Facilitating Adaptive Teaching: Emotion Recognition for Real-Time Instructor Feedback in Augmented Reality

JENNY PENG and BRIAN GUO, University of Washington

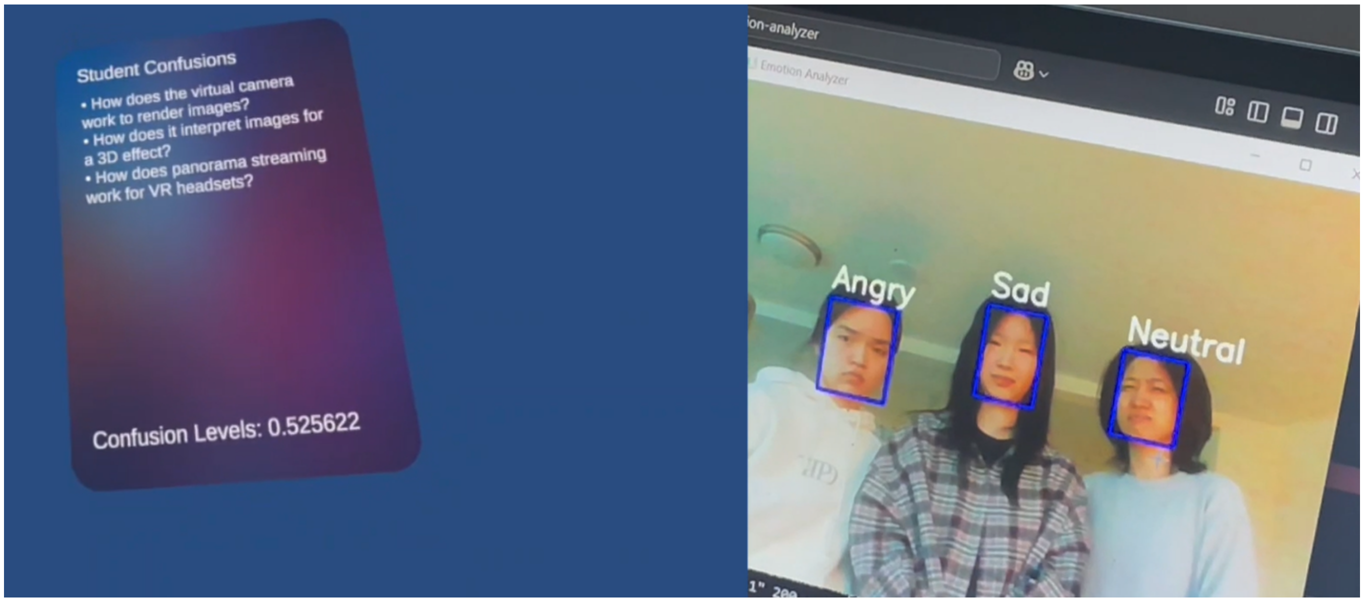
Student Confusions
• How does the virtual camera work to render images?
• How does it interpret images for a 3D effect?
• How does panorama streaming work for VR headsets?
Confusion Levels: 0.525622
Emotion Analyzer
Angry
Sad
Neutral

Fig. 1. Real-time emotion classification to detect spikes in student confusion during lecture, triggering a context-aware LLM pipeline to provide relevant instructor feedback. Github repository: https://github.com/JennyPng/AR-student-emotion-analyzer

Understanding where students struggle is crucial in providing a good educational experience, yet many students suffer in silence making it difficult for instructors to adjust. We propose a context-aware augmented reality (AR) system that utilizes real-time computer vision to analyze student engagement and understanding, dynamically providing feedback on the instructor's teaching. The system aims to enable instructors to efficiently gauge overall student comprehension, even in larger lecture hall settings.

1 INTRODUCTION

Augmented reality (AR) has great potential to enhance educational experiences due to its immersive qualities. Solutions exist for utilizing AR to enhance interactivity and learning for students, but there has been less focus on augmenting the instructor's experience.

It is particularly challenging for instructors, in larger group settings, to assess whether the majority of students are truly understanding the material. Students who are confused or struggling may not feel comfortable speaking up, and methods such as verbal feedback or written assessments, are too delayed to inform real-time adjustments. This creates a gap where instructors lack the immediate situational awareness needed to tailor their teaching in the moment. This serves as motivation for our system, which uses computer vision to monitor the emotion levels of multiple students to analyze student engagement, and provide live LLM-generated feedback to the instructor. It is useful for instructors to have an accurate overview of students' emotional states to improve their teaching [8].

Both emotion recognition and context-aware AR has many accessibility use cases as well, such as for enhancing learning for students with intellectual disabilities [5]. The system's ability to provide real-time emotion-informed feedback can help instructors adjust the pace and complexity of their lessons to accommodate diverse learning needs, enhancing inclusivity in the classroom. The system has the potential to support both general educators and specialized learning environments by improving real-time awareness of student engagement.

1.1 Contributions

- We contribute a real-time pipeline integrating face detection, emotion recognition, and speech recognition to map emotions to speech content.
- We contribute a data-driven framework that identifies spikes in student confusion by establishing baseline emotion levels and dynamically detecting deviations.
- We contribute a novel system to provide instructors with real-time feedback on their teaching in augmented reality.

Authors' address: Jenny Peng, jennypng@cs.washington.edu; Brian Guo, guobri@cs.washington.edu, University of Washington.

2 RELATED WORK

Prior solutions have explored integrating emotion detection with education [8], but these works tend to focus on deep learning techniques for improving emotion classification performance. Furthermore, these solutions focus on different contexts like E-learning, and are not intended for live teaching with a wearable, augmented reality headset.

Previous research has also explored the benefits of augmented reality in education. Studies have found that the immersiveness of augmented reality uniquely benefits specific educational contexts and enrich student understanding [4]. Again, literature tends to focus on how students interacting with AR environments can benefit learning, but not enhance the instructor's teaching experience [7].

Real-time speech recognition has also been utilized in lecture settings. Teachers have reported that reviewing transcriptions of their lectures have helped them reflect on their teaching quality [6], but these were not real-time solutions.

Our approach is unique by integrating real-time emotion detection and speech-to-text for usage specifically in augmented reality during live lecture.

3 METHOD

Our method focuses on building a fully functional system that provides useful information to instructors as they teach a group of students. The pipeline consists of 4 major sections: emotion detection, speech recognition, response generation/streaming, and the user interface in augmented reality.

A. Emotion Detection

The core element of our application involves using computer vision to analyze the emotions of multiple detected faces. The system is developed for usage on a Meta Quest 2, but the headset cameras on commercial models are inaccessible for development. We opted to use our laptop webcam due to time and budget constraints, as the resolution and field of view were sufficient for testing the key functionality.

We utilized YOLOv11-face [3] for real-time face detection, which is an object detection model finetuned on WIDERFace. While there exist other models like DeepFace or Haar-Cascade, YOLO was best suited due to its fast inference speed and accuracy for real-time detection. This stage of the pipeline enables us to identify multiple faces within the live video streamed from the webcam.

To detect when students may be confused with lecture content, we pass all detected faces to a pre-trained emotion classification model [1], specifically a 6-layer convolutional neural network (CNN) trained on FER-2013. Our project timeline made it more practical to use a pre-trained model rather than training our own or finetuning, since we prioritized implementing the entire end-to-end pipeline. The FER-2013 database contains about 30,000 images of various expressions corresponding to seven different emotions: angry, disgusted, fearful, happy, neutral, sad, and surprised. Our system detects confusing lecture topics by calibrating a baseline average of negative emotions from the start of the lecture, and detecting whenever the sampled average spikes to 1.5 standard deviations above the calibrated mean. This value can be adjusted for more or less sensitivity of the application. The emotions we consider as negative are: angry, disgusted, fearful, and sad. We take the mean using the confidence values of the predictions, which allows to more accurately measure emotion levels by accounting for the intensity of the emotion felt.

B. Speech Recognition

To provide contextually relevant feedback on the lecture, our system transcribes and stores the instructor's speech in 7-second intervals. We use Faster-Whisper [2] for speech-to-text, which is a re-implementation of OpenAI's Whisper model with faster inference times. We chose to transcribe in 7-second chunks to reduce the chance of cutting off speech in the middle of a sentence, which may leave out important context that causes an inaccurate transcription, while still retaining granular segments of the lecture.

To be able to correlate lecture content with emotion statistics, we map both the emotion and speech data to the timestamps in which they were recorded. This mapping allows the corresponding lecture content to be retrieved when the system detects a spike in negative emotions.

C. Response Generation/Streaming

In order to provide the instructor with useful and actionable feedback when students experience more negative emotions, we pass the corresponding lecture transcript to OpenAI's gpt-3.5-turbo-instruct LLM. This allows us to analyze what aspects of the recent lecture topics may be confusing to students. Determining how to structure the feedback to be most helpful is subjective, and we were unable to conduct interviews with many educators due to time constraints. From personal teaching experiences, we decided that displaying questions students might have but be too afraid to ask, was the most natural way of encouraging the instructor to consider how they can clarify the current topics. We created a prompt to append the transcript to, which will be shown in the next section.

Since overall lecture context informs potential areas of confusion, we pass in not only the short interval of lecture corresponding to the timestamp of an emotion spike, but also the transcript from the 5 minutes of lecture prior to the spike. 5 minutes was chosen instead of passing in the entire prior transcript, since we found that the longer transcript inputs caused more hallucinations or outdated questions to be generated.

Once we receive the structured instructor feedback, we need to stream it from the Python backend to the Unity client. We initally sought to stream from a laptop to the headset with a TCP connection. However, we faced various networking roadblocks such as firewalls and connection refusals, so we switched to using an HTTP endpoint with a Flask server instead. The LLM-generated feedback, as well as the current average levels of negative emotions, is forwarded to an HTTP endpoint that can be queried by the Unity application.

D. User Interface

We used Unity to implement the AR application, which acts as a client to our server. We query the HTTP endpoint to retrieve the information to display, which includes the generated questions based on the lecture content, as well as the current average negative emotion levels detected.

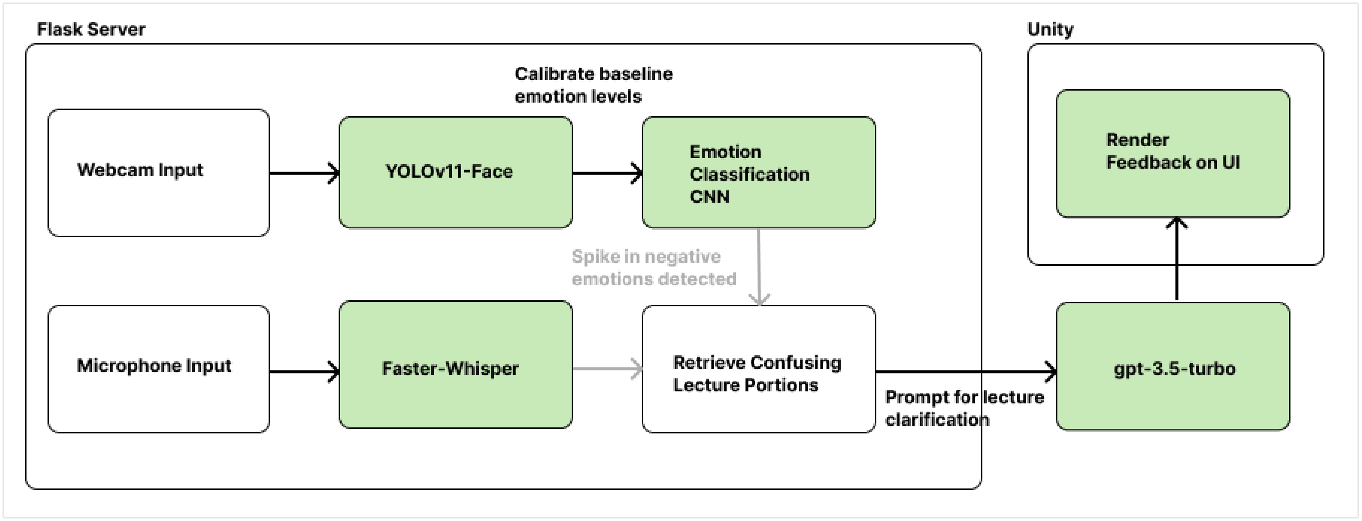
Flask Server
Unity
Calibrate baseline emotion levels
Webcam Input
YOLOv11-Face
Emotion Classification CNN
Render Feedback on UI
Spike in negative emotions detected
Microphone Input
Faster-Whisper
Retrieve Confusing Lecture Portions
Prompt for lecture clarification
gpt-3.5-turbo

Fig. 2. Diagram of system pipeline.

We render the text on a panel that the instructor can move around. This allows it to be unobtrusive and out-of-view during normal instruction, so that the instructor can look for feedback when desired.

4 IMPLEMENTATION DETAILS

Our application is built for the Meta Quest 2, which we chose for its accessibility and pass-through functionality. We used assets from the Meta XR Core SDK and Meta XR Interaction SDK for Unity to implement basic camera movement and interactivity.

We used Python multithreading to concurrently run the emotion detection pipeline with the speech recognition pipeline, storing the data from both threads in separate global Pandas dataframes. The dataframes are indexed by timestamp, allowing us to merge the two data structures and index by time interval.

The emotion detection pipeline uses OpenCV's video capture to get live webcam footage, and a Tensorflow Sequential model loaded with pre-trained weights. The FER-2013 database contains seven emotions: happy, neutral, surprised, sad, fearful, angry, disgust. This does not include confusion, and we initially hoped to conduct some basic experimentation to find which is most closely associated with confusion. However, time constraints forced us to generalize confusion as any of the four negative emotions: sadness, fear, anger, disgust. In order to determine a significant change in emotion, we need to calibrate a baseline for each audience. For demo purposes, our calibration was set to 30 seconds, but for actual use cases 2 minutes may provide better results.

The speech recognition pipeline uses Pyaudio, a Python package for audio I/O, to continuously record and save 1-second chunks of audio. There is one thread for recording audio and adding the audio chunks to a queue, and another thread for transcribing the enqueued audio segments. Whisper includes various model sizes of increasing accuracy (tiny, base, small, medium, large). We used the base Whisper model to balance our CPU usage while maintaining reasonable accuracy on the transcription. Access to more compute would elevate the performance of the project, but only marginally as base was already quite accurate.

For our user interface, we needed to display the feedback to the instructor in a useful format during the lecture, avoiding the possibility of cognitive overload or instead creating a distraction. We designed the following prompt to extract what questions students may naturally have about the content based on the transcript. With one-shot prompting, we request a JSON formatted response from the LLM for ease of parsing individual questions for the Unity client to format. We utilized repetition of requirements and minimally ambiguous instructions to reduce hallucinations.

Our Python backend uses a Flask server with an HTTP endpoint. We also use ngrok to forward our local host onto the internet, allowing Unity to query a specific domain to receive updated values from Python. While this resulted in higher latency for our real-time usecase, it was a simple, satisfactory streaming solution for our use case of building a working pipeline.

# role
You are assisting an instructor during their lecture. You are given the lecture transcript so far.
Students are particularly confused about the most recent topics towards the end of the transcript.

Give 2-5 short, specific and distinct questions students might be having about the lecture, as an array of strings,
to help the instructor understand how to adjust their teaching pace or explanation of topics.

# requirements
You MUST respond in this EXACT JSON format with NO ADDITIONAL WORDS.

Good example response for a lecture about AI and art:
```
{
"confusing_topics": ["Why is generative art unethical?",
"How are generative AI tools trained?"]
}
```
DO NOT just copy the example questions, only put in content relevant to the following lecture transcript.
FOCUS the questions on the last few sentences of the transcript, and only use the rest for context.
Again, you MUST respond in the EXACT json format.

# transcript
This is the lecture transcript: <append the last 5 minutes of the transcript>

5 EVALUATION OF RESULTS

The pre-trained model we use achieves a validation accuracy of around 61% after 50 epochs. Through testing, despite the low webcam resolution, we observed that YOLO can still accurately detect faces even when far away. The CNN also, while not fully accurately distinguishing between more granular classifications like angry or fearful, is sufficiently accurate for classifying between neutral and generally negative emotions.

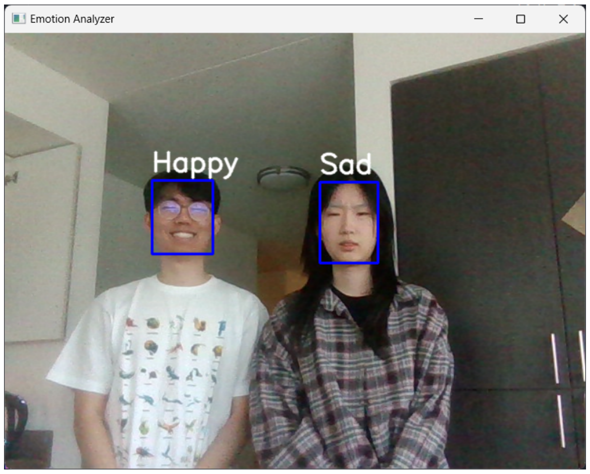

Fig. 3. Real-Time Emotion Classification

Inevitably, more nuanced expressions in a natural environment can pose issues, leading to more "neutral" predictions from the model. However, this is preferable to random predictions that may introduce noise into the system, as we don't consider neutral expressions in our spike detection.

Our lecture transcription was adequate for our purposes, but was somewhat unstable due to various factors. For one, the Whisper model we used was the second smallest due to limitations with our computing power. As a result, the model was quite sensitive to noise and prone to guessing based on its training data, introducing completely unrelated terms occasionally. While this was balanced out as more context was added throughout the transcription, the beginning of the lecture occasionally introduced random responses. Requiring the response in a specific JSON serializable format had a good success rate, but occasional errors would occur where the response was not JSON parseable. Additionally, the accumulation of small inaccuracies from Whisper and GPT's proclivities would occasionally generate responses that critiqued speaker rhetoric instead of focusing on lecture content. Despite this, most of the responses that the Unity application received were related to spoken topics and useful insights.

The networking portion of this project was a major roadblock; we attempted to use a TCP connection to send data from our Python script to our Unity application, but were unable to connect the server and client when running them on different devices. Thus we pivoted to forwarding a local HTTP server onto the internet through ngrok, which requires our Unity application to send continuous GET requests for updated data. While this is computationally inefficient and can introduce latency issues on a larger-scale, this approach lent itself well to the functionality of the system as a whole. Our application was able to quickly receive updates and display them to the user without networking errors.

Finally, our actual UI that the teacher interacts with is quite simple, and most of the results to evaluate this would come down to user preference. Without the resources to test with many users, we settled on the current design as our MVP.

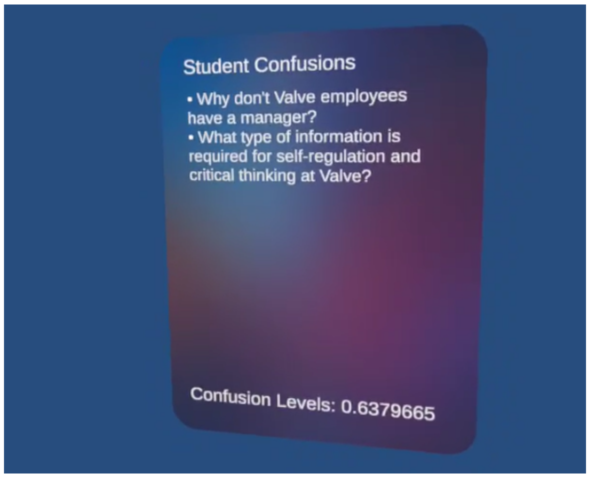
Student Confusions
• Why don't Valve employees have a manager?
• What type of information is required for self-regulation and critical thinking at Valve?
Confusion Levels: 0.6379665

Fig. 4. Unity Application View

6 DISCUSSION OF BENEFITS AND LIMITATIONS

The benefits of our system were that it successfully generated relevant questions based on the speech content, helping the user consider parts of their speech that may have been confusing or uncohesive. It runs in real-time with live updates to the Unity application that have low enough latency to enable quicker adjustments during live teaching scenarios.

For limitations, our webcam resolution was low and we did not test with large classroom settings, so we have not verified how the system would perform in a large classroom or lecture hall. However, these could be addressed with using a higher resolution camera. The performance has slight lag, but we attribute this to the CPU and this can be improved with better hardware. We display only one format of feedback, and we can refine our prompting for displaying better feedback. We could also expand our system to be more multimodal by incorporating lecture slides or other supplementary materials.

Another limitation was the accuracy of the recorded transcripts. Because we use a larger context window when passing lecture content to GPT, errors earlier in the transcript would persist and cause some inaccuracies in the LLM's interpretation of the lecture context. To improve this, we could periodically correct the older transcript based on new context given.

7 FUTURE WORK

One area to look into involves ease of use in an actual teaching environment. The aesthetic quality of headsets impacts the adoption of XR technologies. Currently, it would not be feasible for lecturers to wear headsets like the Meta Quest 2 during class. Developing less obtrusive hardware would greatly benefit the use cases of our application.

Another factor to consider is how we classified confusion. We generalized confusion as four of the seven emotions present in the FER-2013 database. More accurate classification of confused expressions could lead to more accurate insights.

Finally, small details such as increasing computing resources and fine-tuning models for lecture setting would all lead to better performance.

8 CONCLUSION

We presented a novel real-time pipeline that integrates facial emotion recognition, speech-to-text transcription, and augmented reality to provide instructors with immediate feedback during live lectures. Our system processes live video input to detect emotional responses from students, maps these responses to lecture content using real-time transcription, and delivers insights through an AR interface to help instructors adjust their teaching strategies dynamically.

Unlike previous approaches that focus on post-lecture analysis or student-facing AR applications, our system enhances the instructor's teaching experience by offering actionable feedback in real time. We believe that this integrated, instructor-focused AR feedback system has the potential to enhance teaching effectiveness and student engagement in modern learning environments.

ACKNOWLEDGMENTS

Thanks to Douglas Lanman for being a great professor and teaching this class. Thanks to John Akers for providing the headset.

REFERENCES

[1] 2022. Emotion Detection. https://github.com/atulapra/Emotion-detection. Accessed: 2025-03-19.
[2] 2024. Whisper Live Transcription. https://github.com/gaborvecsei/whisper-live-transcription. Accessed: 2025-03-19.
[3] 2025. Yolo Face. https://github.com/akanametov/yolo-face. Accessed: 2025-03-19.
[4] Georgios Lampropoulos, Euclid Keramopoulos, Konstantinos Diamantaras, and Georgios Evangelidis. 2022. Augmented Reality and Virtual Reality in Education: Public Perspectives, Sentiments, Attitudes, and Discourses. *Education Sciences* 12, 11 (2022). https://doi.org/10.3390/educsci12110798
[5] Christopher M. Reardon, Rachel Wright, David F. Cihak, and Lynne E. Parker. 2016. Intelligent Context-Aware Augmented Reality to Teach Students with Intellectual and Developmental Disabilities. In *The Florida AI Research Society.* https://api.semanticscholar.org/CorpusID:6117149
[6] M. Wald. 2005. Using Automatic Speech Recognition to Enhance Education for All Students: Turning a Vision into Reality. In *Proceedings Frontiers in Education 35th Annual Conference.* S3G–S3G. https://doi.org/10.1109/FIE.2005.1612286
[7] Hsin-Kai Wu, Silvia Wen-Yu Lee, Hsin-Yi Chang, and Jyh-Chong Liang. 2013. Current status, opportunities and challenges of augmented reality in education. *Computers Education* 62 (2013), 41–49. https://doi.org/10.1016/j.compedu.2012.10.024
[8] Jiangqin Xu, Zhongqiang Huang, Minghui Shi, and Min Jiang. 2018. Emotion Detection in E-learning Using Expectation-Maximization Deep Spatial-Temporal Inference Network. In *Advances in Computational Intelligence Systems*, Fei Chao, Steven Schockaert, and Qingfu Zhang (Eds.). Springer International Publishing, Cham, 245–252.