in Figure 12.

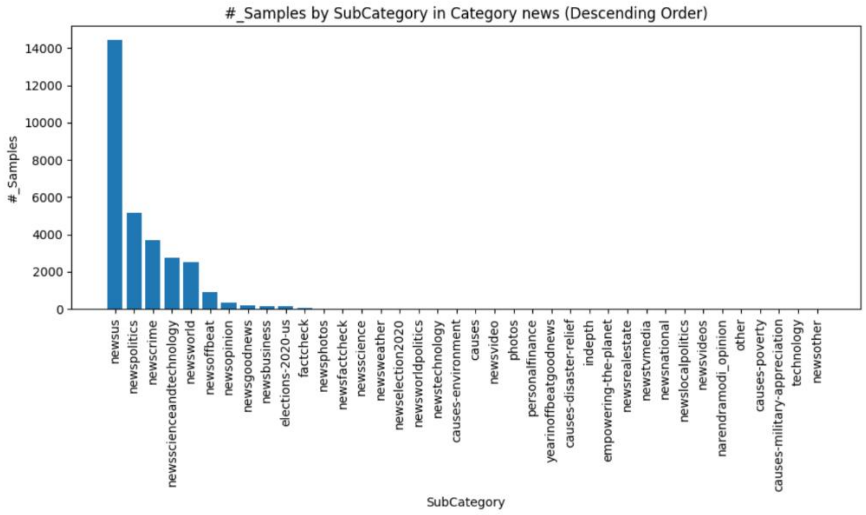

Figure 12. Samples distribution of subcategories in category news

With this exploration, the following insights are drawn:

- Imbalanced class distribution: The data exhibits an imbalanced distribution of categories, with certain categories having significantly more samples than others (e.g., 'sports' and 'news' categories). This may lead to suboptimal model performance on minority classes.
- Inter-category relationships: There may be relationships between categories, such as interrelated subcategories within the 'sports' category.
- Multi-label problem: If each sample can belong to multiple categories, it indicates a multi-label classification problem.
- Evaluation metrics: Choosing appropriate evaluation metrics to measure model performance is crucial, especially considering the imbalanced class distribution.

5.3 Data Preprocessing

Based on the data exploration of last step, pre-processing is needed to make the data applicable to the down-stream ML tasks. In this regard, subcategories that are not help in the task context are excluded; similar subcategories are merged as one; down-sampling and data augmentation techniques are employed to solve the data imbalance problem.

1) Selecting relevant features and excluding samples of low relevance

As the abstract and the categories will be used as inputs and labels separately, three columns are selected – category, subcategory and abstract. Meanwhile, irrelevant subcategories or subcategories with too few samples, namely newsus, kids, northamerica and games, are filtered and samples whose abstract value is null are excluded.

2) Separating and merging subcategories

As shown in Table 1 and Figure 13, separate subcategories with a large number of samples into an individual category, and merge subcategories which are similar to each other and of a small number of samples as one category, meaning those samples share the same label. And filter out samples in Category == “sports” whose SubCategory == “more_sports”.

Table 1. Separate, merge and assign new category

Category	SubCategory	assigned Category
sports	football_nfl, football_ncaa, football_nfl_videos, football_ncaa_videos	sports_football
	basketball_nba, basketball_ncaa , basketball_ncaa_videos, basketball_nba_videos, basketball_wnba	sports_basketball
	baseball_mlb，baseball_mlb_videos	sports_baseball
	mma , boxing-mma , mmaufc , boxing	sports_mma-boxing
	icehockey_nhl	sports_icehockey

	golf	sports_golf
	racing	sports_racing
	‘other subcategories’	sports_others
news	newspolicies	news_politics
	newsprime	news_crime
	newsworld	news_world
	newsscienceandtechnology	news_scienceandtechnology
	‘other subcategories’	news_others

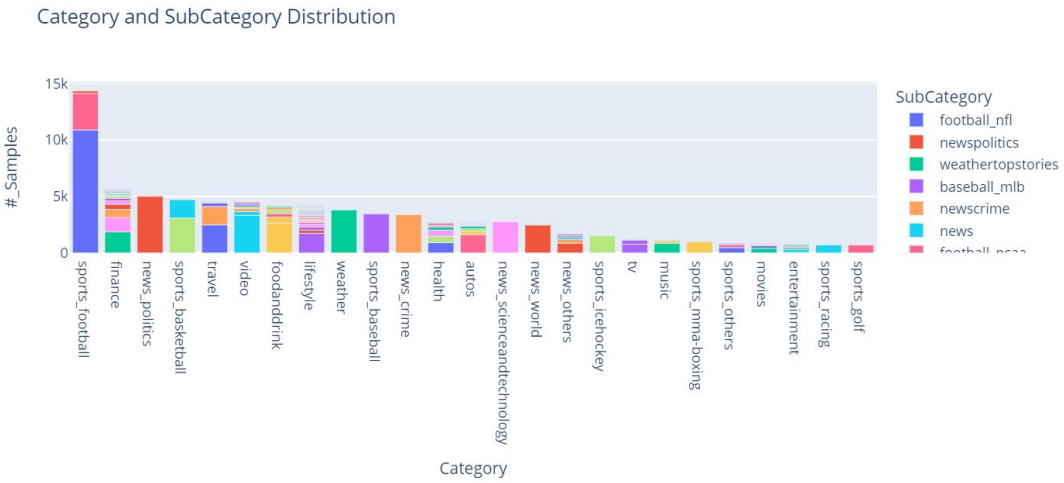

Figure 13. Category and subcategory distribution (after separating and merging)

3) Applying stratified down-sampling technique on categories with large number of samples

Even though some cleaning techniques has been utilized on the data, it’s still category-wise highly imbalanced, shown in Figure 14. For example, there are over 14,000 samples in category sports_football while less than 700 samples belong to category sports_golf. With this situation and the specific classification task considered, setting the threshold of 1000 samples to each category is reasonable.

For categories consisting of more than 1000 samples, stratified down-sampling technique is applied to randomly select out 1000 samples where the number of each subcategory corresponds to the proportion of its quantity in the category, result shown in Figure 5.

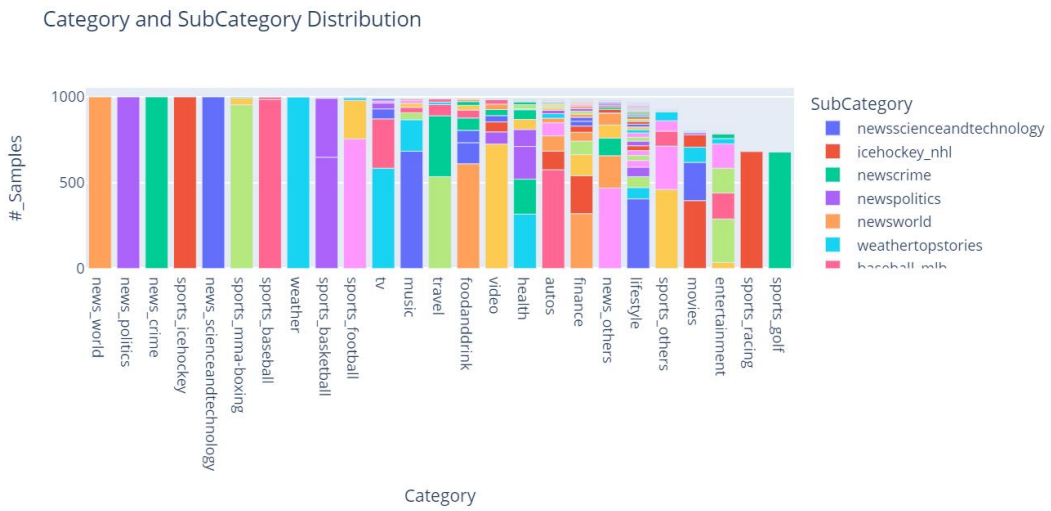

Figure 14. Category and subcategory distribution (after stratified down-sampling)

4) Applying data augmentation techniques on categories with small number of samples

For categories consisting of a number of samples greater than the threshold, data augmentation techniques should be applied. As the goal is to generate similar text from given text, we tried different algorithms (such as back translation, synonym replacement, random insertion, random swap, random deletion, etc.) and decided on Easy Data Augmentation (EDA)[2].

With EDA, synonym replacement, random insertion, random swap and random deletion techniques are combined all together to generate similar texts from given samples, result shown in Figure 15.

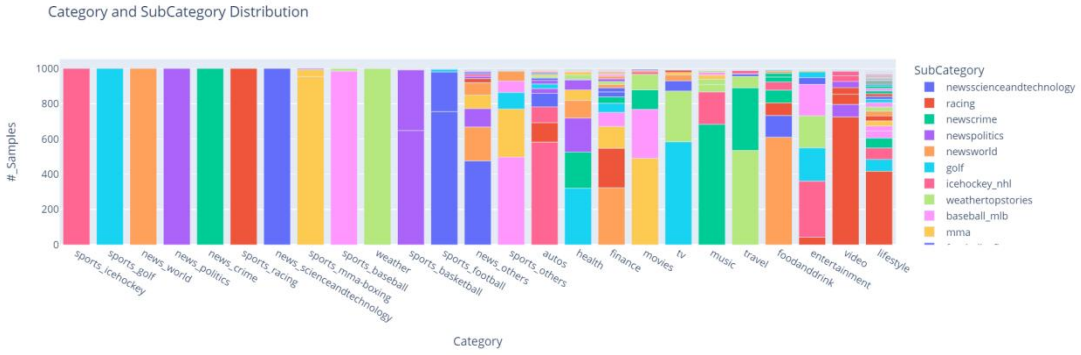

Figure 15. Category and subcategory distribution (after applying EDA)

[2] https://arxiv.org/abs/1901.11196

5) Preprocessed data

After applying the above techniques, the balanced data include 25 categories with 24916 samples in total, as shown in Table 2.

Table 2. Preprocessed data

Category	# Samples	Category	# Samples
news_world	1000	sports_others	997
news_politics	1000	autos	997
sports_racing	1000	health	996
sports_icehockey	1000	movies	995

sports_golf	1000	finance	995
news_scienceandtechnology	1000	music	994
news_crime	1000	tv	994
sports_mma-boxing	999	entertainment	993
sports_baseball	999	foodanddrink	993
weather	999	travel	993
sports_basketball	998	video	992
sports_football	998	lifestyle	987
news_others	997		

5.4 Data Modelling

5.4.1 BertForSequenceClassification as A Feature Extractor

BertForSequenceClassification is a pre-trained large language model built upon the BERT[3] architecture but is adapted and fine-tuned to perform sequence classification tasks. It has a classification head added on top of the pretrained BERT model. This classification head typically consists of one or more fully connected layers that map the contextual embeddings provided by BERT to the desired number of output classes or labels.

In this case, BertForSequenceClassification is employed as a feature extractor to generate contextual embeddings of the input text. Then the embeddings are used as input to a designed classifier to output the possible labels of the text. The working flow is shown in Figure 16.

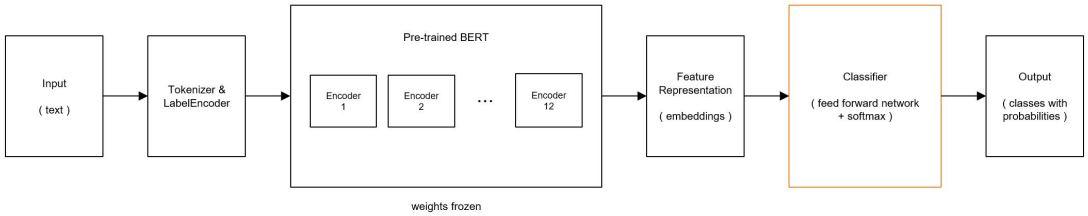

Figure 16. Flow of BertForSequenceClassification as a feature extractor

In the training process, the weights of the pre-trained BERT (BertForSequenceClassification) are frozen and only those of classifier are updated in each iteration. However, with different design of the classifier architecture (different number of layers and notes in each layer), the performance on testset is always far from desired.

Analysis: The BertForSequenceClassification model contains over 110 million parameters and has been trained on various types of large datasets of domains different from this specific task. When the distribution of training data differs significantly from the data used to pre-train BERT and the model's weights are not updated, it can't perform well after the training. A different approach is needed.

[3] https://arxiv.org/abs/1810.04805

5.4.2 Fine-tune BertForSequenceClassification + CNN layers

As the dataset we use might be significantly different from the original pre-training data and the downstream task is complex and requires the model to learn intricate

patterns, updating the BERT layers while updating the weights in the CNN layers during training is beneficial, which allows the model to adapt to the nuances and specific patterns in the data.

1) Model architecture and design

BertForSequenceClassification is a BERT-based architecture customized for sequence classification tasks. It uses the powerful pre-trained BERT encoder to obtain contextualized word representations and adds a task-specific classifier, which is designed based on use cases, on top to make predictions.

In this case, we use BERT-base, which is one of the standard configurations of the BERT model. The BERT-base's architecture includes 12 transformer encoder layers and each layer has two sub-layers: multi-head self-attention mechanism and position-wise feed-forward network. The multi-head self-attention sub-layer allows the model to weigh the importance of different words in the input sentence when generating contextualized embeddings. It helps BERT understand the relationships between words in both directions (left-to-right and right-to-left) due to its bidirectional nature. The position-wise feed-forward sub-layer applies a simple feed-forward neural network to each position separately and identically.

Figure 17 shows the adjusted working flow - updating the BERT layers while updating CNN layers.

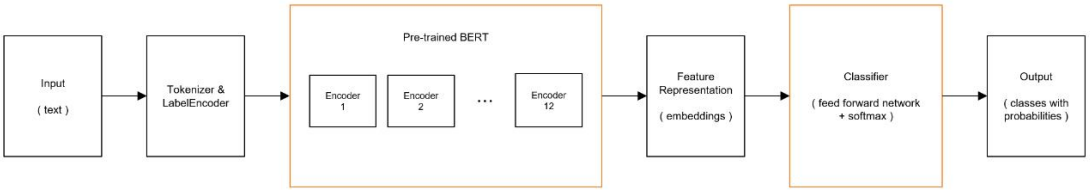

Figure 17. Flow of BertForSequenceClassification + CNN layers

2) Fine-tuning and experiments

Fine-tuning the pre-trained BERT model involves adjusting several hyperparameters to optimize the training process for this specific use case and dataset. To perform hyperparameter tuning, different learning rate, dataset split and several combinations of dense layer configurations and sizes are experimented, shown in Table 3.

Table 3. Hyperparameter adjusting

Hyperparameters		Values
batch_size		8
learning_rate		2e-05; 1e-05; 5e-06
dataset split (train/val/test)		0.7/0.15/0.15; 0.8/0.1/0.1
dense layers	number of nodes in each layer	(512, 256, 128); (512, 128); (256, 128); (512); (256); (128)
	activation function	ReLU
	dropout rate	0.1, 0.2, 0.3, 0.4, 0.5

The experiments are carried out in the deep learning framework PyTorch using a NVIDIA GPU RTX 4060. Limited to the GPU memory, the batch size is set to 8 samples.

As BertForSequenceClassification is pre-trained on a large corpus of text, which means it already has knowledge of language and a lot of useful information embedded in its weights. In the fine-tuning process, the model is expected to mainly preserve its pre-trained knowledge while making slight adjustments to address the task-specific requirements. So, starting with a small learning rate during fine-tuning is a common practice to leverage the valuable information already present in the pre-trained weights and ensure a smooth transition from the general language understanding of BERT to the specifics of this task. Based on past experience and practices of researchers and practitioners, the common starting learning rate is often set in the range of 1e-5 to 5e-5.

From the experiments, we found that the more CNN layers and nodes are added the worse the model performance tends to be. The major reason attributed to that, in this context, is likely overfitting. When more additional layers and nodes are introduced to the model, it becomes more complex and has a higher capacity to capture and memorize details in the training data. However, this increased complexity also makes the model more susceptible to learning noise and idiosyncrasies present in the training data rather than generalizing to the underlying patterns that are applicable to a broader range of data.

3) Results and analysis

From various practices, we obtained a model that performed decently on the testset. In this model only one fully connected layer of 128 nodes is added between the pre-trained BERT and the output layer. And it is iterated with 6 epochs on the trainset with dropout rate set to 0.2 and activation function set to ReLU. The model architecture is shown in Figure 18.

As shown in Figure 19, the report of performance on the testset of 2493 samples, the model's overall accuracy is 0.795, which is quite good as it's the model's performance on 25 classes. And it can generalize well on most of the categories while its classification accuracy is relatively low on a few categories such as news_others. The classification precision equals or exceeds 0.90 on six categories (sports_football, sports_golf, sports_icehockey, sports_mma-boxing, sports_others and sports_racing). Of them, it's confident that the model can perform well on sports_football, sports_icehockey, sports_mma-boxing and sports_others. As sports_golf and sports_racing are categories consisting of only one subcategory (meaning low diversity) and are augmented by around 50% using EDA algorithm, if the augmented samples and the original samples are too similar with each other and are split into the trainset and testset separately, the classification precision on the two categories is of low confidence level.

However, there are four categories (lifestyle, news_others, travel and video), on which the model's classification precision is below 0.70. A possible reason is that the sample volume is no sufficient for the model to learn the underlying patterns. For example, the category lifestyle includes 35 subcategories, which makes it rather challenging for the model to learn efficiently with a limited amount of data.

Overall, the model is robust to distinguish one class from the rest, as the Figure 20 shows.

```
BertForSequenceClassification(
  (bert): BertModel(
    (embeddings): BertEmbeddings(
      (word_embeddings): Embedding(30522, 768, padding_idx=0)
      (position_embeddings): Embedding(512, 768)
      (token_type_embeddings): Embedding(2, 768)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (encoder): BertEncoder(
      (layer): ModuleList(
        (0-11): 12 x BertLayer(
          (attention): BertAttention(
            (self): BertSelfAttention(
              (query): Linear(in_features=768, out_features=768, bias=True)
              (key): Linear(in_features=768, out_features=768, bias=True)
              (value): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
            (output): BertSelfOutput(
              (dense): Linear(in_features=768, out_features=768, bias=True)
              (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
          )
          (intermediate): BertIntermediate(
            (dense): Linear(in_features=768, out_features=3072, bias=True)
            (intermediate_act_fn): GELUActivation()
          )
          (output): BertOutput(
            (dense): Linear(in_features=3072, out_features=768, bias=True)
            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
      )
    )
    (pooler): BertPooler(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (activation): Tanh()
    )
  )
  (dropout): Dropout(p=0.1, inplace=False)
  (classifier): Sequential(
    (0): Linear(in_features=768, out_features=128, bias=True)
    (1): ReLU()
    (2): Dropout(p=0.2, inplace=False)
    (3): Linear(in_features=128, out_features=25, bias=True)
  )
)
```

Figure 18. Model architecture (BertForSequenceClassification + CNN)

```
Accuracy: 0.7950
                          precision    recall  f1-score   support

                   autos       0.80      0.77      0.78        94
           entertainment       0.77      0.68      0.72        87
                 finance       0.71      0.76      0.74        99
            foodanddrink       0.81      0.84      0.82        92
                  health       0.82      0.86      0.84        98
               lifestyle       0.53      0.46      0.49        93
                  movies       0.84      0.83      0.84        96
                   music       0.87      0.87      0.87        78
              news_crime       0.82      0.93      0.87        94
             news_others       0.43      0.45      0.44       106
           news_politics       0.73      0.89      0.80        99
news_scienceandtechnology       0.80      0.68      0.73        99
            news_world       0.82      0.63      0.72        93
         sports_baseball       0.83      0.93      0.88       117
       sports_basketball       0.82      0.86      0.84        98
         sports_football       0.92      0.84      0.88       102
            sports_golf       0.91      0.89      0.90       108
        sports_icehockey       0.96      0.91      0.93       106
       sports_mma-boxing       0.96      0.98      0.97        98
           sports_others       0.90      0.87      0.88       110
           sports_racing       0.99      0.98      0.99       101
                  travel       0.63      0.61      0.62       109
                      tv       0.72      0.73      0.72       104
                   video       0.63      0.65      0.64       102
                 weather       0.87      0.95      0.91       110

                accuracy                           0.80      2493
               macro avg       0.80      0.79      0.79      2493
            weighted avg       0.80      0.80      0.79      2493
```

Figure 19. Report of model performance on testset

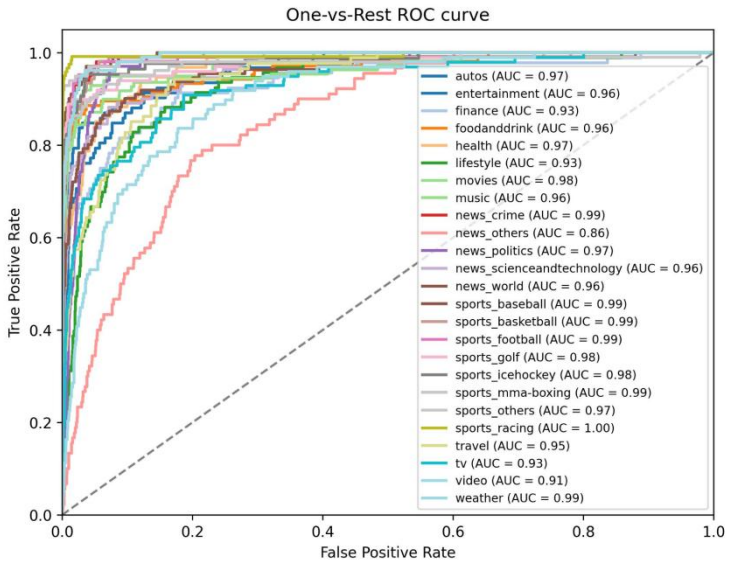

Figure 20. One-vs-Rest ROC Curves

5.5 Algorithm Design

1) Scoring system in recommendation list generation

Any blog, say Blog_i, posted in the groups the user joined will be passed to the classifier, which outputs a 25-dimensional vector corresponding to the probabilities of 25 classes, namely **p_i** = [p_i1, p_i2, …, p_i25]. Then **p_i** will be passed to the designed scoring system, where it calculates how well Blog_i matches the user's features represented by vector **u** = [u_1, u_2, …, u_25] and outputs the matching score, score_i = $< \mathbf{p\_i}, \mathbf{u} > = \sum_j p_{\_ij} * u_{\_j}$.

Based on the scores of different blogs and taking the top ones, a recommendation list will be generated and adjusted dynamically.

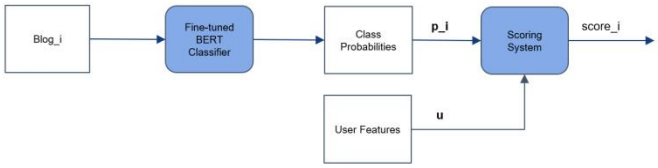

Figure 21. Scoring system

2) Heat ranking algorithm

Heat ranking algorithms are a class of algorithms widely used to assess the popularity of content, items, or entities over a specific period of time. These algorithms play an important role in social media, news aggregation, recommendation systems, search engines and other applications, helping users browse and find the most attractive content.

We first use data such as the number of clicks to calculate the basic popularity of each post, and then add a time decay function, as the Posting time becomes longer, the popularity of the post gradually decreases. From this, we can get a comprehensive

function to calculate the heat, and order the calculated heat to get the final hot search list.

```
# Define the weight of the heat calculation
clicks_weight = 0.6
time_weight = 0.4
# Calculated heat score
max_clicks = df['Click volume'].max()
df['heat'] = (df['Click volume'] / max_clicks) * clicks_weight + (1 - df['time'].rank() / len(df)) * time_weight
```

Figure 22

3) Matching system in AI ChatBot

In this section we use BOW and TF-IDF algorithms to match the input questions to our database:

BOW:
The BOW (Bag of Words) algorithm is a basic and widely used text representation method in natural language processing (NLP). It converts a text document or sentence into an unordered collection of words, taking into account the frequency with which words appear in the text.

How BOW works:
Text preprocessing: Before the BOW algorithm is applied, some preprocessing steps need to be performed on the text first, such as word segmentation (dividing the text into words or marks), removing stop words (common but no information value words, such as "the", "is", etc.), converting the text to lower case, etc.

Building a vocabulary: The BOW algorithm builds a vocabulary that contains all the unique words that appear in the text. Each word is assigned a unique index for subsequent representations.

Text representation: For each text document or sentence, the BOW algorithm converts it into a vector where the dimension of the vector is equal to the number of words in the vocabulary. Each dimension corresponds to a word, and the value in the vector represents the number or frequency of occurrences of the corresponding word in the text. In general, the frequency of words can be represented in one of two ways:

(A) Occurrences (Count) : Each dimension in the vector represents the number of occurrences of the corresponding word in the text.

(B) Word Frequency-Inverse Document Frequency (TF-IDF) : Each dimension in the vector represents the TF-IDF weight of the corresponding word, which is used to consider the importance of the word in the overall corpus.

TF-IDF:
TF-IDF (Term frequency-inverse Document Frequency) is a text feature extraction method commonly used in text analysis and information retrieval. It helps determine which words in a document are most important to the content of the document by weighting the words in the text. TF-IDF takes into account the frequency of a word in a document as well as its importance in the overall collection of documents, resulting in a vector used to represent the text.

TF (Term Frequency) :
TF refers to how often a word appears in a document. It is usually expressed as the number of times a word appears in a document, or as relative frequency (the number of word occurrences divided by the total number of words in the document). A high TF value for a word means that it appears more frequently in the document.

Inverse Document Frequency (IDF) :
IDF is the inverse document frequency of a word. It considers the importance of a single word in the entire collection of documents. IDF is usually calculated as follows:

$$\mathrm{IDF(w)} = \log(\mathrm{N}/(\mathrm{n\_w} + 1))$$

Where N is the total number of documents and n_w is the number of documents containing the word w. Add 1 to avoid having a zero denominator. The higher the IDF value, the less common and therefore more important a word is in the document collection.

TF-IDF value:
The TF-IDF value is the product of TF and IDF and is used to indicate the importance of a word in a document. Words with high TF-IDF values appear frequently in the document, but rarely in the overall document collection, and are therefore considered to contribute significantly to the content of the document.

In practice, we combine the two by weighting, which not only prevents the omission of keywords, but also completes the extraction and vectorization of keywords. From the final practical results, there are quite good effects.

6. Challenges & Solutions

The biggest challenge we encountered was that we needed to complete a huge amount of code in a short period of time. Without any relevant project materials and codes for reference, we started from scratch within the specified time, thoroughly independently completed the design, development and testing of the entire project. Moreover, since we were developing an APP, our front- and back-end framework was also more complex. From the aspects of the product, there is also no similar product in the market for reference.

1）Data Complexity: Dealing with complex data with missing values, high imbalance, inter-class relationships and multi-label problem.

Solutions: Select relevant features and exclude samples not helpful in this task-specific context; separate subcategories with a large number of samples into an individual category and merge subcategories which are similar to each other and of a small number of samples as one category; apply stratified down-sampling technique on categories with large number of samples and employ data augmentation techniques on categories with small number of samples.

2）Modelling: Fine-tuning the pre-trained large language model BERT while updating the extra CNN layers on the top

Solutions: Carry out experiments - design varying combinations of CNN layer configurations and adjust hyperparameters utilizing different methods and strategies and test the model's generalization ability.

3）Privacy and Security: Protecting sensitive information within knowledge models is crucial, and ensuring compliance with data privacy regulations is a challenge.

Solutions: Implement access control and encryption mechanisms to safeguard sensitive data. Regularly audit and assess the security measures in place to identify and address vulnerabilities.

7. Limitations

Data Quality and Availability: The accuracy and availability of data from external sources (academic journals, social media, etc.) can be a limitation. Inaccurate or incomplete data can impact the quality of recommendations and trending topics.

User Engagement: The success of the application relies on user engagement. Encouraging users to actively participate in discussions and share their experiences can be challenging.

User Learning Curve: Users may need time to become familiar with the application's features and capabilities. A steep learning curve can deter some users from utilizing the application to its full potential.

System Performance: As the user base grows, maintaining system performance and scalability can be challenging. Slow response times or system crashes can negatively affect user experience.

Bias in Recommendations: The recommendation system may inadvertently introduce biases based on user preferences and previous interactions. Efforts should be made to reduce biases and ensure diversity in content recommendations.

Content Quality Control: Ensuring the quality and accuracy of user-generated content and discussions can be a challenge. Implementing content moderation and quality control mechanisms is essential.

Cross-Platform Compatibility: While you aim for cross-platform compatibility, ensuring that the application functions seamlessly on all devices and operating systems can be complex.

8. Future Directions

Here are some potential avenues for future development:

1. Continuous Improvement:
Regular updates and improvements to the application are crucial. This includes refining the algorithms used for trend identification, enhancing the user interface, and incorporating the latest advancements in natural language processing and machine learning.

2. Integration of New Data Sources:
Expand the application's capabilities by integrating additional data sources and types. This could include academic databases, research publications, and data from emerging academic platforms. Diversifying the data sources can provide users with even more comprehensive and relevant information.

3. Cross-Platform Compatibility:
Consider developing versions of the application for different platforms, such as web browsers and desktop computers. This will enable a broader range of users, including those who prefer larger screens and desktop environments, to access the platform.

4. Customization and Personalization:
Enhance the personalization features to provide users with even more control over their preferences and the content they receive. This could involve more sophisticated user profiling and recommendation algorithms.

5. Collaboration and Knowledge Sharing Features:
Implement tools for collaborative work and knowledge sharing within the application. This could include features for group discussions, project management, and collaborative research tools, further promoting a sense of community and academic growth.

6. Analytics and Reporting:
Introduce analytics and reporting tools for users to track their engagement and contributions within the academic community. This data can help users understand their academic interests and areas of impact.

7. Integration with Academic Services:
Collaborate with NUS academic services to integrate the application with academic processes, such as course enrollment, academic advising, and career services. This integration can streamline administrative tasks and enhance the overall student experience.

8. Research and Innovation Hub:
Develop the application as a hub for research and innovation, facilitating connections between faculty, researchers, and industry partners. This could lead to collaborative research projects, technology transfer, and opportunities for students to engage in real-world research.

9. Accessibility and Inclusivity:
Ensure that the application is accessible to all members of the NUS community, including individuals with disabilities. This may involve implementing accessibility features and providing multiple language options.

10. User Feedback and Iteration:
Continuously gather feedback from users to identify areas for improvement. Iterative development based on user input is essential for keeping the application relevant and meeting the evolving needs of the NUS community.

By pursuing these future directions, the NUS mobile application can further solidify its position as a dynamic and indispensable tool for knowledge discovery, community engagement, and academic advancement within the university.

Appendix

APPENDIX A: Map of System Functionalities against Modules

System Functionalities	Modules
Recommendation List Generation	Reasoning Systems - Content-based recommendation - Measuring similarity: Cosine similarity, Euclidean distance Cognitive Systems - Natural Language Processing
Trending List Generation	Cognitive Systems - Natural Language Processing
Latest List Generation	Cognitive Systems - Natural Language Processing
AI Chatbot	Machine Reasoning - Chatbot Cognitive Systems -Natural Languages Processing Reasoning System -Knowledge Representation
Blogs Posting	Cognitive Systems - Natural Language Processing
Grouping	Reasoning Systems -Content-based recommendation

APPENDIX B: Installation and User Guide

Installation: Download the App

NOTE: The User Guide documentation can be found under the repository "ProjectReport" directory

APPENDIX C: Project Proposal

Date of proposal: Sep 25th, 2023

Project Title: Intelligent Reasoning-Based Trending Search Generation

Group ID (As Enrolled in Canvas Class Groups):
IRS Project Group 12

Group Members (name , Student ID):
Gao Yumengdie A0265061L
Gong Yixuan A0285815U
Liu Hanyue A0285013M
Wang Xinyu A0286055Y
Zhang Kenan A0285836M

Sponsor/Client: *(Company Name, Address and Contact Name, Email, if any)*
None

Background/Aims/Objectives:

Background:
In today's fast-paced digital world, students often struggle to keep up with the vast amount of information and resources available in academic communities. To address this challenge, we propose the development of an intelligent mobile application that will revolutionize knowledge discovery and community engagement within the National University of Singapore (NUS) academic environment. This application will feature a "Hot Searches" function powered by intelligent systems, enabling users to stay informed about trending topics, articles, and discussions across various academic domains.

Aims/Objectives:
- *Personalized User Platform: We aim to foster a sense of community and enhance user engagement by providing a platform where students and faculty can easily discover, interact with, and contribute to discussions on relevant academic topics.*
- *Intelligent Trends Identification: The primary aim of this project is to develop a sophisticated intelligent system capable of identifying and prioritizing trending topics, articles, and discussions within the NUS academic community.*
- *Meaningful Interactions: Our objective is to facilitate meaningful interactions and discussions among users, encouraging knowledge sharing, collaboration, and academic growth.*

Project Descriptions:

The Hot Search List feature will employ recognition systems to scan various data sources such as academic journals, news articles, social media, and university forums to identify trending topics and discussions within NUS community. This data will be processed through natural language processing (NLP) algorithms to extract relevant keywords and sentiments. A reasoning system will then prioritize and rank these topics based on user engagement, relevance, and individual preferences. Users will have access to a constantly updated list of trending subjects, making it easier for them to stay informed and engaged in their academic communities.

Our comprehensive implementation plan for the NUS mobile application involves creating a Hot Search List page with data collection mechanisms and trend identification algorithms, building a reasoning system using machine learning, establishing user profiles and recommendation systems, designing a user-friendly interface, and incorporating feedback mechanisms. Rigorous testing and regular maintenance are crucial for application performance. These steps aim to empower NUS students and faculty, fostering connections and enhancing academic experiences within the university community.

APPENDIX D: Individual Report

Gong Yixuan A0285815U

1. Your personal contribution to the project:
1. Conceptual Design and Strategic Planning: Took the lead in formulating the product's concept, defining its product logic, and structuring its architecture. wrote versions of project proposals, meticulously deciding on the project's construction approach, and selecting appropriate methods and models for each segment to ensure optimal functionality of the entire product. Design a strategic plan for the product's future development.
2. Front-End Development: Played a pivotal role in crafting almost all the project's front-end, ensuring a user-friendly interface and user experience.
3. Collaboration with Back-End: working diligently with the backend team to build data formats and HTTP requests. Fixing bugs, and making sure that this app can function on its own
4. Chatbot Development and Prompt Engineering: Took charge of developing the chatbot and implementing prompt engineering techniques, contributing significantly to the project's interactive capabilities on the front end.
5. Additional Contributions: so much more that can't be listed, since it is a very complicated project and we planned to make an actual product that is valuable so we did work for the future development of the product and engineering or visualization efforts

2. What you have learned from the project?
I did a lot of design and engineering parts of the project so I think through this project I gained the most is how to build a complete product and application, we did everything from scratch by a team that we built up together from the beginning. We first brainstormed, many versions of project proposals, then kept testing and arguing about product form and implementations. Later we start building the APP from scratch, we did UI design, front-end building and server building. There are so much that is unknown and undiscovered unless one have done it all together once.

I learned the real difficulty of running a model in a market scenario. And how to gain data. I learned that to build a AI system we need to design how to gain data, and what type of data are available to us. Meanwhile, we learned that we need to design a complete intelligent system structure(especially) and the start. We faced many challenges during the process, and a lot of them is caused by failing to find data. The other half is due to the lack of clarity on the project scope among the group.

During the process, I have learned methods and ways to improve model performances, and the importance of switching mind sets, and implement different models to improve performances in different scenario or reach the same result with less consumption.

3. How you can apply this in future work-related projects.
We are definitely going to continue building this project, in the future. Very likely to keep developing it in future courses we are going to have in ISS. I wish we could do build more software agents to solve different problems, or build more

In the future, I wish to improve the data fetching part. Currently, we are merely using common methods we can find and implement them. During the project, I found that it is crucial and necessary to find a new algorithm or a new organization of algorithms and models to build a more accurate data-fetching system. I believe that if it can have a huge improvement in terms of accuracy, this app we have will have much much more potential users and markets.

For data, once we can officially launch our APP, we can collect our own data and build our own data base. In that way, we can collect more features, see how users are going to use them, and improve the product based on actual user feedback. Meanwhile, by then we can see how accurate our algorithm really is, since the real-life scenario is much more diverse and complicated than our test sets.

Zhang Kenan A0285836M

1. Your personal contribution to the project.
My personal contribution to the project consists of several parts.In this project, I am the designer the whole system.My work includes implementing all functions of backend, algorithm adjustment,database design and deploy the project to the server

As for implementing the functions, we used flask as the backend framework. The functions include three main parts: user, blogs and chatbot. User sector includes login, register.User input their personal information to App and system will check whether username and phone number have been used or not. We also used sha256 to encode the password. For the blogs, we designed three functions, including blog list, blog recommendation, latest blog.The core is blog recommendation. My partners designed the algorithm and i adjusted it and apply it to recommend blogs based on users' preference. For the chatbot, we designed two functions: posting and answering.For posting, user input blogs and the system will classify it. Then the blog will have creator , types ,posting time and other attributes, then will be stored in the database. Answering,which is core of our third part, is based on the matching algorithm. When user input question,system will get the types of it based on classification algorithm and give the result through our matching algorithm.

All the database operations I mentioned above are done by SqlAlchamy. I designed the model of every object and used Flask-migrate to map it to the Mysql database.

Finally, I deployed the project to server. I choose the tencent cloud and Centos 7.6 as the operation system. In addition, in order to make our software concurrent I used Uwsgi to achieve Multi-threads. I also tried to use Nginx to proxy backend server which can help frontend access and load balancing.

2.What you have learnt from this project:
In this project, I learned the skills are：
1)Use Flask and ORM to set up backend quickly
2)Design the algorithm and try different methods like Bow,Tf-IDF to vectorize text.
3)Try to use pytorch to adjust the transformers models.
4)Deploy the project to cloud server.
5)Learn a lot about operations based on Linux

In the beginning, the difficulty of system we faced was that there is no App on the market that we can refer to and no appropriate data set.the existing data set that we could find could not meet our needs for model training, so we needed to make specific adjustments to it. We also investigated general software features to meet the integrity of our system.In addition, it is also the first time that I did an app which integrate flask as backend and flutter as frontend. In this process, we faced a lot of problems especially about the form of data.

About the backend, i also faced some problems. Regarding the operation of the database, at first I was using the origin sql. However, in the process of learning flask, I learned about Sqlalchamey and orm, which have a lot of advantages like improving development efficiency,Database independence,Cross-database support.

Another problem is transfer the models to our system. After training the model, I had to adjust my version of some packages because of the incompatibility of models packages and adjust the details of the model, as well as the inputs and outputs, to ensure that it works properly on the system.

The last problem is concurrency and deploying problem. Because we want to implement a real app as much as possible, the deployment is very important. In this process, I learned a lot of operations abut linux system, gained knowledge about threading pool tried to use nginx to proxy the server.

3. How you can apply this in future work-related projects.

I can apply the skills learnt in many ways. Our ultimate goal is to build an LLM for NUS students (like an NUS version of chatgpt). At present, it is only a preliminary work. In the follow-up process, we can:

1) Complete the functions of our App. There are also some areas we can discover about our project and privacy is also problem that we cannot solve well. Some interfaces are not sufficiently standardized.

2) Improve our language model. At present, there is still some room for improvement in the matching degree between our problems and the text. We can improve and modify the text matching algorithm after mastering it, and our goal is to make it have the function close to chatgpt in NUS database.

Wang Xinyu A0286055Y

1. Your personal contribution to the project.
My personal contribution to the project consists of several parts.In this project, I am the main algorithm researcher. my job scope includes implementing data gathering, data processing, model training and algorithm design.

As for data collection and processing, we collected data sets widely used at present, and after comparative analysis, we decided to make modifications mainly with Microsoft's MIND data set. After a series of data pruning, data enhancement, database expansion and balance, we finally determined the data set used for training.

In terms of model training, in the hot search part, we used a regression model to sort the hot search, and modified and adjusted the existing model to a certain extent, so that the popularity value of each post was determined by its clicks and Posting time.

In terms of algorithms, in the Hot Search algorithm section I used the Newtonian cooling algorithm, which makes the popularity of each post decrease over time. In the matching algorithm of chabot, I also use the vector matching algorithm: word embedding and keyword matching algorithm: BOW and TF-IDF. For these two methods, we choose the better one to match, to ensure the accuracy.

2.What you have learnt from the project.
In this project, I learned the skills are：
1) Process the data set so that it fits the requirements of our model.
2) Use the vector and keyword matching algorithm to match the problem with the content, and achieve a high accuracy.
3) building a content-based recommendation system from start to finish using Python tools like Pandas, ScikitLearn, Word2Vec and Hugging Face's BERT Sentence Transformer.

In the beginning, the difficulty we faced was that there was no suitable data set, and the existing data set that we could find could not meet our needs for model training, so we needed to make specific adjustments to it. In this process, we initially only considered the format adaptation problem, and did not pay attention to the content, but after several training we found that the accuracy of the model did not meet our expectations. Therefore, we shifted the focus to the quality of the data. We reduced the redundant data and specially strengthened the relatively small number of categories, so that the whole data set reached a balance in content, so as to achieve an improvement in accuracy.

Then on the text matching and modeling part, in the beginning, the method we use is Word Embedding, but in actual use, we find that the accuracy of this method is too low. In repeated testing, we find that there is inherent similarity difference between the problem and the text, so the accuracy is low. Therefore, we introduce a keyword matching algorithm to effectively eliminate the differences between texts and amplify the influence of more important keywords. In the process, I learned how to choose the most suitable matching algorithm in different situations.

3**.** How you can apply this in future work-related projects.

I can apply the skills learnt in many ways. Our ultimate goal is to build an LLM for NUS students (like an NUS version of chatgpt). At present, it is only a preliminary work. In the follow-up process, we can:

1) Complete the data collection, build our own database, which includes a series of tags such as blog, click volume, category, etc., and use this as the training set to train our own recommendation model, so as to make the recommendation more personalized and accurate.

2) Improve our language model. At present, there is still some room for improvement in the matching degree between our problems and the text. We can improve and modify the text matching algorithm after mastering it, and our goal is to make it have the function close to chatgpt in NUS database.

Liu Hanyue A0285013M

1. Personal contribution

In this project, we developed a campus App from scratch, which is integrated with three intelligent systems powering five major functions: a trending list, a recommendation list, a latest list, a campus AI assistant and a blog posting function.

Our group consists of students with diverse backgrounds and therefore team members are assigned with tasks that best fit their skills so that everyone's strength is leveraged to the most.

And I am mainly responsible for data preprocessing, CNN design and modelling, and algorithm design (recommendation list), which are briefly stated below:

1) Data preprocessing
   The original text dataset is class-wise highly imbalanced and some classes are related to each other. Besides, there are missing values in important features. And some samples are of low relevance to our use case. So, I first selected features and excluded irrelevant samples and samples with missing values in the selected features. Then I separated subcategories with a large number of samples into an individual category and merged subcategories which are similar to each other and of a small number of samples as one category. For the class imbalance problem, I applied stratified down-sampling technique on categories with large number of samples and employed data augmentation algorithms on categories with small number of samples.

2) CNN design and modelling
   To train a model that can best perform on this specific text-classification task. I designed CNN layers of different configurations and sizes adding on the top of a pre-trained large language model Bert. Then I carried out experiments to fine-tune the "pre-trained Bert + CNN" model by adjusting the hyperparameters.

3) Algorithm design (recommendation list)
   I also proposed the algorithm design for generating the recommendation list and it's adopted. And this is how it works: Any blog, say Blog_i, posted in the groups the user joined will be passed to the classifier, which outputs a 25-dimensional vector corresponding to the probabilities of 25 classes, namely p_i = [p_i1, p_i2, …, p_i25]. Then p_i will be passed to the designed scoring system, where it calculates how well Blog_i matches the user's features represented by vector u = [u_1, u_2, …, u_25] and outputs the matching score, score_i = $\mathbf{< p\_i, u >} = \sum_j p_{\_ij} * u_{\_j}$. Based on the scores of different blogs and taking the top ones, a recommendation list will be generated and adjusted dynamically.

4) Other contribution: teamwork and collaboration

There have been a lot of teamwork. In the beginning, I suggested to form the team. Then in the ideation phase, I proposed several project ideas that were taken into consideration. When we decided on developing a campus App, I proposed the functions of trending list and recommendation list and the tool we used for the App UI design. Beside the contribution in many brainstorming sessions, I worked closely with Wang Xinyu on how to design the algorithms and what methods to use. In addition, I collaborated with the Zhang Kenan on integrating the model I trained into the backend. Lastly, I also worked together with other team members on preparing presentations and writing report.

2. What learnt is most useful

For me, the whole project is a learning and practicing process. And I'm very happy and satisfied with what I have achieved and learned in this practice.

Many of the skills I learned in class have been put into practice and been improved. Besides, new skills and knowledge are gained, even though some are basics.
Including but not limited to:

- ✓ Solve data imbalance issue: I applied stratified down-sampling techniques and employed data augmentation algorithms.
- ✓ Mechanism of the large language model Bert: Transformers structure, tokenization, attention mask, encoder + embeddings.
- ✓ Set up a compatible and reliable DL computational environment: I have successfully created environments with different configurations - "Windows 11 + VS Code + CUDA + PyTorch" and "Ubuntu 22.04 + Jupyter Notebook + CUDA + PyTorch".
- ✓ CNN design: build CNN layers of different configurations and sizes.
- ✓ Fine-tune model and adjust hyperparameters; analyze and improve model performance.
- ✓ Debug utilizing online resources, such as DL communities, ChatGPT.
- ✓ Other skills: teamwork, communication, presentation, etc.

3. How to apply the knowledge and skill to other situations or future workplace

The skills and knowledge gained in this practice project have provided me with a valuable set of capabilities that can be applied in future projects across various domains.

1) Further improvement and future courses

   If we receive great user feedback after launching the App, we will update it, in which case I will continue to work on the algorithm and modeling part employing different strategies such as ensemble techniques. Besides, this project has ignited my interest in NLP, so next semester I will take the Practical Language Processing module to go further with my experience and skills gained from this practice.

2) Future work-related projects

- ✓ Setting up compatible and reliable deep learning environments is crucial for any data science or machine learning project. This skill will save me time and ensure a consistent environment for my future work.
- ✓ I can apply my ability to fine-tune models and optimize hyperparameters to various machine learning models and algorithms.

- ✓ My experience in building CNN layers with different configurations and sizes can be applied to tasks like image classification, object detection, and image segmentation.
- ✓ In real-world projects, imbalanced datasets are a common issue. I can apply my knowledge of stratified down-sampling and data augmentation techniques to address this problem and improve the performance of machine learning models in scenarios where data is skewed.
- ✓ Understanding the architecture and workings of large language models like BERT is helpful in NLP tasks. I can leverage this knowledge to work on text classification, sentiment analysis, and language generation tasks.
- ✓ Teamwork, communication, and presentation skills are transferable to any work-related project. Effective collaboration with colleagues, clear communication of findings, and the ability to present my work to stakeholders are essential in any professional setting.

In a word, the skills and knowledge I've gained can be applied in a wide range of work-related projects, from natural language processing and computer vision to data analysis and model optimization. These skills not only enhance my technical abilities but also equip me with the tools to work effectively in a team and communicate results and findings to a broader audience.

Gao Yumengdie A0265061L

I am responsible for the UI design and frontend development using the Flutter framework. This report provides an overview of my contributions to the project, the methodologies I employed, the achievements I made in the UI design and frontend development aspects, the thins I learnt and the future directions.

1. personal contribution to group project:

UI Design:

Design Principles In the UI design phase, I adhered to the following principles:

User-Centered Design:

I prioritized the needs and preferences of the end-users to create a user-friendly interface.

Consistency:

I maintained a consistent design language and layout throughout the application for a cohesive and intuitive user experience. Design Tools I utilized industry-standard design tools, such as Figma, to create wireframes, mockups, and prototypes. These tools allowed me to iterate and gather feedback efficiently.

Achievements Designed an aesthetically pleasing and intuitive user interface that aligns with the project's objectives. Created wireframes and interactive prototypes to visualize the user flow and gather early feedback from the team.

Collaborated closely with the project team to incorporate feedback and make design improvements.

Development Process I followed an iterative development process that involved the following steps:

UI Implementation: I translated the design into code, ensuring pixel-perfect accuracy and consistency with the mockups.

2. what learnt is most useful for you
For front-end
Integration with Backend: I learned to collaborate with backend developers to connect the user interface with the project's data and services.

Testing and Debugging: Rigorous testing was conducted to identify and resolve any issues related to UI responsiveness and functionality.

Performance Optimization: I learned to optimize the application for speed and responsiveness to deliver a smooth user experience.

For the overall project:
Collaboration and Communication: Your experience of collaborating closely with team members to incorporate feedback and make improvements is a valuable skill in any workplace. Effective collaboration and communication are essential in achieving project goals and maintaining a positive work environment.

User-Centered Design: Prioritizing the needs and preferences of end-users is a universal principle that can be applied to various design projects. Whether it's web design, app design, or even product design, creating user-friendly interfaces is key to success.

Consistency and Design Tools: Maintaining a consistent design language and using industry-standard design tools like Figma are practices that can be carried over to different design projects. Consistency is vital for creating a cohesive and intuitive user experience, and proficiency in design tools is a valuable asset in the creative industry.

3. how you can apply the knowledge and skills in other situations or your workplaces.
The UI design and frontend development skills can be applied in various roles and workplaces, including UI/UX design, frontend development, full-stack development, project management, freelance work, startups, education, and cross-functional roles, making them highly versatile and transferable.

To enhance our language model's performance, we acknowledge that there is room for improvement in aligning our responses more closely with the specific problems presented. After acquiring a deeper understanding of this challenge, we aim to refine and adjust our text matching algorithm. Our ultimate goal is to make our model's capabilities more akin to ChatGPT when interacting with the NUS database