We need to monitor the model performance in an ASR system to ensure that transcriptions are accurate and reliable over time, especially if data patterns or user needs evolve. Loosely speaking, we can monitor both the model's input (data quality) and output (performance metrics tracking) to detect drift, and incorporate this in a feedback loop for continuous improvement.

1. Data Quality

First, we need to monitor the quality of the audio files before they are fed into the ASR model. This involves automated scripts that check various aspects, such as SNR, sampling rate or audio length.

Audio files that fall below set thresholds will be flagged out. We can then either automatically reject and log the files for review, or pass them with a warning flag that indicates potential issues with the transcription accuracy.

2. Performance Metrics Tracking

Next, we need to track the model's performance over time using metrics such as word error rate. This can be challenging in a production setting without labelled data for transcription. There are several strategies we can employ, each with trade-offs:

- Manual Review and Annotation

Periodically select a subset of transcriptions for manual review and annotation. While the most labor-intensive, it provides a gold standard for model evaluation over time.

- User Feedback Collection

Implement mechanisms that allow end-users to flag errors or rate the accuracy of transcriptions. The user-generated data may be subjective, but it is more labor-efficient and still highlights areas where the model may be underperforming.

- Automated Quality Checks / Post-Processing Analysis

Utilize automated quality checks that do not require labelled data, such as NLP tools to analyse the transcripts for incoherence or grammatical consistency.

An alert system notifies when performance metrics exceed control limits.

3. Tracking Model Drift

Data drift occurs when the statistical properties of the input data change over time, due to reasons like accents, speech patterns or background noise types.

Concept drift is different from data drift in that the input data may look the same statistically, but the way it should be interpreted changes. This is less relevant for HTX, but examples could

include new terminology or changes in language structure that the model was not trained to recognize.

To detect drift, we can use statistical methods by establishing control limits (upper and lower bounds) for data quality and performance metrics to identify potential anomalies.

To correct for drift, we can update the model with new and diverse training data. To decide whether to update the model, we can use A/B testing by deploying the new model version alongside the existing one and compare their transcriptions on the same audio inputs. This helps in identifying if the new version improves performance or not.

Note that we can also refine the thresholds and preprocessing techniques used to monitor data quality based on the transcription results so that we optimize the balance between the two. For example, if we find that the model is robust to audio length or background noise, we can relax the controls and allow end-users to upload longer or noisier files too.