Embedding Data inside Neural Network Network Bias Model: A Steganographic Approach Using Keras Models

Harvi M. Zakhir

ABSTRACT

Steganographic technique that embeds data into the bias parameter of a Keras neural network model. By encoding data to ASCII code, we can insert the numbers as a floating point inside the bias values where each value contains one character of the desired data. The method requires no encryption and leverages the structural legitimacy of HDF5 model files to evade detection

INTRODUCTION

Steganography is concealing information within an innocuous carrier such as images, audio, text, or network protocols so that the very existence of the hidden message remains undetected. In the era of artificial intelligence, machine learning models themselves have emerged as potential carriers for covert data. Unlike traditional steganographic media, deep learning models are increasingly exchanged, deployed, and shared across platforms, making them ideal yet overlooked vectors for hidden communication or digital fingerprinting.

In this work, we demonstrate a proof-of-concept steganographic technique that embeds secret messages directly into the bias parameters of a Keras neural network model saved in the HDF5 format. The method requires no encryption or external metadata; instead,

ASCII-encoded text is written as floating-point values into the model's bias vectors. While the resulting model appears structurally valid and can be loaded by standard frameworks, it produces meaningless predictions confirming that the model serves solely as a data container.

This experiment originated from a Capture-The-Flag (CTF) challenge but evolved into a deliberate exploration of model-based steganography. Although the current implementation uses a non-functional model, it lays the groundwork for future research on functional steganographic models where prediction accuracy and hidden data coexist.

METHODOLOGY

This paper introduces Hidden Bias Steganography (HBS), a steganographic technique for embedding secret messages directly into the bias parameters of a feedforward neural network. The core idea of HBS is to treat the bias vector of a hidden layer as a covert storage medium, leveraging the fact that bias values are rarely inspected in practice and do not affect the model's structural integrity when modified.

The secret payload such as a digital fingerprint, ownership tag, or authentication token is first encoded into a numerical sequence using the ASCII standard. For example, the string "Cats are cute" (13 characters, including spaces) yields the integer vector:

$$\mathrm{m} = [67, 97, 116, 115, 32, 97, 114, 101, 32, 99, 117, 116, 101].$$

A carrier model is constructed as a multilayer perceptron (MLP) with three input features (e.g., academic grade, attendance rate, and behavioral score), one or more hidden layers, and a binary output for graduation prediction. To embed an n -character message, the first hidden layer must contain at least n neurons, as each neuron contributes one bias parameter. Thus, the embedding capacity of HBS is strictly limited by the width of the chosen layer: a layer with h neurons can carry up to h bytes of data.

The model may be trained on synthetic or real data to mimic a deployed scenario; however, training is optional since HBS operates via direct parameter injection after initialization or after training. The embedding process replaces the first n elements of the bias vector $b^{(1)}$ in the first hidden layer:

$$b_i^{(1)} \leftarrow m_i \quad \text{for } i = 1, 2, \ldots, n.$$

The modified model is then saved in HDF5 (.h5) format, preserving the hidden payload within its serialized parameter space. Extraction is performed by loading the

model, reading the first n bias values of the agreed-upon layer, and decoding them from ASCII to plaintext.

Design Considerations and Limitations

HBS is designed primarily for static model distribution, such as sharing pre-trained models for inference. It is not robust against parameter updates: any fine-tuning, retraining, or pruning will likely overwrite or distort the embedded message. Additionally, because raw ASCII values (e.g., 65–122) are significantly larger than typical bias magnitudes (often initialized near zero), the embedding may:

Introduce numerical anomalies detectable via statistical analysis,
Degrade model performance if the modified biases critically affect decision boundaries.
Despite these limitations, HBS serves a practical purpose in digital fingerprinting: a model owner can embed a unique identifier (e.g., user ID or license key) into each distributed copy. If the model is later leaked or misused, the fingerprint can be extracted to trace its origin—without altering the model's file structure or requiring metadata.