

PROJECT REPORT

**PRACTICE MODULE FOR CERTIFICATE IN:
PATTERN RECOGNITION SYSTEMS (PRS)**



EMOTION AWARE

Intelligent Agent

Altering engagement and questions in real time based on user emotion classification.

TEAM MEMBERS

RICHARD CHAI

Contents

1.) Executive Summary	3
2.) Problem Statement / Business Opportunity	4
2.1) Assumptions	5
2.2) Project Scope	5
2.3) Project Objectives	6
3.) Solution	7
3.1) First function (Emotion Classification)	10
3.1.1) First function (Emotion Classification – Logistic Regression)	10
3.1.2) First function (Emotion Classification – DistilBERT)	11
3.1.3) First function (Emotion Classification – LLM GPT3.5 Turbo)	14
3.2) Second function (Intelligent Agent (Orchestrator) - ReAct Framework)	15
4.) System Architecture	16
5.) Limitation	17
6.) Conclusion	17
7.) Improvements	17
8.) Bibliography & Appendix	18

1.0 EXECUTIVE SUMMARY

Notes: Based on the professors' feedback during the first proposal presentation, which I am truly thankful for, I have modified the scope of this project to incorporate more elements of Supervised Machine Learning and Neural Network, and also the scope of this project to what is achievable within 10 man-days as advised. For the rest of the project scope, I intend to work on them during the Practical Language Processing, Intelligent Software Agents practice modules and Capstone Project to attain bring the full vision of the original project proposal.

In today's corporate landscape, ongoing employee training is a fundamental requirement for companies to stay competitive. Training encompassing sales techniques, marketing promotions, product knowledge, and technical skills is essential to maintain a competitive edge.

Corporate training, unlike other forms of training does not have the goal of differentiating users by scores (grades) as the primary objective.

For example, MAS has regulations that required all IT departments in the finance sector to adhere to, banks would want all relevant staff to know that content and apply it to their daily work. Internal training and tests may be provided to staff, however, such tests tend to focus on measuring the test-takers scores, as opposed to focusing on ensuring that the learner did actually understand the content and is motivated to apply it.

What if the training courseware had the ability to modify the training content and the test questions, to personalise the training, such that the user felt motivated to learn?

To do this, the first step is to know the course taker's emotions, so that the courseware can vary the course content and quiz questions in the right tone and right level; and that is the objective of this project.

A proof-of-concept AI Agent in the form of a chatbot interface that alters question difficulty based on the user's emotions.

For technical details, please refer to the following GitHub repository:

- <https://github.com/atsui888/Pattern-Recognition-Systems>

2.0 PROBLEM STATEMENT / BUSINESS OPPORTUNITY

The Corporate Training market size was valued at USD 145465.91 million in 2022 and is expected to expand at a CAGR of 9.57% during the forecast period, reaching USD 251715.61 million by 2028.

- <https://www.linkedin.com/pulse/corporate-training-market-size-regional/>

Online corporate training can be improved in various ways to enhance the learning experience for participants. Among the many strategies and considerations for improving online training, one that stands out as being sorely needed yet difficult to achieve is:

- Personalised Content:
 - Tailor the training to the individual needs and learning styles of participants. Offer options for self-paced learning and provide opportunities for learners to choose their own paths within the training.
- Where Personalisation is achieved via:
 - Content Chunking:
 - Break down content into manageable chunks or modules, making it easier for learners to digest and retain information.
 - Modifying the difficulty of the questions based on the user's level of proficiency, which we can estimate based on the user's emotion.
 - For example, a user who is facing a series of questions that are far in advance of his capabilities would tend to feel frustrated, angry or perhaps sad.

The difficulty lies in the question of how any courseware system knows the needs of the learner.

The typical method is to have the learner take a test and then based on the results of the test, determine what intervention is required to help the user.

Instead of such reactive methods, an Intelligent Agent which modifies training content and questions proactively during the training session is likely to have higher levels of learning engagement, better learning experience and learning outcomes.

Hence, the motivation behind this proof-of-concept project, to have an Intelligent Agent assess the emotional status of the learner in real-time, alter the training content into manageable chunks and modify question difficulty such that the learner becomes more motivated to understand the objective of the course because it is personalised for them.

2.1 ASSUMPTIONS

The project makes a few assumptions about the users' computer proficiency. First, the project assumed that the users are able to use a computer. Second, the users are assumed to be able to use an internet browser and its basic functions.

2.2 PROJECT SCOPE

The scope of this proof-of-concept:

- Create Chatbot Interface
- Classify the User's Emotion using:
 - o Logistic Regression
 - o DistilBERT
 - o LLM (GPT 3.5 Turbo)
- Create an Intelligent Agent that will orchestrate the emotion classification and alter the responses / questions based on random or custom data provided by user.
 - o User who is frustrated at the difficulty of the current question will have an easier question created for them.

- User in a better emotional state due to finding the current question too easy, will have a more advanced question created for them.
- User who has a neutral emotion with the current question, will have a question generated at the same difficulty level.

2.3 PROJECT OBJECTIVES

Objectives of the Emotion Aware Intelligent Agent:

- Enhance the User Learning Experience
- Significantly increase the likelihood of users understanding the training content and applying it in their workplace due to personalisation of engagement and question generation.

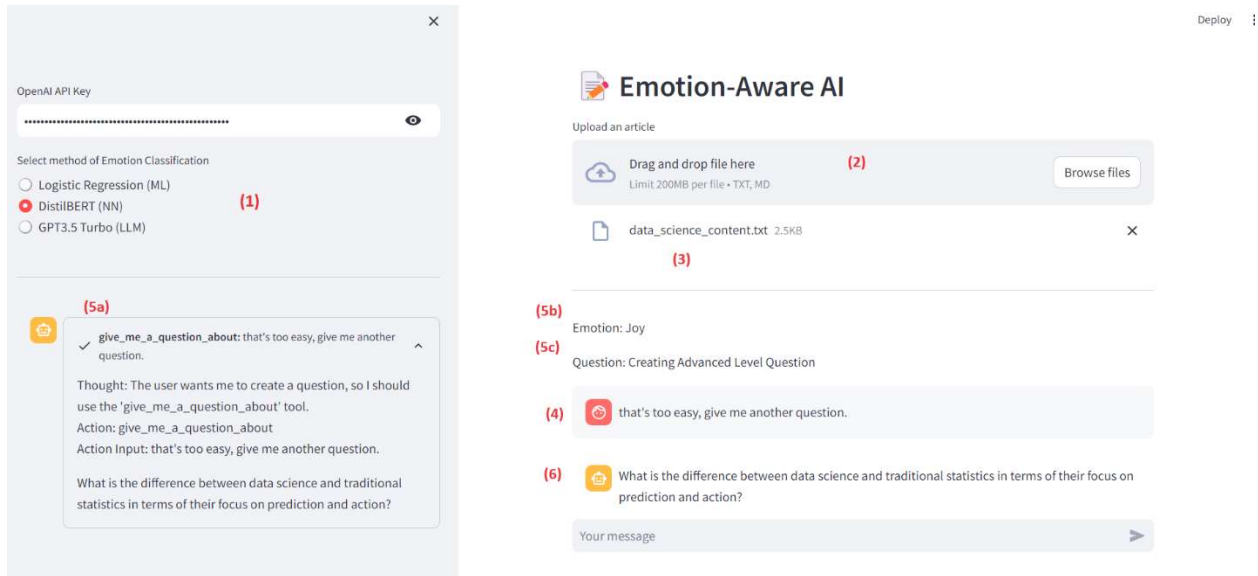
The Intelligent Agent has the following functions and features.

Table 1: Function and Features

Functions	Features
Emotion Classification	<p>Three Methods of classifying the User's Emotion from text input:</p> <ul style="list-style-type: none"> - Logistic Regression - DistilBERT trained neural network - LLM (GPT3.5 Turbo)
Intelligent Agent (Orchestrator) - ReAct Framework	<p>Synergizing Reasoning and Acting in Language Models (ReAct) - https://arxiv.org/abs/2210.03629</p> <p>As implemented in Langchain and adapted for this proof-of-concept.</p>

3.0 SOLUTION

The Emotion Aware AI Agent adopts a chatbot interface to showcase its ability to adapt the questions it asks from random or custom content in accordance with the user's emotional state.



Usage:

1. Method of Emotion Classification, select:
 - a. Logistic Regression
 - b. DistilBERT
 - c. GPT 3.5 Turbo
2. Custom Data/Content
 - a. If desired, add your own custom content for the Agent to use when generating a question.
 - b. If not added, the Agent will randomly select a topic to generate a question.
3. Custom Content Loaded Label
 - a. If a custom content file is loaded, the file name will appear here, otherwise it will be blank.

4. User Message

- a. In the screenshot above, the user was previously given a question and he responded by saying “that’s too easy, give me another question”.

5. AI Agent ReAct framework

a. ReACT framework in action:

i. Thought:

The user wants me to create a question, so I should use the 'give_me_a_question_about' tool. (among the list of tools I have access to)

ii. Action:

I will invoke the 'give_me_a_question_about' tool.

iii. Action input:

And provide the tool with the input argument, "That's too easy, give me another question".

iv. Notes:

The Agent sometimes hallucinates and sends an argument to the tool that is NOT what the user typed in the textbox. Currently, I have implemented a number of hacks in the code to minimise this.

b. Emotion Label

- i. This label displays the emotion that the Agent classified based on the text input argument it was given.
- ii. In the screen shot above, "That's too easy, give me another question." was classified as “joy”.

c. Question Label:

- i. This displays the level of difficulty of the question the Agent is going to generate.
- ii. There are 3 levels, “Beginner”, “Medium”, “Advanced” question.

- iii. In the screenshot above, as the user felt the question given earlier was too easy, the emotion detected was ‘joy’, hence the Agent decided to create an “Advanced” question.

6. Output:

- a. This is the question that the Agent is asking the user.
- b. I think most people will agree that it is a hard/advanced question.
- c. However, there are some inconsistencies which are noted in the section “Areas of Improvement” below.

In this proof-of-concept (POC), the Agent shows its ability to alter the difficulty of question generation based on custom content in real-time depending on the user’s state of emotion.

Next, the report will discuss in-depth and zoom into each function and explain related design background.

3.1 First function (Emotion Classification)

3.1.1 Logistic Regression

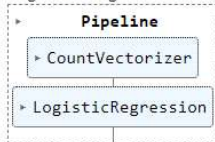
Baseline models using ‘logistic regression’, ‘random forest’ and ‘multinomial naïve bayes’ algorithms were trained on the annotated emotion dataset "dair-ai/emotion" for evaluation.

6.1 Baseline Models

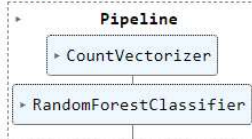
```
In [11]: baseline_models = {}
models = {
    'logistic regression': LogisticRegression(random_state=42),
    'random forest': RandomForestClassifier(n_estimators=10, random_state=42),
    'multinomial naive bayes': MultinomialNB()
}
ml_pipeline = {}
for model_name, model in models.items():
    ml_pipeline[model_name] = Pipeline(steps=[('cv', CountVectorizer()), (model_name, model)])

from sklearn import set_config
set_config(display='diagram')
for model_name, pipeline in ml_pipeline.items():
    print(model_name)
    display(pipeline)
    print()
```

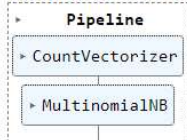
logistic regression



random forest



multinomial naive bayes



The initial results were as follows:

```
'logistic regression' baseline score: 0.8859
'random forest' baseline score: 0.8359
'multinomial naive bayes' baseline score: 0.7466
```

Additional hyper-parameter optimisation tuning was performed and as “logistic regression” was selected for its classification performance.

```

: y_pred_lr = baseline_models.get('logistic regression').get('trained_pipeline_model').predict(X_test)
print(classification_report(y_test, y_pred_lr, target_names=labels_d.values()))

```

	precision	recall	f1-score	support
sadness	0.92	0.93	0.93	933
anger	0.89	0.93	0.91	1072
love	0.83	0.75	0.78	261
surprise	0.90	0.87	0.88	432
fear	0.84	0.83	0.83	387
joy	0.79	0.69	0.73	115
accuracy			0.89	3200
macro avg	0.86	0.83	0.85	3200
weighted avg	0.88	0.89	0.88	3200

Finally, a REST API Endpoint was created the Logistic Regression Emotion Classifier was made available as a Docker Image that can be run as a Docker Container locally or in the cloud.

3.1.2 DistilBERT

The foundation model used is “distilbert-base-uncased“ (<https://huggingface.co/distilbert-base-uncased>) which itself is a distilled version for the BERT base model.

DistilBERT is a transformers model, smaller and faster than BERT, which was pretrained on the same corpus in a self-supervised fashion, using the BERT base model as a teacher.

The “distilbert-base-uncased” model can be used for either masked language modelling or next sentence prediction, but it's mostly intended to be fine-tuned on a downstream task.

And for this project, the model was fine-tuned using the annotated emotion dataset "dair-ai/emotion" with the following parameters:

Setting Training Arguments


```
In [33]: from transformers import TrainingArguments

training_args = TrainingArguments(
    output_dir="distilbert-emotion",
    num_train_epochs = 3,
    per_device_train_batch_size=64,
    per_device_eval_batch_size=64,
    weight_decay=0.01,
    evaluation_strategy="epoch",
    save_strategy = "epoch",
    load_best_model_at_end = True,
    push_to_hub = True,
    report_to = "none"
)

In [35]: from transformers import Trainer

trainer = Trainer(
    model = model,
    args = training_args,
    compute_metrics = compute_metrics,
    train_dataset = emotions_encoded["train"],
    eval_dataset = emotions_encoded["validation"],
    tokenizer = distbert_tokenize
)
trainer.train()
```

And the results are as follows:



Epoch	Training Loss	Validation Loss	Accuracy
1	No log	0.195365	0.926000
2	0.349400	0.147217	0.937000
3	0.349400	0.133256	0.938500

Here is a sample of the classifications (top-k=3):

```
i feel especially strongly about this since i have hated my teeth forever i was one of the unlucky ones who got bad genetics and an even worst orthodontist and pediatric dentist
anger (3)    --> 0.74
sadness (0)  --> 0.25
love (2)     --> 0.01
-----

i feel like we barely know each other and time just isnt being generous with our love
love (2)     --> 0.73
joy (1)      --> 0.27
surprise (5) --> 0.0
-----

i was feeling extremely shitty physically this morning
sadness (0)  --> 1.0
anger (3)    --> 0.0
joy (1)      --> 0.0
-----

i looked around and once again was disappointed that so little had shown up this evening but apparently this was my day to feel selfish
anger (3)    --> 0.99
sadness (0)  --> 0.0
fear (4)     --> 0.0
-----

i feel a bit strange publishing these beautiful photos
fear (4)     --> 0.66
surprise (5) --> 0.33
sadness (0)  --> 0.0
-----

i promised myself that i wont enter anymore giveaways because i feel greedy but i couldnt resist this one
anger (3)    --> 0.99
fear (4)     --> 0.0
sadness (0)  --> 0.0
-----

i feel like im almost uh afraid of everything so to speak
fear (4)     --> 0.99
sadness (0)  --> 0.0
anger (3)    --> 0.0
-----

i was feeling abnormally wimpy so i staked out my bird feeder
fear (4)     --> 0.99
surprise (5) --> 0.0
sadness (0)  --> 0.0
-----

i feel like my trust is being abused the less i feel like theres a future for us
sadness (0)  --> 1.0
fear (4)     --> 0.0
anger (3)    --> 0.0
-----

i still feel sleep deprived she is almost sleeping through the night giving us
sadness (0)  --> 1.0
fear (4)     --> 0.0
anger (3)    --> 0.0
-----

I've been waiting so long for this day, finally, it's here!
anger (3)    --> 0.82
joy (1)      --> 0.14
fear (4)     --> 0.02
-----
```

The results are decent, but the model is far from perfect. The last sample, “I’ve been waiting so long for this day, finally, it’s here!” should not have the emotion classified as “Anger”.

3.1.3 LLM (GPT3.5 Turbo)

For the third method of Emotion Classification, GPT3.5 Turbo was used.

```
@tool
def classify_emotion(text: str) -> str:
    """Returns the emotion of a text content it is given.
    Use this for any questions that is about the detection or classification of a piece of text content.
    """

    detected_emotion = None
    if CLASSIFY_EMOTION_METHOD == "Logistic Regression (ML)":
        endpoint = "http://127.0.0.1:3000/classifyemotion"
        response = requests.post(endpoint,
                                headers={
                                    "accept": "text/plain",
                                    "content-type": "text/plain"
                                },
                                data=text)
        detected_emotion = response.text
    elif CLASSIFY_EMOTION_METHOD == "DistilBERT (NN)":
        preds = emotion_classifier_distilBERT(text, return_all_scores=True)

        top_k_results = distilBERT_classify_emotion(distilBERT_CLASS_INT2STR,
                                                    preds[0],
                                                    top_k=1,
                                                    desc=True,
                                                    verbose=False)

        for k in top_k_results:
            # print(f"{k.label_name} ({k.label_num}) \t--> \t{round(k.label_score, 2)}")
            return k.label_name
    elif CLASSIFY_EMOTION_METHOD == "GPT3.5 Turbo (LLM)":
        prompt = f"classify the emotion this content is written in:\n{text}."
        prompt += "\nRemember, ONLY return the emotion you classified and nothing else."
        detected_emotion = LLM(prompt)

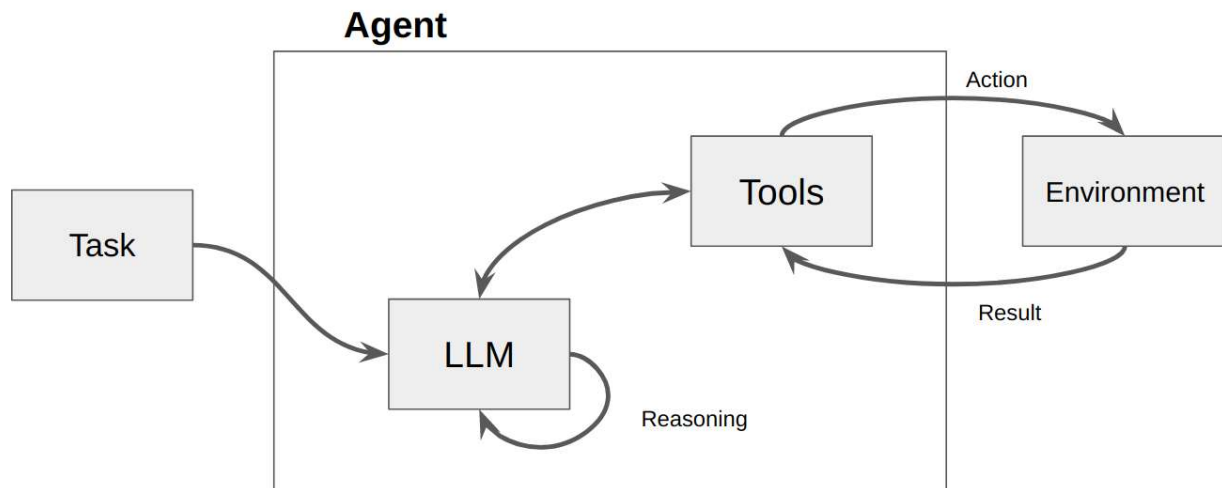
    return detected_emotion.strip().lower()
```

It turns out that the performance is quite good and it's actually quite simple to implement. (although there is a cost for every API call to OpenAI).

However, at the end of the day, the LLM is a Large Language Model that takes in text as input.

If we wish to include other modalities like the user's facial expressions or speech, we still require other Classical Machine Learning algorithms or Neural Networks to perform classification of image and audio data.

3.2 Intelligent Agent (Orchestrator) - ReAct Framework

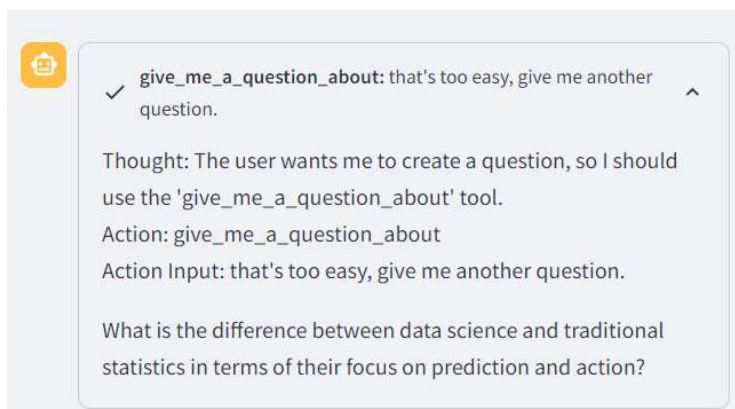


Source: <https://betterprogramming.pub/implement-generative-agent-with-local-llm-guidance-and-langchain-full-features-fa57655f3de1>

In the ReAct Framework, the Agent is given a task and the Large Language Model (LLM) functions as the reasoning engine which creates a plan of action and utilizes the tools that it has been provided with to complete the task. At each time step, the Agent takes an action with a tool and observes the result. The reasoning engine then evaluates the observation and determines the next tool to use and input to the tool. This process continues until the reasoning engine determines that it has obtained the final answer and stops. However, in practice, the current implementation of ReAct doesn't always work well, the Agent can get into infinite loops or make decisions to use tools that do not exist or hallucinate and create input that is not what the user typed in the chat textbox.

For this proof-of-concept, there are various Langchain parameters that were set, python code and prompt guardrails that were created to minimise this from occurring.

Below we see the ReAct framework in action in the Emotion Aware Quiz Bot.



Thought -> Action -> Action Input -> Observation -> Loop

4 SYSTEM ARCHITECTURE

The Emotion Aware Quiz Agent proof-of-concept is created using Python, Langchain, Sklearn, Pytorch, Huggingface DistilBERT, GPT 3.5 Turbo and Streamlit.

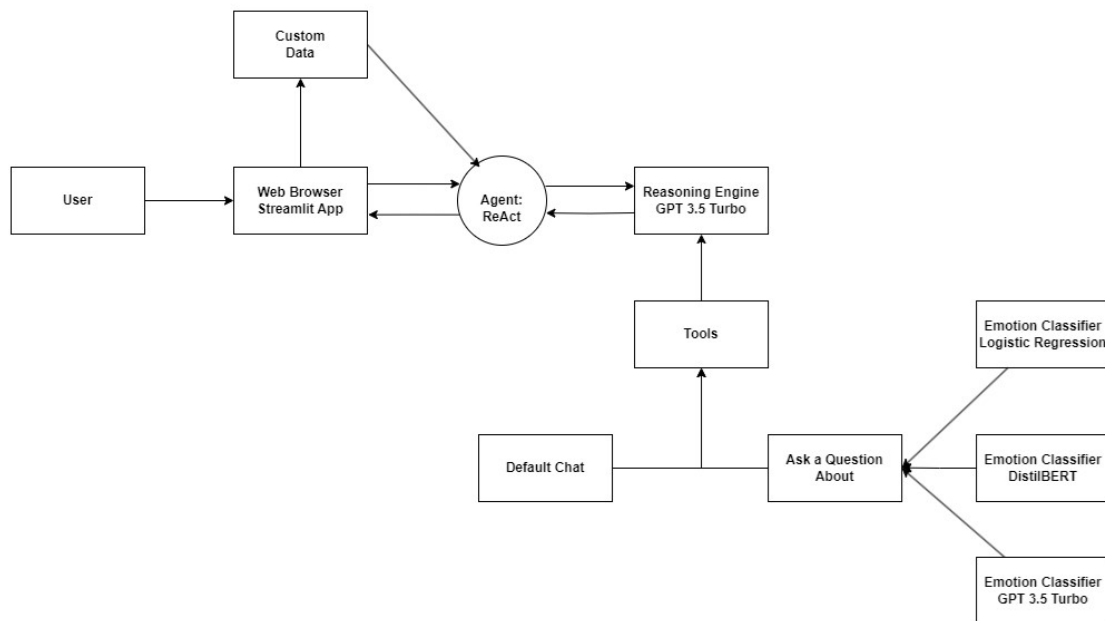


Figure: Emotion Aware Quiz Agent System Diagram

The user interacts with the Agent via a Streamlit App using text input. The Agent uses the input and decides on what tool to use. If the user is not asking the Agent to give it a question, then the ‘Default Chat’ tool will be used.

If the “Ask a Question about” tool is used, the Agent will check if custom data has been loaded, if so, it will be used in question generation. If not, a random topic will be used for question generation.

In generating the question, the Agent will assess the user’s emotional state and then create a question that is either at a beginner level, mid-level or advanced level. The objective being not to have the user feeling overwhelmed by questions that are beyond their ability to answer.

5 LIMITATIONS

The current agent implementation is sufficient for a proof-of-concept but is brittle and not suitable for use in production.

6.0 CONCLUSION

This Proof-of-Concept is but the first step towards developing AI Agents that can personalise training content, the manner in which the course is taught and also how learners are tested, based on their emotional state during the course.

The intent is to improve on the Intelligent Agent in subsequent Practice Modules to achieve the vision presented in the original proposal paper.

7.0 IMPROVEMENTS

Future enhancements include:

- Multi-modal Emotion Classification
 - o Live stream of user
 - o Speech of the user
- Modification of test questions not only based on user emotional state but also:
 - o The number of times in a row that the user answered correctly.
 - o How close the user was in the answer when it was wrong.
- Improve the definition of what is an advanced/hard or beginner/easy question.
 - o Emotion classification using Logistic Regression and DisilBERT is pretty good, while the performance by the LLM (GPT3.5 Turbo) in this task is exceptionally good, and the agent does vary the difficulty of the generated questions based on the detected user emotion.
 - o However, what the LLM reasoning engine considers to be hard or easy may not always be what most people will agree with.
 - o It will require better usage of multi-shot prompting and fine-tuning of the base model. The challenge is whether it can be generalised or if this must be done specifically for each domain. Further research and experimentation is required.

- Improved Intelligent Agent
 - Less prone to hallucination:
 - Currently control is via temperature and prompt engineering, future enhancement would include fine-tuning of the foundational large language model.
 - Better ability to follow plans:
 - The agent can sometimes go off-tangent completely and end up in a infinite loop.
 - Future improvements would include fine-tuning of model and code enhancements.
- More methods of Custom Data Ingestion and file formats
 - Internet content
 - Database content
 - PDF
 - Excel
- Automation Features
 - Sending Notifications
 - Conducting Test and Quizzes
 - Creating and Storing Training Content e.g.
 - Lesson Plans, Flow Charts etc.

8.0 BIBLIOGRAPHY & APPENDIX

APPENDIX 1 - Mapped System Functionalities

- https://github.com/atsui888/Pattern-Recognition-Systems/tree/main/Project_Report/03%20Other%20Docs

APPENDIX 2 – Project Proposal

- https://github.com/atsui888/Pattern-Recognition-Systems/tree/main/Project_Report/03%20Other%20Docs

APPENDIX 3 – Installation and User Guide, System Code

- https://github.com/atsui888/Pattern-Recognition-Systems/tree/main/Project_Report/03%20Other%20Docs

APPENDIX 4 – System Code

- <https://github.com/atsui888/Pattern-Recognition-Systems/tree/main/Code>