ML/DL Intern Technical Assignment

Deadline: 28 May 2025

Upload the file in google form here: https://forms.gle/LBgcQbG5A5UBkqK78

Bee-Acoustic ML Challenge (Choose one of two options)

Topic: Acoustic data related to honeybee hives.

Using only **audio recordings or related acoustic features**, build an end-to-end solution.

Context:

- Bioacoustics is the interdisciplinary science that studies sound production, transmission, and reception in living organisms.
- It encompasses signals ranging from complex animal vocalizations to subtle insect wingbeat frequencies.
- By capturing and analyzing environmental sounds, bioacoustics enables non-invasive monitoring of species behavior and ecosystem health.
- In ecology, it serves as a cost-effective tool for population surveys, habitat assessment, and early warning of environmental stress.
- Acoustic data can reveal temporal patterns in activity—diurnal or seasonal—without ever disturbing the subjects.
- For bee monitoring, bio-acoustic analysis allows real-time tracking of hive health and foraging behavior via characteristic buzzing and wingbeat signatures.
- Subtle shifts in acoustic profiles can signal queen absence, swarming, disease onset, or environmental stressors before they become visually evident.
- Integrating bioacoustics into ML pipelines thus provides scalable, automated insights to support precision apiculture and colony management.

Options (pick one)

1. **Activity Chart Generation**
    a. Predict and visualize daily hive foraging activity levels (e.g., foraging flight frequency or vocalization rates) over time.
2. **Queen Bee Presence (Yes/No)**

a. Classify whether the queen bee is present in a hive audio clip.

Pipeline Requirements

1. **Reproducible Environment**
    a. Provide a single Jupyter Notebook (`.ipynb`) or Python script (`.py`) with docstrings and markdown cells explaining each step.
    b. Include a `requirements.txt` (or equivalent) listing all packages and versions.
2. **Data Acquisition**
    a. Identify and download an open acoustic dataset appropriate for your chosen option (e.g., publicly available bee audio recordings).
    b. Clearly state the dataset name, source URL, and any preprocessing steps applied.
3. **Preprocessing & Feature Engineering**
    a. Audio loading and normalization (e.g., sample rate, duration trimming).
    b. Noise reduction (e.g., spectral gating, bandpass filtering).
    c. Feature extraction: spectrograms, Mel-frequency cepstral coefficients (MFCCs), or embeddings from a pre-trained audio model.
    d. Document each transformation with inline docstrings or markdown.
4. **Model Development**
    a. Define one or more ML/DL architectures suited to the task (e.g., CNN, LSTM, CRNN, or fine-tuned audio foundation model).
    b. Clearly state which algorithm or pre-trained model you use and why.
5. **Data Split**
    a. Partition into train/validation/test (suggested split: 70/15/15), ensuring no overlap in recording sessions or hive instances.
6. **Training & Hyperparameters**
    a. Implement training loops with hyperparameter settings and early stopping criteria.
    b. Use explicit docstrings to explain function inputs, outputs, and logic.
7. **Evaluation Metrics**
    a. For **Activity Chart Generation** (regression): RMSE, MAE, R^2, and qualitative time-series plots against ground truth.
    b. For **Queen Bee Presence** (classification): AU-ROC, AU-PRC, accuracy, precision, recall, F1-score, plus a confusion matrix and ROC/PR curves.

    c. Provide visualizations and commentary on model performance.
8. **Inference & Visualization**
    a. Generate predictions on held-out test data.
    b. Produce clear charts: time-series activity plots or classification probability histograms.
    c. Document how to run inference in one command or notebook cell.

Deliverables

- **Code:** One `.ipynb` or `.py` file, fully runnable from start to finish.
- **Environment:** `requirements.txt` or environment specification.
- **Documentation:** Inline docstrings and markdown explanations for every major block.
- **Results:** Numeric metrics, plots, and a brief written summary of findings (in a markdown cell or top of script).
- **References:** List any datasets, papers, or GitHub repositories consulted.

Suggested Evaluation Criteria

Criterion	Notes
Reproducibility	Code runs end-to-end without errors
Data Understanding	Appropriate preprocessing and feature engineering
Model Selection & Justification	Clear rationale for algorithm choice
Metric Reporting	Correct implementation and interpretation of metrics
Documentation & Code Quality	Readable code, comprehensive docstrings, and markdown
Visualization & Insight	Informative plots and concise discussion of results