# **Evaluating Open-Source Voice Emotion Recognition Models for Real-Time Coaching**

**SUMMARY:**

After evaluating three open-source emotion recognition models and comparing them to Hume AI, I recommend:

* **Best Overall**: **Emolysis** – rich taxonomy, real-time, deployable locally in the UAE
* **Best for Quick Integration**: **MVP** – plug-and-play, solid performance, easy Docker setup
* **Lightweight Option**: **MixedEmotions** – simple, efficient, but lacks discrete emotion output
* **Hume AI** offers unmatched emotion nuance but isn't UAE-deployment friendly due to cloud-only access

**Conclusion**: For real-time emotion coaching in UAE-based sales scenarios, **Emolysis is the most robust open-source choice**, balancing accuracy, nuance, and on-prem deployment capabilities.

See full comparison below ↓:

## **Emotion Granularity and Taxonomy**

**MVP Emotion Recognition (Husseinshtia1)** - The MVP project is a multi-modal emotion recognition dashboard supporting text, audio, and video inputs. It aims to detect a range of human emotions in real time, though the exact taxonomy of emotions is not explicitly documented in the repository. Based on common speech emotion datasets, it likely recognizes **basic emotions** (e.g. happiness, sadness, anger, fear, etc.) and possibly a neutral state. Its multi-modal design suggests it may integrate vocal tone with facial expressions or text sentiment for a more nuanced interpretation, but its **granularity** appears to focus on the standard discrete emotion categories rather than a very fine-grained taxonomy.

**MixedEmotions** - The MixedEmotions toolbox takes a different approach, using a **dimensional model** for voice emotions. Instead of outputting discrete categories, the open-source audio module predicts an emotion in terms of **continuous Arousal and Valence values**. This two-dimensional representation can describe emotions as points in a continuous space (e.g. high arousal + negative valence might correspond to anger, low arousal + positive valence to calm happiness). While this allows a spectrum of emotional nuance, it does **not classify named emotions** like “happy” or “sad” directly. The taxonomy is thus less explicit - essentially any emotion must be inferred from the arousal-valence coordinates. MixedEmotions did have other modules (some proprietary) for categorical emotion recognition (e.g. a commercial module by Phonexia for detecting discrete emotions in audio), but the open audio module itself only provides the valence/arousal dimensions. This makes its granularity more **coarse** in terms of labeled categories (just two continuous scales) even though in theory it can represent many nuanced blends of emotion.

**Emolysis** - Emolysis offers the most **comprehensive taxonomy** among the open-source options. It predicts both **discrete categorical emotions and continuous dimensions** for group-level emotion analysis. Emolysis’s label space is based on **Plutchik’s eight primary emotions** (joy, trust, fear, surprise, sadness, anticipation, anger, disgust) plus a “no emotion”/neutral class. Notably, it treats emotion prediction as a **multi-label** problem, acknowledging that multiple emotions can co-occur in human expression. In addition, Emolysis outputs **valence and arousal scores** (on a 0-1000 scale) for a fine-grained continuous measure of emotional tone. This dual approach means Emolysis can recognize a nuanced emotional state (for example, a person might be both *surprised* and *happy* with high arousal) and also quantify the overall positivity/negativity and energy of the voice. Its taxonomy is more nuanced than basic six-emotion models - by including categories like **trust** or **anticipation**, it covers subtle social-affective states that are highly relevant (e.g. an agent’s tone conveying confidence/trust vs. a customer’s tone showing anticipation or hesitation).

**Hume AI Baseline** - Hume AI’s system (Empathic Voice Interface) is built on the premise that emotion is **high-dimensional**. In public research, Hume’s models recognize **many more emotion types** beyond the classical set. In fact, Hume’s speech prosody model covers at least **15-20 distinct emotions**, including nuanced states such as *Amusement, Awkwardness, Boredom, Calmness, Confusion, Pride, Tiredness*, etc., in addition to basic emotions. This far exceeds the granularity of the open-source models. Hume’s taxonomy is both **broad and fine-grained**, reflecting decades of research that identified rich emotional nuances in voice. In essence, compared to Hume’s ~18+ emotion categories (with continuous intensity scores for each), the open-source models are more limited - Emolysis comes closest with 8+ categories, while MVP likely handles a basic set, and MixedEmotions only provides two continuous dimensions.

*Takeaway:* In terms of emotion granularity, **Hume AI** offers the richest and most nuanced set of emotions (dozens of categories and dimensions). Among the open models, **Emolysis** has the most nuanced taxonomy (8 key emotions + valence/arousal), whereas **MVP** likely covers the common discrete emotions and **MixedEmotions** reduces emotion to valence-arousal coordinates. The appropriate granularity depends on the use-case: for sales coaching, detecting a variety of relevant emotional cues (beyond just positive/negative) is valuable - e.g. recognizing *confusion vs. anger vs. excitement* - which suggests Emolysis’s or Hume’s broader taxonomy would be advantageous over a simple valence/arousal metric.

**Granularity Comparison Table:**

| **Model** | **Emotion Taxonomy** | **Distinct Emotions** |
| --- | --- | --- |
| **MVP (Husseinshtia1)** | Discrete basic emotions (audio, video, text); exact set not specified, but likely standard categories (happy, sad, angry, etc.) | (Not explicitly documented; likely ~5-7 core emotions + neutral) |
| **MixedEmotions** | Dimensional model for audio: outputs continuous **Valence** (positive/negative) and **Arousal** (calm/excited). No direct discrete labels from voice. | N/A (Uses 2 continuous dimensions instead of fixed categories) |
| **Emolysis** | Multi-label **Plutchik 8** primary emotions (joy, trust, fear, surprise, sadness, anticipation, anger, disgust) **+ “no emotion”**, **plus** continuous valence & arousal scores. | 9 discrete classes (8 emotions + neutral), with simultaneous multiple emotions possible; also 2 continuous dimensions. |
| **Hume AI (baseline)** | Fine-grained proprietary taxonomy; measures **~18+ nuanced emotions** from voice (e.g. *Amusement, Calmness, Pride, Confusion, etc.*), modeled in a high-dimensional continuous space. | 15-25+ emotion types (with continuous intensity scores for each; far more granular than open-source models) |

## **Latency and Real-Time Performance**

Real-time performance is critical for live sales coaching, where feedback on a speaker’s emotion must be delivered with minimal delay. All three open-source models under review emphasize real-time or near-real-time operation, but their latency characteristics differ based on model complexity and design:

* **MVP Emotion Recognition:** The MVP system is explicitly designed for **real-time analysis** across modalities. It is described as an “emotion recognition dashboard” suitable for edge devices, implying that it can process audio streams with low latency on local hardware. In practice, if MVP uses a deep learning model for speech (e.g. a transformer or CNN on audio), one can expect a slight buffering (perhaps processing audio in short windows of a second or less), but its architecture suggests it is optimized for responsiveness. The inclusion of **edge device support** indicates the model can run with reasonably low latency on-premise hardware (possibly even on high-end CPUs or modest GPUs) without needing a round-trip to a server. Thus, MVP should be capable of **low-latency inference**, suitable for live feedback while a sales call is ongoing.
* **MixedEmotions:** The MixedEmotions audio module was developed with a RESTful service in mind and is relatively lightweight. It likely uses **feature extraction (e.g. openSMILE descriptors) plus a classical model (SVM/regression)** to output valence and arousal. Such an approach is computationally inexpensive - feature extraction can be done in real-time as audio streams in, and the model itself (an SVM) is fast on a CPU. MixedEmotions was in fact demonstrated in a **call center monitoring context** to analyze client mood on each call. Given that context, we can infer it handled call audio in (near) real-time. The latency for valence/arousal estimation would be very low (on the order of milliseconds after each analysis frame), making MixedEmotions **highly usable for live coaching**. The potential drawback is that it may require a short segment of audio (e.g. a few seconds) to accumulate a stable estimate of valence/arousal. However, overall it is suitable for **low-latency scenarios**, and its simplicity helps ensure prompt responses.
* **Emolysis:** Emolysis processes **multimodal inputs in “nearly real-time”** according to its authors. It performs synchronization of video, audio (and even text) streams and outputs group-level emotion continuously. The pipeline involves deep learning models (for face emotion, speech, etc.), which are more computationally intensive than the MixedEmotions approach. Nevertheless, Emolysis has been optimized to run as an interactive toolkit, achieving close to real-time performance with a capable machine. In a live scenario, one might experience a minor lag (perhaps on the order of a second or two) due to video frame processing and the fusion of modalities. The term "nearly real-time" suggests there may be a small buffering delay to ensure audio and video are synchronized before emotion inference. Emolysis supports deployment on mobile and desktop (Android, iOS, Windows) which implies that with appropriate model compression or hardware (mobile GPUs), it can maintain reasonable speed. However, for **best performance and minimal latency**, running Emolysis on a machine with a GPU is recommended, especially if processing audio *and* video in parallel. For audio-only use (as in a phone sales call), the load is lighter - one can disable the video pipeline and just run the audio model, which would improve latency. In summary, Emolysis can deliver **real-time or near-real-time emotion updates**, but it may introduce slightly more latency than simpler models due to its complexity (still likely within acceptable bounds for live coaching).
* **Hume AI:** Hume’s Empathic Voice API (EVI) is designed for **real-time emotion recognition from voice** as well. In terms of raw processing, Hume’s models (being highly optimized and running on cloud servers with GPUs) can analyze audio streams with low latency. The major consideration is network latency - using Hume involves sending audio data to the API and receiving predictions. For a live call in the UAE, this round-trip could introduce delays depending on server location and bandwidth (e.g. a 200ms-500ms network latency each way can add up). Hume likely processes streaming audio in small chunks to mitigate delay, but there is inherently at least some additional latency beyond on-prem solutions. In a best-case scenario, Hume’s end-to-end latency might be on the order of a second or less, which could be acceptable for coaching (the feedback would slightly lag the speaker’s utterances). However, if network conditions are poor or if data needs to be buffered for quality, latency could increase. The **trade-off** is that Hume offloads computation to the cloud, so even a low-power client can get results quickly, provided the connection is fast. Overall, Hume’s system is **real-time capable**, but one must account for any network-induced lag in a live setting.

**Summary:** All models under review can function in or near real-time, but their latency profiles differ. **MixedEmotions** is extremely fast (lightweight CPU inference), suitable for instantaneous feedback, but only provides coarse signals. **MVP** and **Emolysis** use deeper models; with proper hardware, they still achieve low latency, though Emolysis in particular may have a slight delay due to multimodal processing. **Hume AI**, while powerful and fast in processing, introduces network dependency which could add latency in a UAE deployment. For **live sales coaching**, a locally-run model (MVP, MixedEmotions, or Emolysis) avoids network delays entirely, which is a strong advantage in ensuring **consistently low latency feedback** during calls.

## **Reliability and Accuracy of Emotion Detection**

**MVP (Husseinshtia1):** The MVP project does not publish specific accuracy metrics or benchmarks, so we must infer reliability from its design and training methodology. It advertises “cutting-edge machine learning and deep learning models”, which suggests it may leverage state-of-the-art pretrained models (for example, wav2vec 2.0 or similar for speech, and transformers for text). If the author fine-tuned these models on standard emotion corpora, the accuracy would be in line with those benchmarks - typically, modern speech emotion recognition on acted datasets (like RAVDESS or IEMOCAP) can reach 70-80% accuracy on 6-8 class classification. However, real-world reliability depends on data quality: many academic datasets feature acted, exaggerated emotions, whereas sales calls involve more **subtle, spontaneous expressions**. Without explicit info, it’s likely MVP’s speech model was trained or tested on common datasets (perhaps IEMOCAP or CREMA-D). Its reliability in terms of aligning with human interpretation is uncertain; it might detect clear emotional cues well (anger vs. neutral, etc.), but subtle tones (e.g. customer skepticism or salesperson stress) might be harder. On the positive side, MVP’s emphasis on **explainable AI and federated learning** hints that it aspires to transparency and possibly continuous improvement. For example, explainability could mean it provides confidence scores or rationales for detected emotions, and federated learning could eventually allow training on sensitive call data without centralizing it - which would improve accuracy on domain-specific nuances over time. In summary, MVP is an **early-stage open project**; it likely achieves reasonable baseline accuracy on basic emotions, but its real-world reliability would need to be validated. There is no published error rate or validation beyond the developer’s claims.

**MixedEmotions:** MixedEmotions’ audio module relies on classical ML, and its reliability can be viewed through the lens of known results in valence-arousal prediction. Typically, **valence and arousal regression** from speech has moderate accuracy - for instance, models might achieve correlation scores in the 0.5-0.7 range with human ratings for each dimension (in some studies). The MixedEmotions audio module likely uses an established feature set (the presence of an openSMILE license in the repo suggests use of that toolkit for acoustic features) and a trained regressor or classifier for valence/arousal. The data used isn’t explicitly stated in the documentation; it could have been trained on a dataset like **RECOLA or SEMAINE** (which have continuous emotion annotations) or possibly the audio from the Affectiva or DEAP corpora. Given the EU project context, the model might not be cutting-edge by today’s deep learning standards, so we should expect **moderate accuracy**. In practical terms, it can correctly gauge the *general mood* (positive vs negative, energetic vs calm) of a speaker’s voice, but it may not be very **fine-tuned to specific emotion categories**. For example, it might output a slightly positive valence for both “happy” and “calm” voices (though those are different emotions) because both feel generally positive. Alignment with human interpretation is reasonable for broad affective tone - e.g. a highly frustrated customer will show as high arousal, very low valence, which a coach can interpret as anger or upset. However, MixedEmotions might **miss specific context** - it won’t explicitly tell if the emotion is anger or fear if both map to similar valence/arousal values. The reliability also depends on noise: telephone audio quality, cross-talk, etc. could impact the feature extraction. Since it’s not deep learning-based, it might be less adaptive to varied conditions. Nonetheless, MixedEmotions was **tested in real-world scenarios** (call center monitoring) and found useful enough to integrate, indicating its reliability was acceptable for gauging customer mood trends. In short, MixedEmotions provides **stable but coarse-grained accuracy** - good for detecting positive vs negative tone shifts, but not for nuanced emotion classification.

**Emolysis:** Emolysis is an academic toolkit with models trained on large, in-the-wild datasets, which suggests relatively **strong accuracy** for its components. It leverages datasets like **AffectNet** (faces) and **CMU-MOSEI** (multimodal emotion in video) for training. These datasets are extensive (AffectNet has 0.4 million labeled facial images; MOSEI has thousands of annotated video clips) and capture a variety of spontaneous expressions. Emolysis uses those to produce a unified prediction; for voice specifically, MOSEI provides labels for multiple emotion categories per utterance, which Emolysis maps into its 8-class system. Because it is multi-label, Emolysis can capture complex emotional mixes more faithfully - this can actually improve reliability in conversation settings where people often feel more than one emotion (e.g., nervous and hopeful). The authors note that they mapped different label spaces to a common one, which implies careful calibration of the models. They unfortunately do not report a single accuracy number; instead, they did a **qualitative evaluation with independent reviewers**, who found the system’s outputs “promising” on test videos. This qualitative validation (10 reviewers) suggests the model’s output generally made sense to human observers in a variety of group situations. We can infer that Emolysis performs well in distinguishing broad categories (e.g. joy vs. sadness vs. anger) given the robust training data. It likely also tracks valence and arousal accurately (within a small error margin) since those come from regression on continuous annotations. One challenge could be **group emotion fusion** - Emolysis has to aggregate emotions of multiple people. If used for a single speaker (like one salesperson), this is simpler and may even be more accurate since group-level confusion is removed. So in a one-on-one call, Emolysis could focus on the agent and/or customer individually (the toolkit allows selecting target persons), which should yield reliable personal emotion readings. Overall, Emolysis is **state-of-the-art for open-source**: its reliability benefits from modern deep learning and large data. It should align fairly well with human judgments (e.g., it knows the difference between *angry yelling* and *excited yelling* by using both valence and category cues). However, as a complex system, there is some risk of misclassification, especially for emotions like *anticipation* or *trust* which are subtle and context-dependent. Those might be harder to detect from voice alone. Emolysis mitigates this by also analyzing text (if transcripts are available) to improve accuracy. In summary, Emolysis’ accuracy is **strong** on core emotions and continuous measures, and reasonably good on nuanced emotions, making it quite reliable for coaching purposes. The absence of numeric accuracy aside, it is likely on par with contemporary research models.

**Hume AI (Baseline):** Hume’s emotion recognition is built on a very large, **proprietary dataset and advanced model ensemble**, which generally implies excellent reliability. According to Hume, their models are fine-tuned with “scientifically controlled data” and millions of data points, ensuring that the system’s interpretation of emotion is **scientifically validated and robust**. In practice, Hume’s voice model benefits from having learned from a huge variety of vocal expressions (cross-cultural vocal bursts, acted and natural speech, etc.), likely giving it an edge in capturing subtle or rare emotions. We do not have exact accuracy figures publicly, but Hume’s selling point is “measuring emotional expressions with unmatched precision”. One can expect that for a given set of emotion categories, Hume’s model would score at or above state-of-the-art accuracy, simply due to the scale of training. Moreover, Hume’s model can detect emotions that are hard for others to detect (e.g. **awkwardness or guilt** in a voice) - whether it’s 100% accurate is unlikely, but even a reasonable detection of such states is beyond what the open models attempt. Alignment with human interpretation is a core focus: the taxonomy Hume uses was derived from human studies on perception, so if Hume says a voice clip contains “confusion” with 0.8 confidence, it’s likely reflecting acoustic patterns that humans also hear as confusion. In short, Hume’s reliability is presumably **very high** for both common and nuanced emotions, given the investment in data and research. However, it is a closed system, so one must **trust Hume’s evaluation** without the ability to inspect or fine-tune the model. In sensitive deployments like sales coaching, there might be a need to validate the model on specific accents or languages (e.g. Arabic-accented English, if relevant in UAE) - something you could potentially fine-tune in an open model but not with Hume’s closed model. That said, Hume’s broad training likely included diverse speakers, so it should generalize well.

*Overall Reliability:* In comparison, **Hume AI** is expected to be the most **accurate and nuanced**, thanks to extensive training data and research backing. **Emolysis** is the next in line, with strong deep-learning models trained on large datasets - likely achieving solid accuracy on core emotion recognition tasks. **MVP** is a bit of an unknown but presumably uses modern techniques; it might perform decently on obvious emotions but would need domain-specific tuning for optimal results. **MixedEmotions** is the simplest and thus has the highest risk of oversimplifying or missing nuanced emotions; it gives reliable readings for overall sentiment (valence) and excitement (arousal) but not discrete emotion classes. Notably, MixedEmotions and Emolysis both output **continuous valence/arousal**, which often correlates well with human judgments of positivity and intensity. For live sales coaching, those continuous measures can be very useful (e.g. a steadily dropping valence score is a red flag that the customer is becoming unhappy). The categorical outputs (from MVP, Emolysis, Hume) add interpretability - e.g. distinguishing if negative valence is due to *anger* vs. *sadness*. Having multi-label output (Emolysis, Hume) further improves reliability in complex interactions (a single label system might force a choice even if the emotion is mixed). In summary, all these models have trade-offs in reliability, but **Emolysis and Hume** stand out for their robust training and richer outputs, which likely align more closely with real human emotional nuances in conversation.

## **Integration into Real-Time Voice AI Systems**

Integrating an emotion recognition model into a **live sales coaching architecture** involves considering how the model will ingest audio (and possibly other data) from calls, how it will output results in real-time, and how easily it can be managed within the overall system (e.g. as a service or component). Here we compare the integration aspects:

* **MVP Emotion Recognition:** MVP was built as a **complete dashboard and service** - it includes a web application (app.py), monitoring via Grafana, and even cloud deployment scripts. This means it can be integrated either as a **standalone service** that your architecture calls (e.g. via REST API or gRPC, depending on how app.py serves data), or as a library (since it’s open-source, one could import its modules directly). It uses Docker and Kubernetes in its setup, which simplifies integration: you could deploy the MVP model as a containerized microservice within your AI coaching platform. For example, one could have a “voice emotion service” container that receives audio from the call (perhaps via a message queue or streaming protocol) and returns emotion analysis in real-time. The presence of **Terraform and CI/CD scripts** indicates that deploying it on cloud or on-prem infrastructure is straightforward and automated. In a live coaching scenario, MVP’s real-time outputs can be subscribed to by a coaching module that decides on interventions (like flagging if the salesperson sounds frustrated). Since MVP supports text and video too, integration could be extended: e.g. a transcript sentiment service (text emotion) and a webcam feed analysis (if analyzing the salesperson’s facial expressions) - all under the same framework. This unified approach is a plus for integration consistency. The **explainable AI** feature could mean it provides insight (perhaps via attention weights or salient features) that the coaching UI could display (e.g. “customer’s tone indicates anger because voice pitch and intensity increased”). In summary, MVP integration is **highly flexible** - it’s already designed with microservices and containers in mind, and can slot into an existing system either as a backend service or by cherry-picking its code for custom integration. The only caution is that as a relatively new project, documentation may be light; the integrator might need to spend time understanding its internal API. But since it adheres to modern deployment practices, it should be one of the easier ones to integrate and scale.
* **MixedEmotions:** MixedEmotions provides its functionality as a set of **Docker modules with RESTful APIs**. For the audio emotion recognition, there is a Docker image (on DockerHub) for the “up\_emotions\_audio” service. Integration would involve running that container in your environment. The service expects either an audio file upload or a URL to an audio source and returns a JSON (JSON-LD formatted) with the valence and arousal results. For real-time streaming, one would likely need to send short audio clips (for example, a rolling window of the last few seconds of conversation) to the service continuously, and get back the emotional readings. This is a bit less straightforward than a streaming API - you might have to implement a small buffer and call the API in intervals (e.g. every second or two with the latest audio snippet). However, since the calls are lightweight, this is feasible. MixedEmotions was architected to be **modular**, so you could integrate multiple modules (for instance, also use their sentiment analysis or entity extraction if needed) in a pipeline. The advantage for integration is that the heavy lifting of deployment is solved: just run the container and hit the API. It’s also **multilingual** in design (the project targeted multiple languages), though the open audio module was English-focused. The MixedEmotions platform even had an orchestrator for linking modules in workflows, but for our purposes, likely just calling the audio emotion API is enough. In a sales coaching app, you’d integrate MixedEmotions by having a component feed it audio and interpret the valence/arousal output to decide coaching actions (e.g. if customer valence goes very negative, alert the coach or the salesperson). Because MixedEmotions only gives two numbers, that integration logic would be relatively simple (thresholds or trend detection). One thing to note is MixedEmotions is an older project (circa 2018); while it’s stable, it may not have the slick developer experience of newer frameworks. The output being JSON-LD is a bit unusual (it’s essentially JSON with semantic web structured context), but you can parse it as normal JSON for the values. Overall, integration difficulty is **low** - run container, call REST endpoints - but achieving a truly streaming solution might require a custom loop. In terms of compatibility, MixedEmotions being open-source under GPL-3.0 means you have full control to modify it if needed (for instance, you could alter the service to accept raw audio stream if you have the skills).
* **Emolysis:** Emolysis is provided as a **standalone toolkit with a GUI**, but it is also open-source Python code and comes with a Docker image for deployment. Integration can happen in a couple of ways. The simplest might be to use Emolysis as a reference or library: since it’s modular, one could integrate its **models and inference code** directly into your application (bypassing the GUI). For example, you could import the Emolysis project in Python and call its functions to analyze an audio stream. This would give you more control (e.g. only use the audio modality component). Alternatively, you could run the Emolysis Docker container, which presumably hosts a web service or interface for feeding in video and getting out results. The toolkit’s focus on an interactive UI suggests it might not have a ready REST API out-of-the-box, but given the code availability, wrapping one around it is feasible. Emolysis supports **multi-person, multi-modal input selection via the GUI**, which means under the hood it is capable of taking an input video (with audio) and producing a structured output (likely containing per-person emotions or overall group emotions). For integration into a voice-only coaching system, one would utilize just the audio pathway. This might involve modifying a config to disable face analysis, or simply providing audio with a dummy video frame. The documentation mentions they will provide deployment guidelines and that everything runs locally (no external dependencies aside from standard ML frameworks). Using Emolysis in real-time might require some adaptation: since it was designed for videos (which have an end), one might need to continuously feed a live video stream. But because they tout real-time usage, the system likely can process frame-by-frame on the fly. Integration complexity for Emolysis is a bit **higher** than MVP or MixedEmotions, because it wasn’t originally packaged as a simple API service - it’s a full application. You may need to write some glue code to feed your call audio into Emolysis’s audio model at runtime. On the plus side, Emolysis being open-source and documented in an academic paper means you have transparency. Also, the authors explicitly mention an example use-case of video conferencing analytics and lecture feedback, which aligns with live monitoring. They also provide a Docker on DockerHub for ease of setup. One might run that container and then interface with it (perhaps it opens a local web app for the GUI; if so one might hook into its back-end). Given its academic origin, integration might require **some engineering effort**, but nothing prohibitive - it supports multiple OS, and with Docker it can be deployed on a server accessible to the coaching system. If needed, it’s possible to trim Emolysis down to just the needed parts (only audio emotion). Importantly, Emolysis does not require internet and will run entirely on local hardware, which simplifies integration in secure environments. In summary, Emolysis integration is **flexible** but potentially the most involved: you might treat it as a black-box toolkit (and perhaps manually start it and read outputs), or dive into the code for a custom integration. The reward is a rich output (multi-label emotions + scores) that your coaching logic can use in sophisticated ways.
* **Hume AI:** Integrating Hume is quite different since it is a **cloud API service**. Hume provides a well-documented API (REST and likely streaming endpoints) to send audio and receive emotion analytics. Integration in this case means making API calls from your application to Hume’s service. The **upside** is that this is very straightforward - no model management on your end. Hume’s API can handle audio streams in real-time, so your system could open a websocket or streaming request to Hume’s server, pipe the call audio through it, and get back continuous emotion data. The Hume API would directly give you a structured result, presumably with various emotion scores (e.g. a JSON containing scores for each emotion it detects, updated over time). This ease of integration comes with a few considerations: you must handle **authentication** and abide by Hume’s usage quotas/pricing. Also, sending audio to the cloud means you need a robust, low-latency network connection from the UAE to Hume’s servers. If your platform is already cloud-based and latency is acceptable, Hume could be integrated relatively quickly (their documentation and support are oriented to making this easy). Another consideration is **data governance** - audio from sales calls is sensitive, and integration with Hume means that data will be leaving your controlled environment and going to Hume’s cloud. This might require approvals or might not be allowed at all in some architectures. If it is allowed, you might still want a fallback if connectivity fails. In terms of output integration, Hume’s rich set of emotion signals might require more complex handling on your side (for example, deciding which of the 20 emotions to pay attention to for coaching). But Hume likely offers some guidance or even higher-level metrics. One can integrate it such that certain emotions trigger alerts (e.g. high *frustration* or *anger* score triggers a notification to a supervisor). Because Hume’s model is fixed, integrating also means less flexibility - you can’t tweak the model if it doesn’t perfectly align with your needs; you rely on Hume’s updates. In summary, **Hume integration is the easiest technically** (no local deployment, just API calls) but has external dependencies and potential compliance hurdles. It’s essentially an “out-of-the-box” solution for emotion AI: you plug it in and adjust your application to its output format.

To visualize the integration and deployment features, the table below summarizes key points:

| **Model** | **Integration Interface** | **On-Premise Deployment** | **Real-Time Streaming** | **Dependencies/Requirements** |
| --- | --- | --- | --- | --- |
| **MVP (Husseinshtia1)** | Python-based service (Flask app) with dashboard; containerized for microservice use. Can be called as an API or integrated at code level. | **Yes** - Designed for local or cloud container deployment (Docker/K8s). Edge-device support means it can run on-prem on modest hardware. | **Yes** - Built for real-time emotion detection from streaming audio/video. Low latency on local hardware. | Requires Python environment or Docker. Uses deep learning (GPU recommended for heavy load, but edge CPU possible). No internet needed for inference. |
| **MixedEmotions** | REST API (Docker module). Send audio file or buffer via HTTP; returns JSON-LD with valence & arousal. | **Yes** - Provided as standalone Docker containers. Can be deployed on-prem or cloud host freely (Apache 2.0 for platform, GPL-3 for module). | **Yes** - Supports near-real-time use by sending sequential audio segments. (No built-in streaming API, but can be implemented with periodic requests.) | Very lightweight - runs on CPU efficiently. No external dependencies at runtime (just the container). No internet required (unless using external modules). |
| **Emolysis** | Open-source toolkit (Python) with GUI. Can be integrated by importing code or via Docker deployment. Likely can expose a local API or use directly in-app. | **Yes** - Fully on-prem. Docker image available for easy setup. Cross-platform support (Windows, Linux, even Android/iOS) for local deployment. | **Nearly** - Processes input video/audio in real-time with slight buffering. For pure audio, can be adapted to stream processing. Designed to handle live data for interactive analysis. | Requires deep learning frameworks (e.g. TensorFlow/PyTorch) - better with a GPU for real-time, especially for multiple modalities. No internet required; all inference is local. Some integration effort to feed live audio. |
| **Hume AI** | Cloud **API** (REST/Streaming). Send audio via HTTPS or websocket; receive emotion data in JSON. Well-documented API endpoints. | **No** - **Cloud only.** (Hume’s models run on their servers; no self-host option given publicly.) | **Yes** - Specifically built as a real-time voice emotion API. Streaming support for low-latency continuous analysis. | Requires internet connectivity. Must handle API auth and potential data compliance issues. No infrastructure needed on client aside from network; computation done in Hume’s cloud (they presumably use GPUs). |

## **Comparison to Hume AI’s Emotion Recognition Baseline**

It is valuable to contrast the open-source models with **Hume AI’s emotion recognition**, which can be considered a commercial state-of-the-art baseline:

* **Emotion Coverage:** Hume AI clearly outstrips the open models in the **breadth of emotions** it can identify. As noted, Hume’s system can detect on the order of 15-25 distinct emotion expressions from voice, including many that the open models do not explicitly handle (e.g. *pride*, *guilt*, *calmness*, *confusion*). None of the open-source models match this breadth: Emolysis comes closest with 8 categories (covering basic emotions but not these nuanced ones), while MVP likely sticks to basic emotions, and MixedEmotions doesn’t do categories at all. So in terms of **granularity**, Hume provides a much richer emotional profile of a speaker. This means Hume might catch subtleties in a sales call (like detecting a customer’s *hesitation/uncertainty* or an agent’s *confidence*) that an open model might miss or just roll into a generic positive/negative reading.
* **Accuracy and Human Alignment:** Hume’s models are trained on an unparalleled scale of data (proprietary datasets with **millions of data points**, over a decade of research). In theory, this should give Hume a **higher accuracy** and better generalization to real-world variance than any single open model, which are typically trained on smaller public datasets. Hume’s emphasis on scientifically controlled data collection means their model’s outputs have been validated against human perceptions extensively. For example, Hume references studies that found people can distinguish 12+ emotions in prosody, and they built models to reflect that. The open models, while effective, are limited by their training data: Emolysis uses large but still limited academic datasets (maybe tens of thousands of samples), MVP presumably uses available smaller datasets, and MixedEmotions likely used older corpora. In practical terms, **Hume’s reliability** in correctly identifying the emotion should be superior - especially for nuanced or mixed emotions. It might also be more robust to different speakers, languages, and contexts because of diverse training data. That said, Hume’s model is not infallible; it could overfit to certain vocal indicators that don’t generalize to every scenario. But given its continuous improvement and data, one would treat Hume as a **performance benchmark** that open models would strive to reach.
* **Real-Time Use and Latency:** Hume is designed for real-time use as an API and can handle streaming audio quickly. However, unlike the open-source solutions, it introduces **network latency** (since audio must be sent to Hume’s cloud and results sent back). In a scenario with strong connectivity, this might only add a few hundred milliseconds, making it effectively real-time. But if low latency is critical (say, sub-second feedback), an on-prem model could have an edge by avoiding network transit. Additionally, open models can be embedded at the edge (close to the source of audio), whereas Hume centralizes processing. For a UAE deployment, if the Hume servers are not nearby, latency could be a concern. Summarily, pure inference speed of Hume’s model is likely very fast (they presumably use optimized hardware), but **end-to-end latency** depends on internet connectivity - something the open models sidestep.
* **Integration and Privacy:** Using Hume means relying on a third-party service. This has integration advantages (ease of use, less maintenance) but also **privacy and compliance implications**. In a UAE context, there may be regulations about data sovereignty; streaming customer call audio to a foreign cloud (Hume is a US-based service) might be problematic. The open-source models can all run **fully locally**, ensuring data never leaves the company’s environment. This is a major difference when considering deployment constraints (next section). Additionally, Hume’s integration is subject to licensing costs, whereas the open models are free to use (aside from infrastructure costs). Over time, if a coaching platform scales to many users, Hume’s API costs could accumulate, whereas running an open model on your own GPU server might be more predictable in cost. Hume does, however, bring expert support and updates - you benefit from their continuing improvements. The open models would require you to keep them updated or improved yourself (for instance, integrating new training data if needed).
* **Specific Performance vs. Sales Coaching Needs:** Hume’s broad capability is impressive, but not all of it may be necessary for effective sales coaching. In practice, a sales coach might be most concerned with a subset of emotions: e.g. **customer anger, frustration, engagement, enthusiasm, confusion**, and **agent confidence, empathy, nervousness**. Some of these map to basic categories (anger, happiness), which any model can detect. Some are nuanced (confusion, nervousness) - Hume and possibly Emolysis could detect those, whereas MixedEmotions would only reflect them indirectly in valence/arousal. Hume’s model might output a rich emotional readout that needs interpretation. Meanwhile, an open model like Emolysis could be tailored: since it’s open, one could retrain or adjust it to focus on the most relevant emotional signals for sales contexts (for example, ensure it picks up *customer confusion* by adding training examples or by using the text modality to catch phrases like “I’m not sure I understand”). Hume’s closed model is not easily tunable to a specific domain’s subtleties, though it’s broad enough that it likely covers many scenarios.

In essence, **Hume AI’s baseline sets a high bar**: it offers **greater emotion granularity and likely higher accuracy** than any single open-source model out-of-the-box. However, it requires cloud integration and may not satisfy data locality constraints. The open models are more **specialized or limited**, but they can be deployed and adapted freely. If absolute accuracy and nuance is the top priority (and sending data to the cloud is acceptable), Hume would be a top choice for emotion recognition. But if one prioritizes **control, privacy, and integration into a self-contained system**, the open models (especially Emolysis with its relatively advanced capabilities) might be preferable, even if they don’t cover every emotion Hume does.

To put it succinctly: Hume AI is like a **high-end, turnkey solution** for emotion AI - broad, powerful, and easy to consume - whereas the open-source models are **flexible, inspectable components** that you can host and modify, but might require combining or refining to approach Hume’s coverage and performance.

## **UAE Deployment Considerations (On-Premises Feasibility)**

Deploying an emotion recognition system in the UAE introduces some practical constraints and requirements, particularly regarding data privacy (possibly keeping data on-premises), network reliability, and hardware availability. Here we evaluate each model in the context of a local deployment in the UAE, assuming we want to minimize dependence on external services:

* **MVP Emotion Recognition (on-premises):** MVP was explicitly designed with **edge deployment** in mind. It supports running on local machines and even IoT/edge devices, which implies it can function without internet connectivity. For a UAE deployment, you could set up MVP on a local server (or a set of servers) possibly equipped with GPUs for best performance. Since it uses Docker/Kubernetes, one can containerize the whole stack and deploy it within a closed network (for example, within a call center’s data center). All processing of audio, video, and text would occur locally, satisfying any regulatory requirements about data residency. **CPU vs GPU:** MVP’s workload for voice emotion will run on CPU, but real-time processing of multiple concurrent calls might demand a GPU to keep up, especially if using a heavy model (like a transformer) per audio stream. If the scale is one call at a time, a modern CPU could manage. The nice aspect is you can scale horizontally - spin up multiple containers on different servers to handle more calls, orchestrated by Kubernetes. **Internet dependency:** None at inference time - the only time internet might be needed is to fetch any pre-trained model weights (if not included) or updates, but those can be done offline as well by manual transfer. **Containerization:** Provided out-of-the-box, which eases deployment and management. **UAE specifics:** There’s no known restriction unique to UAE that would hinder running MVP locally. Power and cooling for GPUs, etc., are standard considerations. In summary, MVP is **feasible to deploy fully within UAE** infrastructure - it aligns well with an on-prem strategy, giving full control over data and systems. The organization just needs to ensure they have the computing resources to run it (which can be modest if only a few concurrent analyses, or significant if scaling to many users).
* **MixedEmotions (on-premises):** MixedEmotions was an EU project with emphasis on deployment in industry (including possibly telecom scenarios). It is provided as **Docker images** that you can run locally, so deployment in UAE is straightforward: no internet connection is required for operation since the model runs entirely on local compute. **CPU vs GPU:** MixedEmotions’ audio emotion module does not need a GPU - it’s lightweight enough to run on CPU in real-time. This makes it attractive if GPU resources are scarce or if you want to deploy on standard commodity servers. It also means edge devices (even a high-end laptop or an embedded PC) could potentially run it. Since it uses an older approach, memory and CPU requirements are low (openSMILE feature extraction can run in real-time on a single core typically). **Scalability:** Because each analysis is small, one server could handle dozens of concurrent audio streams by running multiple instances or threads of the model. **Containerization and orchestration:** MixedEmotions modules can be orchestrated using Mesos or just Docker Compose; integrating it into a container environment in UAE is trivial. **Internet dependency:** None, unless you choose to use some MixedEmotions modules that call external APIs (not the case for audio emotion). All data stays local. One caveat: the project is a few years old, so you’d want to ensure compatibility with current OS/Docker versions, but as it’s fairly simple, it should still build/run fine. **Local modifications:** If needed, you can retrain or adjust the SVM model with local data - though given its performance, you might rather move to a deep model if retraining. In short, MixedEmotions is **very friendly to on-prem deployment** in UAE - minimal requirements and entirely self-contained.
* **Emolysis (on-premises):** Emolysis was explicitly designed to be a **standalone toolkit** running locally (the paper even emphasizes privacy: no data leaves and nothing is stored). For UAE deployment, Emolysis offers a Docker image on DockerHub which can be pulled and run on a local server. All the heavy computation happens within that container. **Hardware needs:** Emolysis’s use of deep learning (face and audio models) means a **GPU is strongly recommended** for real-time performance. On CPU-only, Emolysis might lag or struggle, especially if analyzing video. However, if we restrict to audio analysis, the computational load lessens - possibly a high-end CPU could manage a single stream in real-time (particularly if the audio model is lighter than the video CNN). But for multiple streams or for future-proofing, having a GPU (like an NVIDIA Tesla or even a gaming-grade GPU) on the server in UAE would ensure the model runs smoothly. We can also scale Emolysis by running multiple containers (each could handle one video/audio feed) if needed, as long as hardware resources (GPUs or CPU cores) are sufficient. **Internet dependency:** None - Emolysis does not call out to any cloud service; once the models are downloaded and running, all inference is local. In an offline environment, you’d need to have the Docker image and model weights pre-fetched (so initial setup might require internet or manual transfer). But after that, no connectivity is needed for operation. **Containerization:** Provided, which eases deployment on local Docker hosts or Kubernetes clusters. **Mobile deployment:** Interestingly, they mention support for Android/iOS - implying that, in principle, one could deploy Emolysis on-device. That likely involves using a smaller model or only certain features due to limited mobile compute. For a sales coaching scenario, mobile deployment is probably not needed (the analysis can happen on a server). But it’s good to know the toolkit is flexible enough if one ever wanted to run it on an agent’s device (though that’s an unusual choice because you’d rather centralize analysis). **UAE considerations:** None beyond standard on-prem. If certain GPU models are hard to source, one might have to adjust (e.g. ensure compatibility with available hardware). But being open-source, there’s no legal or service barrier. Emolysis can be fully compliant with data regulations since everything stays in-country on your machines. In summary, Emolysis is **well-suited for local deployment** in UAE, provided the necessary compute hardware is allocated. It delivers advanced capability without external dependencies, aligning with any strict data governance requirements.
* **Hume AI (on-premises):** Hume AI, as of public information, does not offer an on-premise or private deployment option. It is a cloud-hosted SaaS/API product. For deployment in UAE, this presents a **challenge** if data privacy laws or company policies demand that no customer data (like call audio) leaves the country or the controlled environment. Using Hume would mean streaming audio to Hume’s servers (likely in Europe or the US), which could violate such constraints. Even if not outright illegal, it could pose security and trust issues - financial or government clients in UAE might not approve of external data transfer. Additionally, network reliability could be a factor; if connectivity to Hume’s cloud has high latency or is intermittent, the system’s performance would be affected. There’s also the issue of **localization** - if calls happen in Arabic or Hindi (common in UAE) or other languages, one would need to confirm that Hume’s models handle those languages well. Hume’s research is largely on universal vocal expressions (which might transfer across languages), but some aspects (like linguistic content understanding in their EVI) could be language-specific. Without an on-prem version, adapting Hume to those nuances is not directly in your control. **Possible workarounds:** One could theoretically use Hume’s API from the UAE if they accept the data transfer risk, perhaps using encryption and ensuring compliance with any customer consent. Alternatively, if Hume AI offered an enterprise on-prem solution (not publicly advertised, but maybe possible via special arrangement), that could change the equation - but that would likely be costly and involve close collaboration with Hume. In the absence of that, **Hume is essentially off-prem** and thus the least favorable for a fully local deployment.

In summary, for **UAE deployment**, the open-source models (MVP, MixedEmotions, Emolysis) are all **viable to run locally** and each comes with containerization support, making them easy to deploy on local servers. They require varying hardware (MixedEmotions can run on CPU easily; MVP and especially Emolysis benefit from GPU acceleration). None of them require an internet connection at runtime, which is ideal for a secure environment. **Hume AI, by contrast, requires internet connectivity to a foreign service**, which may conflict with UAE data management requirements and introduces external dependencies. Therefore, from a deployment standpoint, the open models have a clear advantage for an on-prem, self-contained system.

## **Conclusion and Recommendations**

After evaluating granularity, latency, reliability, integration, and deployment factors, we can make the following recommendations for **real-time emotion coaching in sales scenarios (especially under UAE constraints)**:

**Best Suited Model(s):** **Emolysis** emerges as a top choice among the open-source options for a sophisticated real-time coaching system. It offers a rich emotion taxonomy (covering more than just the basics) and provides continuous valence/arousal scores, aligning well with the nuanced feedback needed in sales conversations. Emolysis’s design for real-time, multi-person analysis means it can handle live call streams and even scale to analyze both the salesperson and customer if desired. Critically, it runs **fully on-premises in a Docker container**, requiring no internet and thus meeting UAE deployment requirements. With a proper GPU, Emolysis can operate with low latency and high accuracy. Its ability to detect multiple concurrent emotions (e.g. a mix of *anticipation* and *nervousness*) could be very valuable in coaching, where understanding complex customer signals is key. We recommend Emolysis for organizations that have the technical capacity to integrate a slightly complex toolkit and want the **most comprehensive open-source solution**.

If computational resources or integration effort for Emolysis are a concern, the **MVP Emotion Recognition** project is a strong alternative. MVP provides a more **plug-and-play infrastructure** (complete with dashboard, monitoring, and cloud-native deployment scripts) which could accelerate development of a coaching platform. It handles real-time audio emotion detection and can be extended to text and video, providing a holistic view of interactions. While its exact emotion granularity is less defined (likely focusing on core emotions), it should suffice for detecting key cues like anger, joy, or frustration in voices. MVP is also built with **edge deployment in mind**, so it aligns with running locally in UAE. For a team that wants a ready solution to start with and possibly improve over time, MVP could be a good starting point - with the understanding that you may later enhance its emotion taxonomy or accuracy by incorporating new models. Essentially, MVP is recommended for a scenario where ease of integration and a full-stack solution is prioritized over the absolute cutting-edge of emotion nuance.

The **MixedEmotions** audio module, while outdated in approach, could still play a role as a **lightweight baseline**. It would reliably give a quick read on whether a call’s tone is positive/negative and energetic/calm. In a live sales coaching scenario, this might be used for simple alerts (e.g., if customer valence drops sharply, flag the call). MixedEmotions is extremely easy to deploy and requires minimal resources, making it an option for scaling out broadly on many streams without heavy infrastructure. However, its lack of discrete emotion output limits its coaching usefulness - a coach would have to infer the specific emotion cause. Thus, MixedEmotions might be best used in combination with other models: for instance, use MixedEmotions to continuously monitor all calls for potential issues (since it’s efficient), and when a low-valence/high-arousal event is detected, use a more detailed model like Emolysis or MVP to diagnose whether that was due to anger, fear, or some other emotion. This kind of tiered approach could be a pragmatic way to balance resource usage with insight depth. If choosing a single model, though, MixedEmotions on its own is **not as recommended** for this use-case, because it doesn’t give the nuanced direction that a sales coach would need (it tells *how* the customer feels in broad terms, but not *why* or *what kind* of negative feeling, for example).

As for **Hume AI**, it undoubtedly provides excellent performance and a rich understanding of emotions - likely superior to the open-source models in a vacuum. However, given the **UAE deployment constraints** (and generally, concerns of sending call audio to an external cloud), Hume is less suitable in this context. In a scenario where those constraints are relaxed (say, if data residency is not an issue and ultra-nuanced analysis is needed), one could consider Hume’s API as a gold-standard. It could even be used to benchmark the open-source models - for example, run Hume on some sample calls to see what insights it provides, and then aim to replicate the most relevant insights with your on-prem models. But for an actual deployed system in the UAE, we **advise against relying on Hume’s cloud service** for real-time coaching, due to the potential legal, ethical, and latency issues of offshoring sensitive audio data. The benefit Hume would bring (like identifying very subtle emotions) might not outweigh the complexity of dealing with cross-border data flow and API dependence in a mission-critical coaching system.

**Recommendation Summary:** For real-time emotion coaching in sales calls under UAE conditions, **Emolysis** is recommended as the primary model due to its combination of nuanced emotion recognition, on-premise deployment, and real-time capability. It should be set up on a local GPU-equipped server, possibly containerized, and integrated into the call monitoring pipeline. Coaches and automated systems can then receive a stream of both high-level emotion labels and continuous sentiment indicators, enabling timely and context-aware interventions (e.g., advising the salesperson to clarify something if *confusion* is detected, or to adjust tone if customer *anger* is rising). **MVP** can be considered if a quicker-to-implement, multi-modal solution is desired, or even alongside Emolysis (for instance, MVP’s text analysis could complement Emolysis’s audio analysis). **MixedEmotions** can serve in a supplementary role where simplicity and efficiency are needed (e.g., as a watchdog on all calls). Finally, direct use of **Hume AI** is not recommended on-premises; instead, one might take inspiration from Hume’s capabilities to guide which emotions to focus on.

By choosing an open-source model and deploying it locally, a sales coaching platform in the UAE will maintain **full control over sensitive data**, ensure **low-latency feedback**, and avoid licensing costs, all while benefiting from cutting-edge emotion recognition technology. Emolysis (or a comparable open model) backed by a solid integration into the call center workflow will provide coaches with real-time insights - for example, highlighting that *“The client’s tone has turned angry (high arousal, very negative valence) while discussing pricing”* - allowing the sales team to respond proactively and empathetically in the moment. Such an approach leverages the strengths of the recommended model and aligns with the operational constraints of a UAE deployment, ultimately enhancing the effectiveness of real-time sales coaching.

**Sources:**

* Husseinshtia1 - *MVP Emotion Recognition* (open-source multi-modal emotion dashboard)
* MixedEmotions Project - *Open-Source Emotion Analysis Toolbox* (audio valence/arousal module)
* ControlNet - *Emolysis Toolkit* (multimodal group emotion recognition in real-time)
* Hume AI - *Empathic Voice API and Research* (high-dimensional emotion recognition)