Université de Carthage
Institut National des Sciences Appliquées et Technologiques
Ministère des Études Supérieures et de la Recherche Scientifique

Personal Professional Project:

Stegalens : Intelligent Multimodal Steganography Detection System

Field of study:

Computer Networks and Telecommunication

Prepared by:

Ghada Jeddey
Emna Drihem
Yasmine Ferjani

Supervised by:

Dr. Wided SOUID MILED

Année Universitaire : 2024 - 2025

Contents

Project Summary

This project develops an intelligent steganalysis system that automatically detects hidden data across multiple file types through specialized deep learning models . The end goal is an integrated interface where users can upload any suspicious file, which the system will automatically classify by type (image, audio, or network packet) and route to the appropriate detection model—analyzing pixel patterns for images, audio waveforms for sound files, or packet structures for network traffic—to determine if the file contains hidden steganographic content and flag potential security risks. The modular design allows each model to specialize in its domain while sharing a unified detection framework. This project bridges the gap between theoretical steganalysis research and practical cybersecurity tools, offering a proactive defense against covert data exfiltration.

Problem statement and Objectives

2.1 Problem Statement

Steganography—the practice of hiding data within digital files—poses a growing cybersecurity threat, as malicious actors increasingly use it to conceal malware, exfiltrate sensitive data, or bypass detection systems. Traditional security tools often fail to detect such hidden content, especially when spread across different file types (images, audio, network packets). Manual analysis is impractical due to the volume and complexity of modern digital communications. There is a critical need for an automated, multi-modal steganalysis system capable of accurately identifying steganographic content across diverse file formats while maintaining usability for security professionals.

2.2 Objectives

1. **Develop Specialized Detection Models**
   - **Image Steganalysis:** Build a CNN-based model using a modified ResNet34 architecture to detect hidden data in images.
   - **Audio Steganalysis:** Create models to analyze WAV and MP3 files for steganographic content using features such as LPC and MFCC.
   - **Network Steganalysis:** Develop detection mechanisms for covert data in packet captures using extracted network features.
2. **Automated File-Type Routing**
   - Design and train a file-type classifier to automatically identify the input file type (image, audio, or network packet).
   - Route the classified files to the corresponding steganalysis model.
3. **Unified Threat Assessment**
   - Integrate all detection models into a unified system.
   - Output a clear risk score for each file: *safe* or *malicious*.
4. **Real-World Usability**
   - Optimize detection algorithms to reduce false positives and avoid unnecessary alerts.
   - Ensure scalability to support batch processing of large volumes of files efficiently.

Literature Review and Theoretical Concepts

This section presents the theoretical foundations and related work that guided the development of our steganalysis system. We highlight key concepts such as steganography techniques across different modalities (image, audio, and network traffic), as well as the use of transfer learning in deep neural networks.

3.1 Steganography in Digital Media

Steganography is the practice of concealing information within digital media to prevent detection. Unlike encryption, which scrambles content, steganography hides the very existence of the message. The goal of steganalysis, therefore, is to detect the presence of such hidden content.

- **Image Steganography:**
  - **LSB (Least Significant Bit):** This method embeds secret data by replacing the least significant bits of pixel intensity values. For example, changing the last bit of a grayscale pixel value (e.g., from 100 to 101) causes negligible visual difference but can carry binary data.
  - **WOW (Wavelet Obtained Weights):** This adaptive method minimizes distortion by assigning embedding costs based on wavelet filter responses. It embeds data in parts of the image that are less likely to reveal artifacts, typically in textured regions.
  - **HILL (High-pass, Low-pass, Low-pass):** This algorithm uses high-pass and low-pass filtering to guide embedding in noise-like areas of the image. It aims to reduce statistical detectability by spreading changes over perceptually insignificant regions.
- **Audio Steganography:**
  - **CNV-QIM (Constrained Neighboring Vector Quantization Index Modulation):** This technique is specifically tailored for low bit-rate speech codecs like G.729a, which utilize Vector Quantization (VQ). CNV-QIM operates within the Linear Predictive Coding (LPC) domain by dividing the quantization codebook into two sub-codebooks—one for embedding a binary 0 and the other for a binary 1. To ensure imperceptibility, codewords are assigned such that neighboring vectors reside in different sub-codebooks (the CNV condition). During embedding, the encoder selects the nearest codeword from the relevant sub-codebook (0 or 1), ensuring low distortion and preserving audio quality.

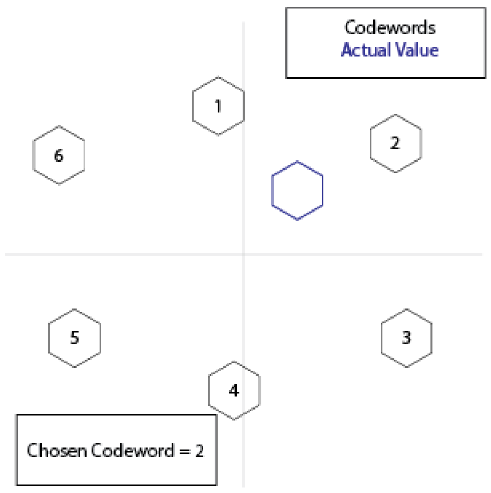


Figure 1: The codeword choice in a normal case

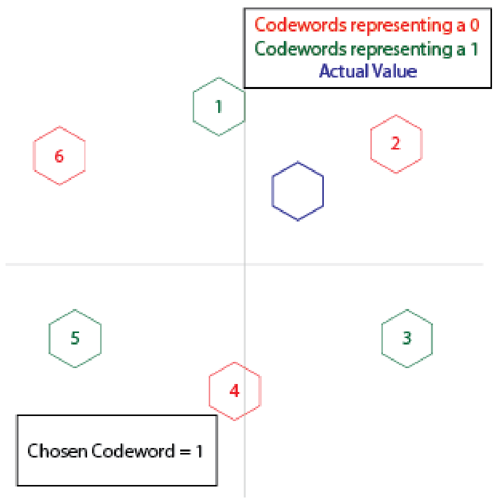


Figure 2: the codeword choice using the CNV-QIM algorithm

- **PMS (Pitch Modulation Steganography):** PMS works by embedding secret data into a speech signal through subtle, imperceptible modifications of its pitch contour (the pattern of how the perceived pitch, or fundamental frequency, changes over time). It exploits the human ear's relative insensitivity to very small changes in pitch, especially when those changes are within the natural fluctuations of speech.

- **Network Steganography:**
  - **TCP Header Manipulation:** This method embeds hidden data in rarely used or optional fields of protocol headers, such as TCP sequence numbers, reserved bits, or unusual flag combinations (e.g., SYN and FIN both set). These subtle alterations are often overlooked by intrusion detection systems, allowing covert data transmission without disrupting network functionality.

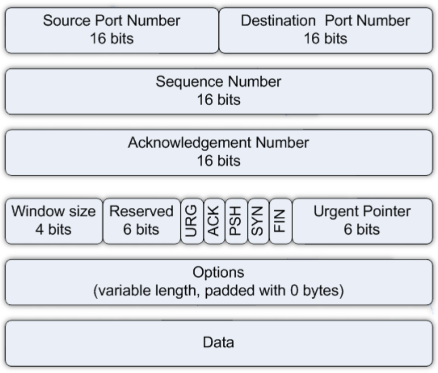


Figure 3: TCP Datagram

3.2 Transfer Learning in Deep Neural Networks

Transfer learning involves taking a pre-trained model—typically trained on large-scale datasets—and adapting it to a new, often smaller, task. It is particularly useful when domain-specific labeled data is limited.

In this project, we use a **ResNet34** architecture pre-trained on ImageNet. This allows us to leverage generic image features learned from large natural image datasets and fine-tune the model for steganography detection.

Proposed solution

4.1 Image Model

The model architecture is based on **ResNet-34**, a residual neural network known for its depth and its ability to mitigate vanishing gradients through skip connections. This architecture is particularly suitable for steganalysis tasks due to its proven performance in image classification problems.

To enhance its capability in detecting subtle steganographic patterns, we adopted a transfer learning approach and augmented the ResNet-34 backbone with specialized components. These include **SRM-based convolutional filters** for capturing steganographic noise residuals, and a dedicated **High Frequency Path** to emphasize frequency-sensitive details often modified in stego images. These additions integrate seamlessly as part of the model's extended architecture and significantly improve its sensitivity to steganographic artifacts.

To tailor the ResNet-34 architecture to the steganalysis task, several modifications and enhancements were made:

– **Input Adaptation:** The first convolutional layer was adjusted to accept single-channel grayscale images instead of three-channel RGB inputs. This change aligns the model with the nature of the dataset used, which contains grayscale images.

– **Output Adaptation:** The final fully connected (FC) layer of ResNet-34 was replaced with a new layer that outputs four class probabilities. These correspond to the four categories of images in the dataset: clean, LSB stego, WOW stego, and HILL stego.

– **SRM Filter Layer:** A Spatial Rich Model (SRM) filter layer was introduced to improve the model's sensitivity to statistical artifacts and noise patterns typically introduced during steganographic embedding. These filters are designed to capture high-order residuals and edge-level inconsistencies.

– **High-Frequency Path Layer:** A high-frequency enhancement layer was added to extract fine-grained image details. Since steganography often affects high-frequency regions of an image, this layer helps the model detect minute changes that may not be visible in lower-frequency components.

These adaptations collectively enhance the model's ability to identify and classify stego images by focusing on the statistical and structural features that are most likely to be altered during the embedding process.

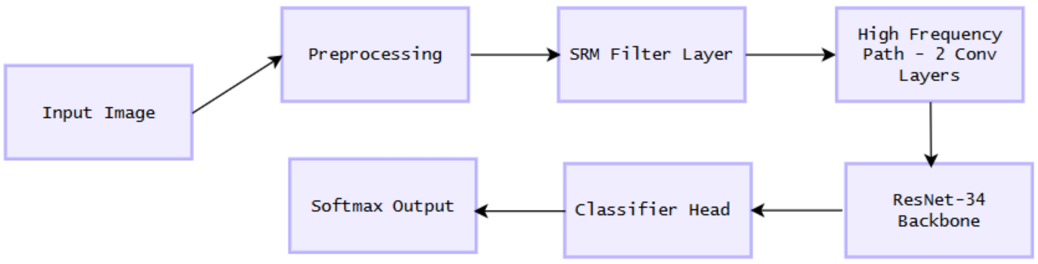


Figure 4: Image Model Architecture

4.1.1 Dataset Description : BOSSbase + LSB generated

The dataset used in this study consists of a total of 8,000 grayscale images, equally distributed across four distinct classes (cover , LSB , WOW , HILL) , with 2,000 images per class. Each image is resized to a uniform dimension of 256 × 256 pixels and has been manually labeled to reflect the type of steganographic content it contains.

The samples from classes cover , WOW and HILL are from the BOSSbase Dataset . Meanwhile, the samples from LSB class are generated from cover samples taken from BOSSbase with a python function that performs LSB . This balanced class distribution ensures that the model is trained on a diverse and representative dataset, which is essential for effective multi-class steganalysis.

4.2 Audio Model

For the Audio steganography detection model we heavily worked on feature extraction and on teaching some models such RandomForst or xgboost

4.2.1 Dataset Description: VStego800k

For the audio steganalysis model, we used VStego800k, a large-scale dataset specifically designed for steganalysis in streaming voice data. The dataset consists of low bit-rate audio files encoded in G.729a format, with both training and testing sets composed of short, 1-second audio clips. Each sample is either a clean audio file or a steganographic file embedded with hidden data using one of two algorithms—CNV-QIM or PMS—selected at random. The embedding rates vary between 10% and 40%, allowing for controlled evaluation across different levels of steganographic intensity. To ensure diversity, the dataset includes a balanced mix of male and female voices in both Chinese and English.

4.2.2 Initial feature extraction

We started by extracting some basic audio features that provide an overall overview of the signal such as

- **Mel-Frequency Cepstral Coefficient (MFCC):** a feature extremely used in automatic speech recognision, efficiently represents the overall shape or spectral enveloppe of the sound within short frames (10-20ms long). It computes the energy distribution across the spectrum and applies a series of Mel-scaled triangular filters which effectively group and sum energy within frequency bands that are perceptually more relevant to human hearing. After taking the logarithm of these band energies, a Discrete Cosine Transform (DCT) is applied

- **Spectral Centroid:** It is described as the center of mass of the spectrum it's calculated by this formula.

  $$\text{Spectral Centroid} = \frac{\sum_{k=1}^{N} f_k.S_k}{\sum_{k=1}^{N} S_k} \tag{1}$$

  where :

  - f_k *: the frequency of bin k*
  - S_k*: the magnitude of the spectrum at frequency* f_k
  - *N: total number of frequency bins*

  It helps detect the presence of steganography that introduces shift energy distribution across frequencies.

- **spectral Bandwidth:** measures the dispertion of the spectrum around its centroid.

  $$\text{Spectral Bandwidth}=\sqrt{\frac{\sum_{k=1}^{N}(f_k - \text{centroid}).S_k}{\sum_{k=1}^{N} S_k}} \tag{2}$$

- **Spectral Flatness:** quantifies how noise-like or tone-like a sound is. It is a useful indicator of spectral irregularities led by the CNV-QIM algorithm.

- **Zero Crossing rate:** the rate at which the sign of the audio signal changes.It helps detecting subtle changes in the waveform's periodicity due to pitch manipulation. but these features aren't enough for the classification.

4.2.3 Updated feature extraction

Since CNV-QIM directly alters the LPC domain, it was logical to extract LPC-based features. **Linear Predictive Coding (LPC)** is a widely used technique in digital audio signal processing, particularly for speech. Its primary purpose is to efficiently represent the spectral envelope of a speech signal in a highly compressed form.

It analyzes short segments of the speech wave to find a set of linear prediction coefficients. These coefficients allow for predicting the current speech sample as a linear combination of past samples as represented by the Fundemental LPC Prediction equation.

$$\hat{S}(n) = \sum_{k=1}^{p} a_k \hat{S}(n-k) \tag{3}$$

$\hat{S}(n)$: the predicted value of the speech signal at time n
$\hat{S}(n-k)$: the past samples of the speech signal
p: the order of the LPC model
a_k: the LPC coefficients

The resulting set of coefficients effectively describes the resonant characteristics (formants) of the vocal tract during that segment. Using an LPC order of 12, we computed the mean, variance, and delta (inter-coefficient differences) for each audio sample.

4.2.4 Updated Model:

Since we're working on wav files, the steganographic alterations are deluted and more subtle. To address this, we replaced the Random Forest classifier with **XGBoost**, which is known for its higher sensitivity to small feature variations and its ability to capture complex patterns in data.

4.3 Network Model Architecture

4.3.1 Flow-Based Detection Using Classical Machine Learning

The architecture for the network-based steganography detection model is grounded in classical machine learning, leveraging flow-based features extracted from packet captures (PCAPs) using CICFlowMeter. This choice is motivated by the structured nature of network traffic and the statistical anomalies introduced by steganographic embedding in TCP/IP header fields. Unlike the image model, which benefits from spatial representation, the network model focuses on flow-level behavioral patterns and statistical signatures.

The model pipeline consists of three stages: **flow generation**, **synthetic stego injection**, and **classification** using an ensemble tree-based model. The use of a **Random Forest classifier** is justified by its robustness, interpretability, and ability to handle high-dimensional tabular data without requiring feature scaling or complex architecture tuning. Additionally, this model serves as a lightweight yet effective baseline for real-time steganography detection in packet streams.

Stage 1: Flow Generation from PCAP using CICFlowMeter
Raw PCAP files are converted into structured flows using CICFlowMeter, a tool that extracts over 80 statistical features per bidirectional connection. These include metrics such as 'Flow Duration', 'Total Forward Packets', 'Average Packet Size', and inter-arrival times. This flow-based representation captures temporal and volumetric patterns at a granularity appropriate for detecting behavioral deviations caused by stego algorithms.

Stage 2: Steganographic Injection and Labeling
To simulate stego traffic, we programmatically modify selected TCP header fields—such as Sequence Number, Window Size, URG Flag, and Header Length—in a subset of flows, while ensuring protocol compliance. These injected samples are labeled as *Stego (1)*, and the rest as *Benign (0)*, forming a balanced binary classification dataset. The manipulation is subtle and preserves flow validity, mimicking real-world steganographic techniques that avoid detection by signature-based firewalls.

Stage 3: Classification with Random Forest
After cleaning and preprocessing the dataset (including NaN handling, correlation-based feature pruning, and normalization), a Random Forest classifier is trained. We tuned hyperparameters such as tree depth, number of estimators, and split criteria via grid search. The model is trained using 5-fold cross-validation to prevent overfitting and evaluated on metrics such as Accuracy, Precision, Recall, F1-score, and ROC-AUC. Feature importance analysis reveals that TCP-specific metrics (like 'Init Win bytes forward' and 'URG Flag Count') play a critical role in detecting anomalies.

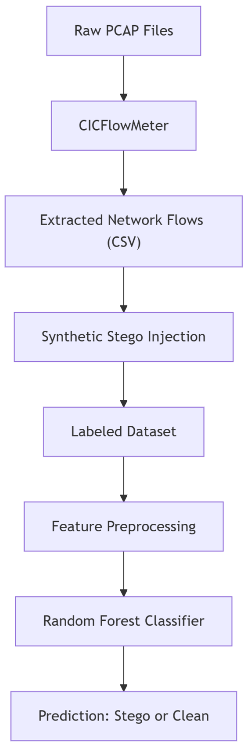


Figure 5: Network Steganography Detection Model Pipeline

4.3.2 Dataset Description: CICFlowMeter + CICIDS2017 Traffic

The dataset used for this model is derived from the CICIDS2017-Thursday morning traffic PCAP files, processed via CICFlowMeter to extract flow-based statistical features. The resulting CSV file contains network flows with labeled classes, including both benign and attack traffic (e.g., Brute Force, DoS). To adapt this dataset for steganalysis, we manually injected stego-like modifications into TCP headers for selected benign flows, creating a labeled binary classification task: Stego (modified) and Benign (unmodified). Class balance was maintained to ensure fair learning and unbiased evaluation. The final dataset contains ∼40,000 labeled flows, split into training and testing sets using stratified sampling.

Implementation and Results

5.1 Image Model

5.1.1 Workflow during training

Phase 1: Baseline Training

In Phase 1, a ResNet34 model, pre-trained on ImageNet, is adapted for steganalysis to classify grayscale images into four classes: Cover, LSB, WOW, and HILL. The first convolutional layer is modified to handle single-channel input, with weights initialized by averaging the original RGB weights and applying Kaiming normalization. The classifier head is replaced with a custom sequence including dropout (0.5 and 0.3), a 256-unit linear layer, batch normalization, LeakyReLU (0.1), and a final linear layer for four-class output. Only the final ResNet block (layer4) and classifier head are trainable, optimized using AdamW with learning rates of 3e-5 and 1e-4, respectively, and a cosine annealing scheduler. Training runs for up to 10 epochs with early stopping after 7 epochs without improvement, using CrossEntropyLoss, mixed precision via GradScaler, and gradient clipping. Performance is evaluated on validation data using accuracy, F1-score, AUC, and Cohen's Kappa, with the best model saved based on validation accuracy.

Phase 2: Full Attention Fine-tuning

In Phase 2, the Phase 1 ResNet34 model is augmented with a Spatial Rich Model (SRM) layer and a High-Frequency Path (HighFreqPath) to enhance steganalysis for classifying grayscale images into Cover, LSB, WOW, and HILL classes. The SRM layer uses fixed convolutional filters (Laplacian, Gabor, Hybrid Edge) to extract steganographic features, while the HighFreqPath employs trainable convolutions for high-frequency patterns. The ResNet34's first convolutional layer is modified to accept five channels (three from SRM, two from reduced HighFreqPath), and a new classifier head combines ResNet features (512 dimensions) with HighFreqPath features (3136 dimensions). Only the SRM layer, HighFreqPath, first convolution, layer4, and classifier are trainable, optimized with AdamW (learning rates: 1e-4 for SRM/HighFreqPath, 1e-5 for first convolution/layer4, 5e-4 for classifier) and a cosine annealing scheduler. Training proceeds for up to 15 epochs, with early stopping after 7 epochs without improvement, using CrossEntropyLoss, mixed precision, and gradient clipping. Performance is assessed with accuracy, F1-score, AUC, and Cohen's Kappa, saving the best model based on validation accuracy.

5.1.2 Results and interpretations

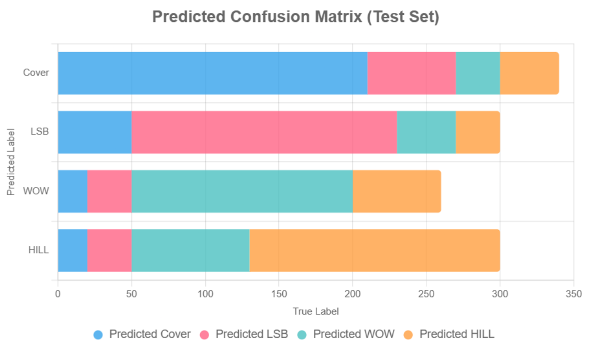


Figure 6: Confusion Matrix

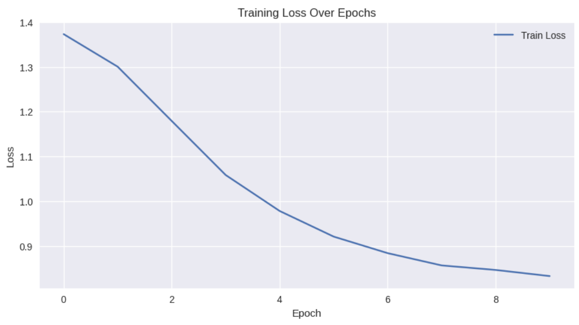


Figure 7: Training Loss over Epochs

The steganalysis model, a modified ResNet34 architecture enhanced with Spatial Rich Model (SRM) filters and a High Frequency Path (HighFreqPath) module, achieved a test set accuracy of approximately 59% on a balanced dataset of 8,000 grayscale images (2,000 per class: Cover, LSB, WOW, HILL). This indicates moderate classification performance and highlights the difficulty in distinguishing between various steganographic techniques.

The model demonstrates the highest performance on **Cover** images, with an F1 score ranging from approximately 0.65 to 0.70. This is likely due to the absence of embedded artifacts in cover images, which makes them more distinguishable. Performance on the **LSB** (Least Significant Bit) class is reasonably good as well, with F1 scores between 0.60 and 0.65, attributed to its detectable pixel-level modifications.

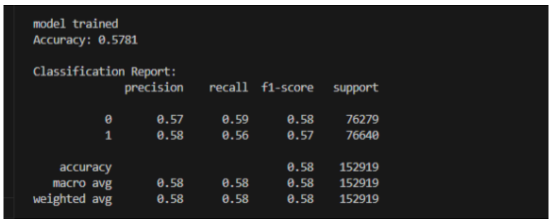
However, the model struggles significantly with the **WOW** (Wavelet Obtained Weights) and **HILL** (High-pass, Low-pass, and Local prediction) steganographic methods. The F1 scores for WOW range from 0.50 to 0.55, and for HILL from 0.45 to 0.50. These methods use adaptive embedding schemes designed to minimize detectable statistical distortions, which makes accurate classification more difficult.

Analysis of the confusion matrix reveals several key insights:

- **Cover** images are occasionally misclassified as **HILL**, likely due to HILL's subtle embedding strategy.
- **WOW** and **HILL** are frequently confused with each other, as both employ wavelet-based and adaptive embedding techniques that result in similar statistical signatures.

These findings suggest that while the incorporation of SRM filters and the HighFreqPath module does improve feature extraction, the model still requires further refinement. Future improvements could involve more sophisticated feature representations, additional data augmentation, or ensemble learning techniques to enhance differentiation between complex steganographic methods.

5.2 Audio Model



```
model trained
Accuracy: 0.5781

Classification Report:
              precision    recall  f1-score   support

           0       0.57      0.59      0.58     76279
           1       0.58      0.56      0.57     76640

    accuracy                           0.58    152919
   macro avg       0.58      0.58      0.58    152919
weighted avg       0.58      0.58      0.58    152919
```

Figure 8: The model's stats

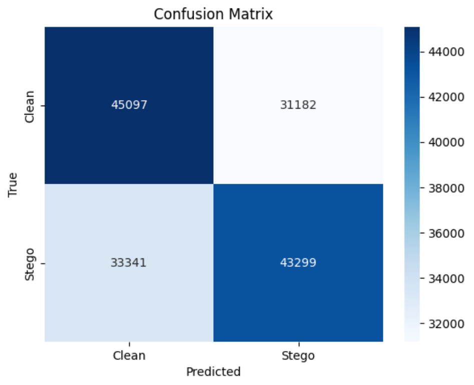


Figure 9: the confusion matrix

These are the best results we could achieve. The model seems to guess slightly better than totally at random, reaching an accuracy of 0.57. The classificaiton report shows relatively balanced precision and recall for both classes, with a slight advantage in detecting clean files (precision: 0.56, recall: 0.58) over stego files (precision: 0.57, recall: 0.54)

5.3 Network Model Performance and Evaluation

The Random Forest-based steganalysis model for network traffic achieved a high classification accuracy of **94.57%**, highlighting its effectiveness in identifying steganographic patterns embedded within TCP header fields. The model was trained on a balanced dataset containing clean and synthetically injected stego traffic, and evaluated using standard classification metrics.

Precision and **Recall** scores show that the model is highly capable of distinguishing between clean and steganographic flows:

- For the **Clean** class:
  - Precision: **0.92**
  - Recall: **0.98**
  - F1-score: **0.95**
- For the **Stego** class:
  - Precision: **0.98**

  – Recall: **0.91**
  – F1-score: **0.94**

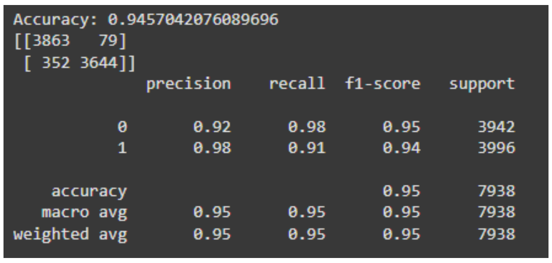
The high precision for the **Stego** class indicates that the model rarely misclassifies clean flows as steganographic (i.e., few false positives), while the high recall for the **Clean** class demonstrates its ability to correctly identify the majority of clean traffic.

Confusion Matrix Analysis

- Out of 3942 clean flows, only **79** were misclassified as stego, indicating excellent specificity.
- Out of 3996 stego flows, **352** were misclassified as clean, showing that the model still struggles slightly with steganographic traffic designed to preserve normal statistical patterns.
- The confusion between clean and stego traffic remains relatively low, but the false negatives in the stego class suggest that more subtle or sophisticated embedding techniques (e.g., minimally perturbing sequence numbers or using rarely monitored header fields) may evade detection.

Macro vs. Weighted Averages

Both the macro and weighted averages for precision, recall, and F1-score are at **0.95**, further confirming the model's balanced performance across both classes. This is crucial in steganalysis, where the cost of false negatives (missed stego) is often as critical as false positives.



```
Accuracy: 0.9457042076089696
[[3863   79]
 [ 352 3644]]
              precision    recall  f1-score   support

           0       0.92      0.98      0.95      3942
           1       0.98      0.91      0.94      3996

    accuracy                           0.95      7938
   macro avg       0.95      0.95      0.95      7938
weighted avg       0.95      0.95      0.95      7938
```

Figure 10: The model's stats

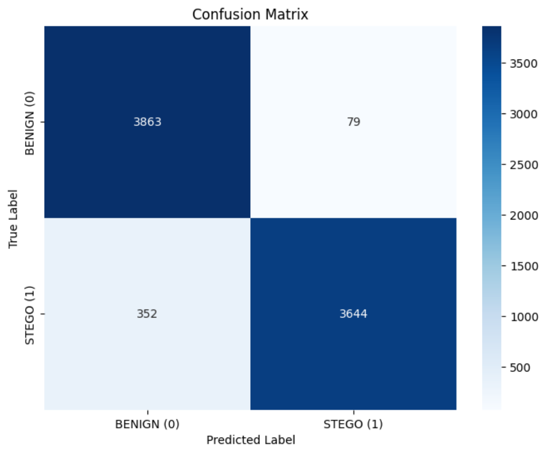


Figure 11: the confusion matrix

5.4 Application Interface and Workflow

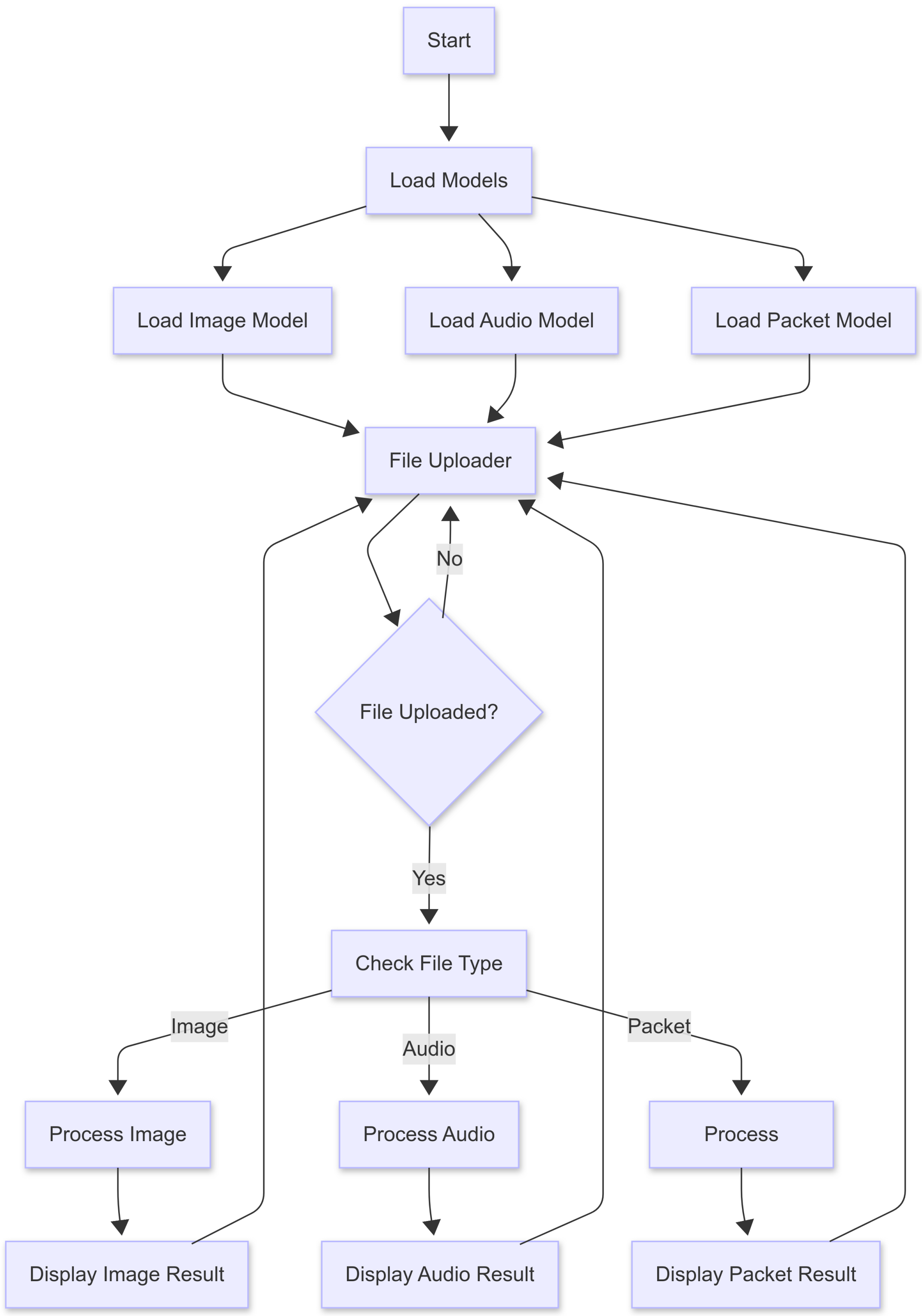
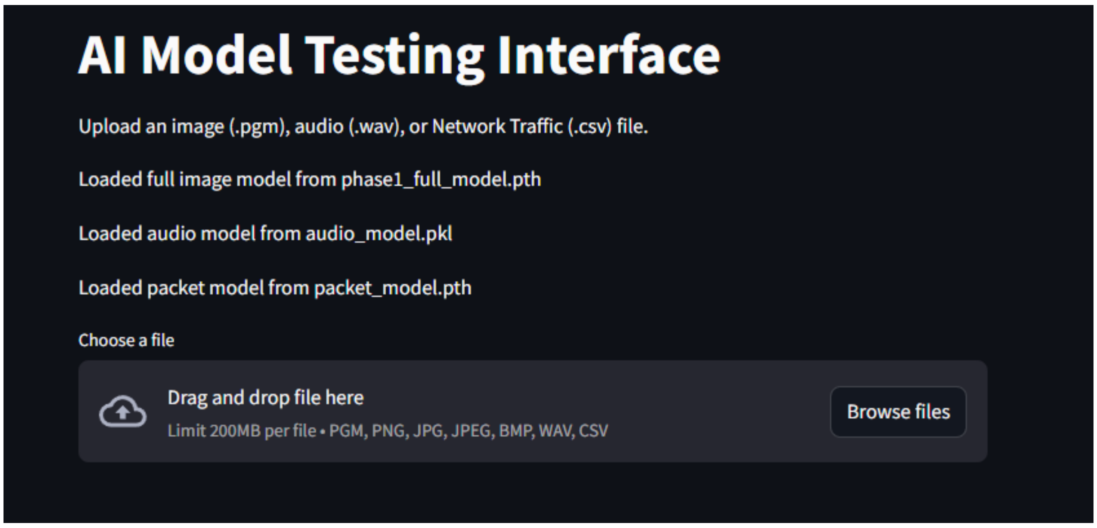


Figure 12: Application Workflow

5.5 Application Interface and Workflow


AI Model Testing Interface

Upload an image (.pgm), audio (.wav), or Network Traffic (.csv) file.

Loaded full image model from phase1_full_model.pth

Loaded audio model from audio_model.pkl

Loaded packet model from packet_model.pth

Choose a file

Drag and drop file here

Limit 200MB per file • PGM, PNG, JPG, JPEG, BMP, WAV, CSV

Figure 13: Application Workflow

The AI Model Testing Interface, built with Streamlit, is a minimalist and user-friendly platform designed for uploading and testing various data files—images (e.g., .pgm, .png, .jpg, .jpeg, .bmp), audio (.wav), or network traffic (.csv)—against preloaded machine learning models . It features a straightforward drag-and-drop or browse upload mechanism with a 200MB file size limit, making it ideal for testing the steganalysis models on new files.

Discussions and potential improvements

6.1 Audio Model

The audio model currently performs poorly, likely due to the conversion from .g729a to WAV, which weakens the already subtle steganographic traces. To improve detection, several strategies can be explored:

- Extract higher-order statistical features that are more sensitive to subtle modifications.
- Analyze inter-frame correlations, as steganography often operates frame-by-frame.
- Study inter-feature relationships, which may be disrupted by embedding.
- Use more complex models like 2D CNNs applied to spectrograms or Mel-spectrograms. These capture time-frequency patterns and can learn steganographic cues automatically, bypassing some limitations of hand-crafted features.

6.2 Image Model :

The steganography detection model shows promising results but there is a lot of room for improvement and can be enhanced for better performance while managing computational costs. Below are key improvements and strategies for efficient deployment.

Model Improvements

- **Gradual Unfreezing of ResNet34 Layers**: Unfreeze earlier layers (e.g., layer3, layer2) to fine-tune deeper features, improving detection of subtle stego patterns, especially for WOW and HILL classes.
- **Spatial and Channel Attention Blocks**: Add CBAM or SE blocks to focus on critical image regions and channels, enhancing detection of faint stego artifacts.
- **Patch-Based Feature Extraction**: Divide images into patches for detailed analysis, boosting accuracy for subtle embeddings at higher computational cost.

Computational Optimization

- **Cloud Deployment**: Use cloud servers (e.g., AWS, Google Cloud) with GPU/TPU support for efficient training and inference.
- **Optimization Techniques**: Leverage mixed-precision training, pruning, or quantization to reduce memory and speed up processing.
- **Batch Size Tuning**: Optimize batch sizes (starting from 32) for memory and stability.

6.3 Network Model

The network-based steganalysis model demonstrates strong performance, achieving an accuracy of 94.57% when classifying between clean and stego-injected network flows. However, there are both practical challenges and enhancement opportunities to consider.

Limitations and Areas for Improvement

- **False Negatives in Stego Class**: The model tends to miss a portion of stego traffic (recall = 0.91), especially when the embedding strategy minimally alters packet fields.
- **Static Feature Limitations**: Current detection relies heavily on handcrafted features from TCP headers due to *the lack of open source ressources*, which may not capture deeper contextual or sequential anomalies.

Future Enhancements

- **Sequence-Aware Models**: Incorporate LSTM or Transformer-based architectures to analyze temporal patterns in packet flows, improving detection of low-rate or adaptive steganography.
- **Feature Expansion**: Include features from other layers (e.g., IP or application) or statistical aggregations over multiple packets to enrich context.
- **Ensemble Learning**: Combine multiple models (e.g., Random Forest + XGBoost + DNN) to improve robustness across varying stego techniques.

6.4 Conclusion

These enhancements, paired with cloud-based deployment and optimization, will improve accuracy and scalability for steganography detection.

Conclusion

This project explored machine learning approaches for detecting steganographic content in audio, image, and network data. Across the three domains, we developed specialized models tailored to the nature of the embedded information.

The Random Forest classifier for synthetically modified network packets produced strong and reliable results. While the ResNest-based model applied to the image steganalysis and the audio model showed decent and promising performance.

All three models were integrated into a Streamlit application that automatically detects the file type and routes it to the appropriate model.

Overall, the project demonstrates the potential of machine learning in steganalysis across multiple data types, while highlighting the need for further refinement for real life applications.

References

1 https://www.researchgate.net/publication/3343272_Steganalysis_of_LSB_matching_in_grayscale_images

2 https://ieeexplore.ieee.org/document/6197267

3 https://www.sciencedirect.com/science/article/abs/pii/S138904171930511X

3 https://www.researchgate.net/publication/354122133_Ocular_Disease_Detection_Using_Advanced_Neural_Network_Based_Classification_Algorithms

4 https://pytorch.org/hub/pytorch_vision_resnet/

5 https://www.researchgate.net/publication/221286432_An_Approach_to_Information_Hiding_in_Low_Bit-Rate_Speech_Stream

6 https://taylorandfrancis.com/knowledge/Engineering_and_technology/Engineering_support_and_special_topics/LPC/

7 https://www.mdpi.com/1999-5903/12/1/17

8 https://ieeexplore.ieee.org/abstract/document/8943350

9 https://colab.ws/articles/10.1007%2F978-3-031-53488-1_37?utm_source=chatgpt.com

10 https://medium.com/%40darshan.nere/introduction-to-network-steganography-14c2bbecc609

11 https://www.sciencedirect.com/science/article/abs/pii/S0957417424016634?utm_source=chatgpt.com

12 http://cicresearch.ca/