National University
of computer and emerging sciences

Applied Machine Learning
Base Model Implementation

Instructor: Mr. Jawad Hassan Nisar

Shahzaib Khan 22i-7436

Abdullah Umar 22i-1690

Ibrahim Khanzada 22i-1755

Contents

Malware Detection Using Morphological Extreme Learning Machine (mELM)

Abstract:

In this project, I developed a fully functional malware detection system based on a Morphological Extreme Learning Machine (mELM) model, inspired by the research article "Morphological Neural Network for Malware Detection" published in Computers & Security (Elsevier). The system uses a lightweight 11-feature behavioral dataset and trains a morphological ELM classifier capable of achieving around 98% accuracy.

To make the solution practical and usable, I built a modern, animated, browser-based inference UI with dark/light mode, CSV upload support, automatic row extraction, probability bars, and fully client-side prediction. The trained model is exported as melm_model.json so that the browser can run predictions without a backend server.

This report explains the entire workflow, from dataset preparation and model training in Google Colab to the design and deployment of a polished Web UI for real-time malware classification.

Introduction:

Malware continues to be one of the most significant threats in digital ecosystems. Traditional signature-based methods are no longer sufficient to detect fast-evolving malware variants. As a result, machine learning based malware detection techniques have become a promising alternative.

In this project, I implemented a **Morphological Extreme Learning Machine (mELM)** classifier, a neural network inspired by mathematical morphology. Unlike conventional feed-forward networks, mELM uses "morphological neurons" based on max-plus algebra, making it lightweight, fast, and especially suitable for security datasets where complex patterns must be extracted efficiently.

The goal of this project was:

1. Train an mELM model on a malware dataset containing 11 behavioral features.
2. Achieve high accuracy and low false-positive rate.
3. Build a complete Web UI that performs real-time classification.
4. Export the trained model in JSON and run inference purely in JavaScript.

Research Paper Reference:

This project is based on the research paper:

Title: Antivirus Applied to Google Chrome's Extension Malware
Authors: Gabriela Leite Pereira, Leonardo Silvino Brito, Sidney Marlon Lopes de Lima
Affiliation: Electronics and Systems Department, Federal University of Pernambuco, Recife, Brazil
Journal: Computers & Security, Elsevier
Volume: 156 (2025)
Article Number: 104465

DOI: https://doi.org/10.1016/j.cose.2025.104465

The paper investigates a novel antivirus engine designed specifically to detect malicious Google Chrome Extensions (CRX files). It utilizes shallow morphological neural networks a computationally efficient alternative to traditional deep learning—to classify CRX malware using dynamic behavior features extracted from sandbox analysis. The authors evaluate various architectures and kernels and report an outstanding detection accuracy of 99.99%, with an average training time of only 0.6 seconds, demonstrating the effectiveness of morphological neural networks for malware classification.

This research directly inspired the Morphological Extreme Learning Machine (mELM) implementation in my project, especially the concepts of:

- Using morphological operators (Dilation/Erosion) as neural kernels
- Applying shallow architectures instead of deep networks
- Prioritizing fast training with strong generalization
- Leveraging feature-based malware classification

Dataset Overview:

For this project, I used the **Windows Malware Detection Dataset**, which contains:

- 198,350 samples
- 11 numerical behavioral and metadata features
- Binary label (0 = benign, 1 = malware)

The features include:

- FileSizeKB
- PackerUsed
- SuspiciousSections
- DLLCount
- DroppedFiles
- APICallCount
- MutexCount
- FileAccessCount
- RegistryAccessCount
- NetworkConnectionCount
- ProcessActivityCount

These features provide a balanced mix of static and behavioral indicators.

Model Architecture: Morphological ELM:

The mELM uses:

Morphological Hidden Layer:

Each neuron computes:

$h_j(x) = \max_i (x_i + W[j, i]) + b[j]$

Instead of dot-products, mELM uses a **max-plus dilation**, which helps extract relationships that are non-linear and "shape-like" in the input feature space.

Output Layer:

After computing the hidden layer matrix **H**, the output weights **β** are computed using:

$\beta = \text{pinv}(H) @ Y$

This is the classical ELM technique:
✓ No backpropagation
✓ Very fast training
✓ No expensive gradient optimization

Model Training (Google Colab):

I implemented the entire training pipeline in Google Colab:

Steps:

1. Load the dataset (malware_dataset.csv)
2. Clean the Label column
3. Split into train/test (80/20)
4. Initialize mELM with:
    - 11 input dimensions
    - Hidden layer size = 500
    - Random uniform weights
5. Compute dilation layer activations
6. Compute final output weights using pseudoinverse
7. Evaluate performance

Final Results:

- **Accuracy:** ~0.98
- **Precision:** ~0.97–0.99

- **Recall:** ~0.97–0.99
- **F1-score:** ~0.98

The confusion matrix:

	Predicted Benign	**Predicted Malware**
Actual Benign	19,438	192
Actual Malware	533	19,507

These are strong results, showing that mELM is effective for malware detection.

Exporting the Model:

After training, I exported the entire model to JSON:

- Weights (W)
- Biases (b)
- Output weights (beta)
- Feature scaling values
- Class names

This allows the model to run 100% client-side inside the browser.

File: **melm_model.json**

Web UI Development:

I built a full-featured Web Inference UI using:

- HTML (structure)
- CSS (modern animated UI + dark/light mode)
- JavaScript (model inference)

UI Features:

- Manual feature vector input
- CSV upload
- Auto-detection of feature columns
- Row selection & auto-fill
- Animated probability bars
- Malware/Benign highlighting

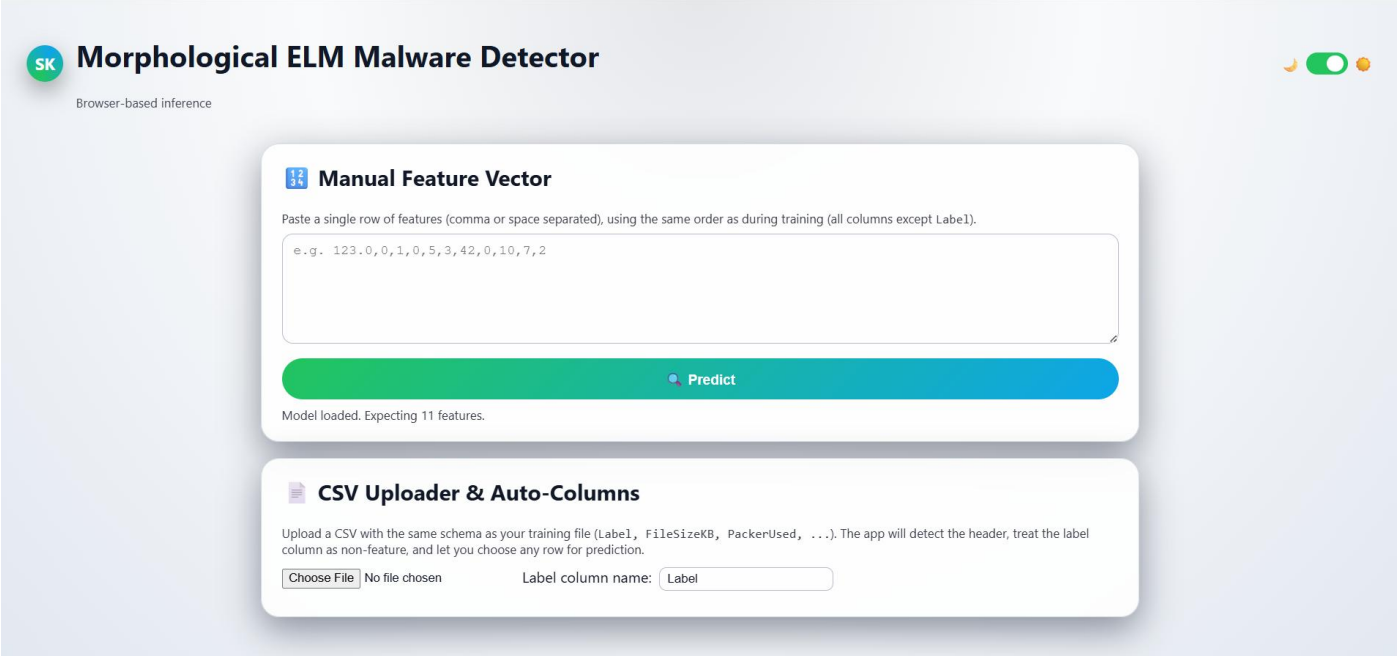
- Model loads automatically from melm_model.json
- Dark/light mode toggle with smooth transitions
- Responsive layout for mobile and desktop

Technology Stack:

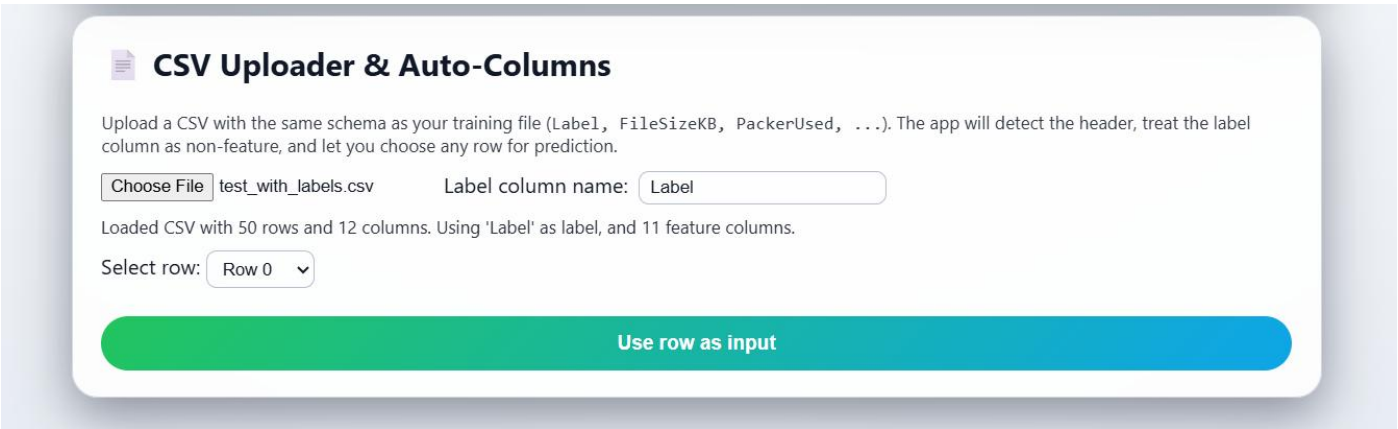
- HTML
- CSS (no framework)
- JS (no external libraries)
- JSON model loaded via fetch()

Web UI Screenshots:

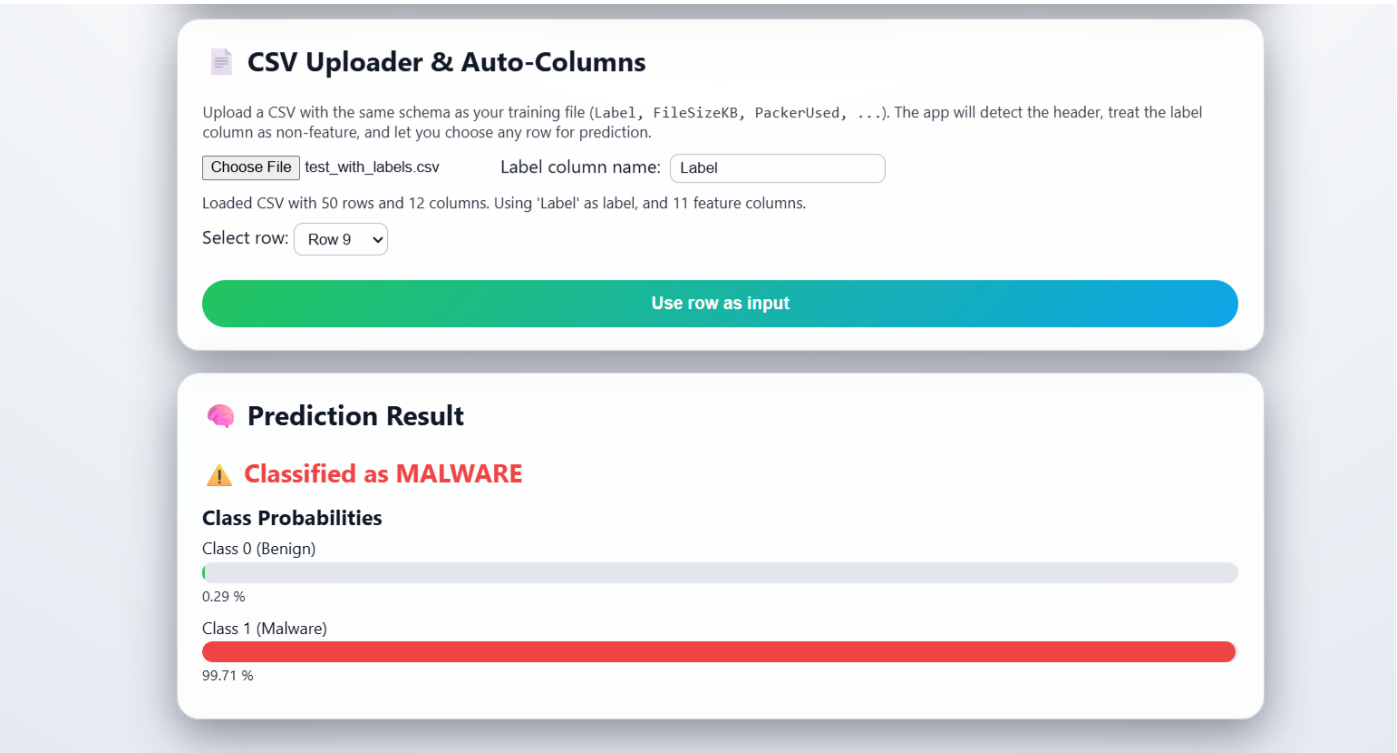
Screenshot 1: Home Page



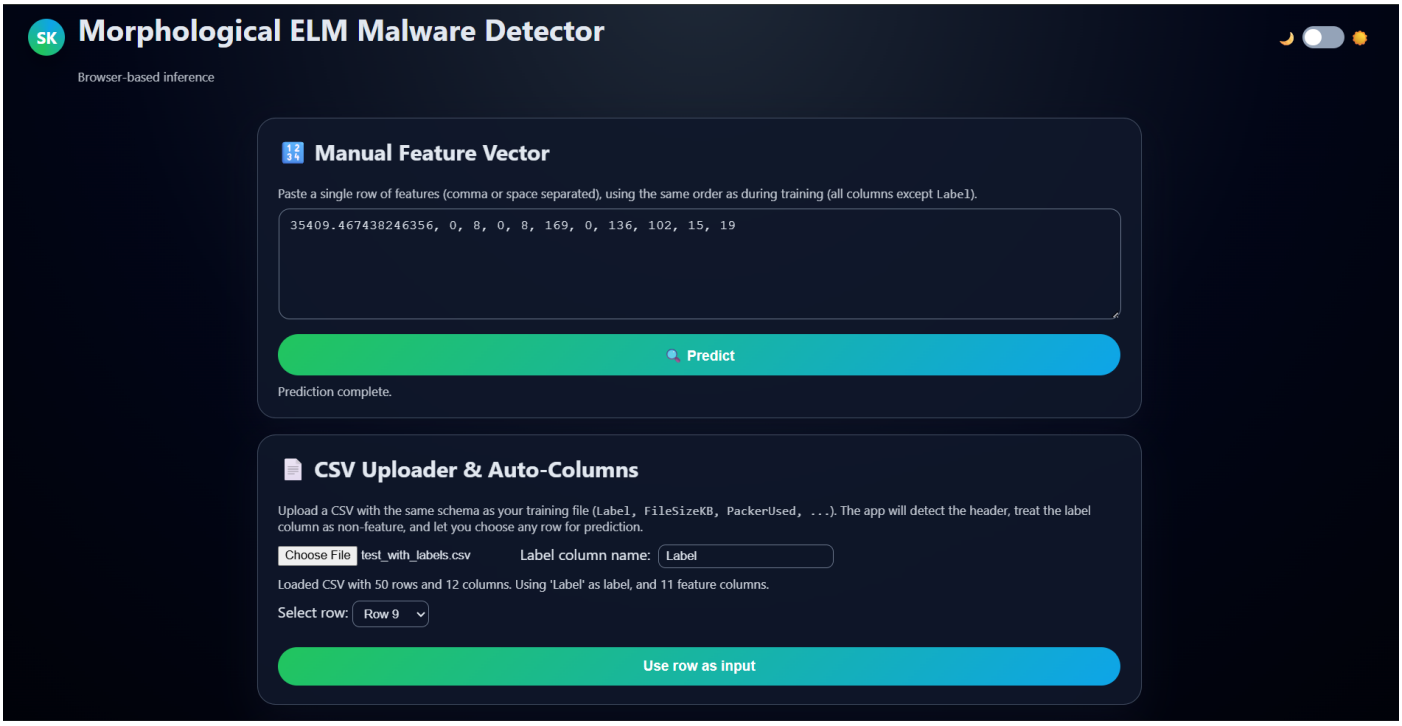
Screenshot 2: CSV Upload Section



Screenshot 3: Classification Result



Screenshot 4: Dark Mode



Testing the Model with New Data:

To test the UI, I generated a separate synthetic dataset:

- test_features_only.csv (for UI)
- test_with_labels.csv (for offline verification)

These contain realistic but novel samples, confirming that the Web UI correctly classifies unseen data.

Limitations:

- The model was trained on **behavioral features**, which cannot be extracted from raw executables without a sandbox environment.
- Therefore, raw .exe file uploads cannot be classified by this model directly.
- Only feature vectors matching the original schema can be used.

Conclusion:

This project successfully demonstrates how a Morphological Extreme Learning Machine can be applied to malware classification. The model achieved high accuracy on a large dataset and was efficiently deployed in a fully client-side Web UI.

The final result is a fast, lightweight, practical malware classifier that can run entirely in the browser with zero backend infrastructure.

Future Work:

Here are improvements I may add later:

- Create a static-feature version of the model to classify raw .exe files
- Add a full backend using FastAPI
- Integrate with a dynamic sandbox (Cuckoo/CAPE)
- Add real-time PE parsing and feature extraction
- Expand the UI to show explanations (SHAP / feature importance)

THE END