The Embedded Echoes Challenge

Deciphering Meaning from Lost Data

DSS ML Competition Committee

November 15, 2025

1 Introduction: The Lost Archive

A few weeks ago, a catastrophic data corruption event struck our primary archive. We lost the original, vibrant collection of images and their associated text descriptions. What survived was not the data itself, but its digital "ghosts": the high-dimensional embeddings generated by our internal models.

These "embedded echoes" are all that remain. We have:

- A 512-dimensional vector for each image.
- A 512-dimensional vector for the text that once described it.

While the raw pixels and characters are gone forever, these embeddings still hold the semantic essence of the original data. We also managed to recover a partial metadata file indicating whether a sample was **"important" (label 1)** or **"not-important" (label 0)**.

Your mission, as participants in this challenge, is to build a model that can sift through these echoes and learn to classify the data, even without ever seeing a single image or word.

2 The Challenge

The goal is to build a binary classification model that predicts the `target` variable (0 or 1) for a given test set. The competition is divided into two exciting stages.

2.1 Competition Timeline & Stages

2.1.1 Stage 1: The Open Qualifier

- **Starts:** November 16th, 10:00 AM
- **Ends:** November 17th, 3:30 PM
- **Data:** You will be provided with `train_part1.json` to build your models.
- **Goal:** Achieve the highest possible score on the private leaderboard. Your standing at 3:30 PM will determine your advancement.

2.1.2 Stage 2: The Finalists' Round

- **Starts:** November 17th, 4:00 PM
- **Eligibility:** The top 15 teams from the Stage 1 private leaderboard will be invited to compete.
- **New Data:** At 4:00 PM, the second file, `train_part2.json`, will be released to the finalists to augment their datasets, test generalization, and refine their final models.

2.2 File Descriptions

2.2.1 `train_part1.json` (Stage 1)

This file contains the first half of the shuffled training dataset. Each entry is a JSON object with the embeddings and the ground-truth label.

```
{
    "id": "a9d8c7...", // Hashed ID
    "label": 0,   // The target: 0 (not-important) or 1 (important)
    "image_embedding": [
        0.0440,
        -0.0269,
        ...
        (512 floats)
    ],
    "text_embedding": [
        0.0474,
        -0.0405,
        ...
        (512 floats)
    ]
}
```

Listing 1: Example record from `train_part1.json`

2.2.2 `test.json` (Used in all stages)

This is the test set for which you must predict the labels. It follows the same structure as the training file but **omits the `"label"`** field. The `"id"` field is a sequential integer (1, 2, 3,...) that you must use in your submission.

```
{
    "id": 1,
    "image_embedding": [
        0.0123,
        -0.0567,
        ...
        (512 floats)
    ],
    "text_embedding": [
        0.0890,
        0.0112,
        ...
        (512 floats)
    ]
}
```

Listing 2: Example record from `test.json`

3 Submission Format

Your submission must be a CSV file with exactly two columns: `row_id` and `target`.

- `row_id`: Must correspond to the sequential `id` from the `test.json` file (1, 2, 3, ...).
- `target`: Your model's binary prediction (0 or 1).

A `sample_solution.csv` is provided to show the exact format.

```
row_id,target
1,0
2,1
3,0
4,1
...
```

Listing 3: Format of `sample_solution.csv`

4 Evaluation

Submissions will be evaluated using the **F1 Macro** score. This metric is robust to class imbalance (if any exists) by calculating the F1 score for each class independently and then taking the unweighted average.

5 Getting Started: Hints & Ideas

You are working with 1024-dimensional data (512 from images, 512 from text). How you use this data is up to you. Here are a few starting points:

- **Classical ML Approach:** You can think of the embeddings as 1024 features. Concatenate the `image_embedding` and `text_embedding` into a single 1024-dimensional feature vector. You can then train models like **XGBoost**, **SVM**, or **Random Forest** directly on this flat data. Don't forget to experiment with feature reduction techniques like **PCA**!
- **Deep Learning / MLP Approach:** You can leverage neural networks for more complex feature interactions and fusion.
  - the current location is Karachi, Sindh, Pakistan. **Early Fusion:** Concatenate the 1024 features and feed them as a single input vector into a simple Multi-Layer Perceptron (MLP).
  - **Late Fusion:** Build two separate "towers" (small MLPs), one for the 512-dim image embedding and one for the 512-dim text embedding. Then, combine their output representations (e.g., by concatenation or averaging) before passing them to a final classification layer.

Creativity is yours to explore. The best approach may be a novel one. Good luck!