Ratcheted Steganography Using Generative AI

Fritz Hain, Britta Hale, and Richard Thompson

Forward Secrecy and Post-Compromise Security

Forward Secrecy (FS): Previous states remain secure even if the current state of a communicating party is compromised.

Post-Compromise Security (PCS): Self-healing property, assuming adversarial compromise of the secret state of a party, secrecy of future states can be restored under certain conditions.

Confidentiality and Covertness

- **Confidentiality**: An adversary knows something is being said but does not know what it is.
- **Covertness**: An adversary does not know that private conversation is even happening.

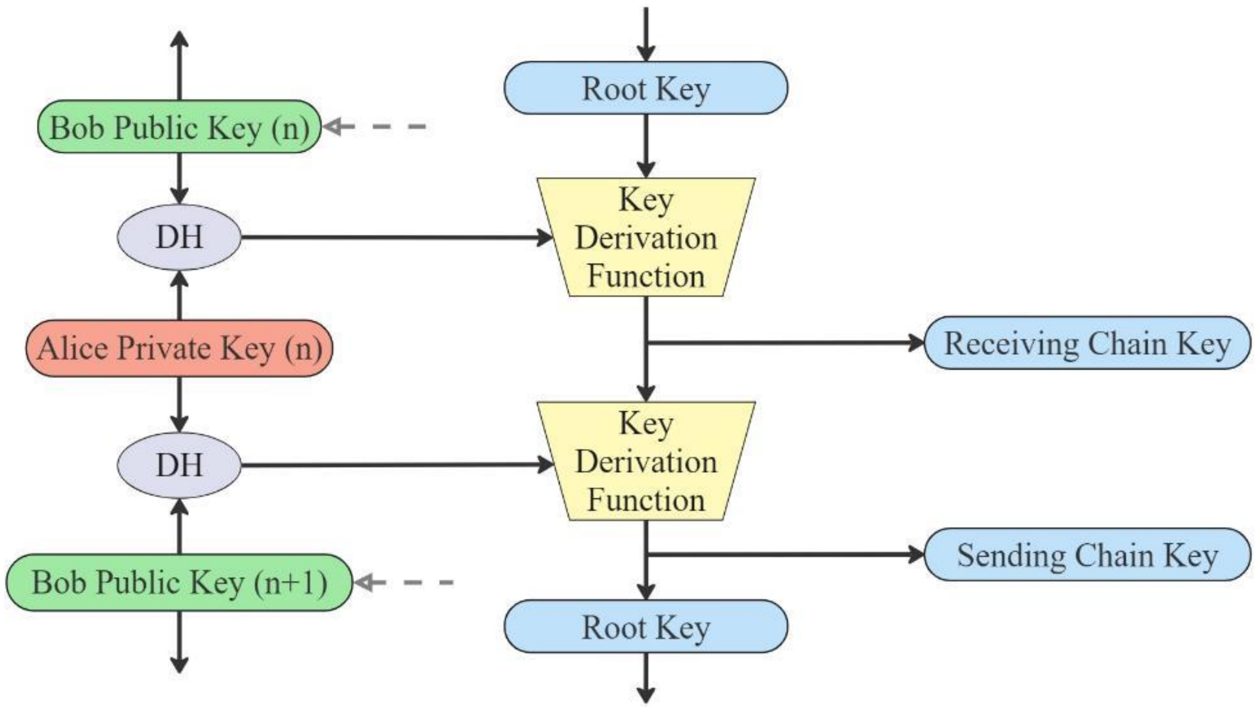
Different uses:
Steganography – covert
Encrypt – confidential

Forward Covertness and Post-Compromise Covertness

Forward Covertness (FC): If the existence of a steganographic message is detected or its embedding method becomes known, the existence of an embedded message in previously sent covers remains undetectable.

Post-Compromise Covertness (PCC): Self-healing property, if a current embedded message is detected or its embedding algorithm becomes known, future message covertness can be restored

Signal Protocol Overview

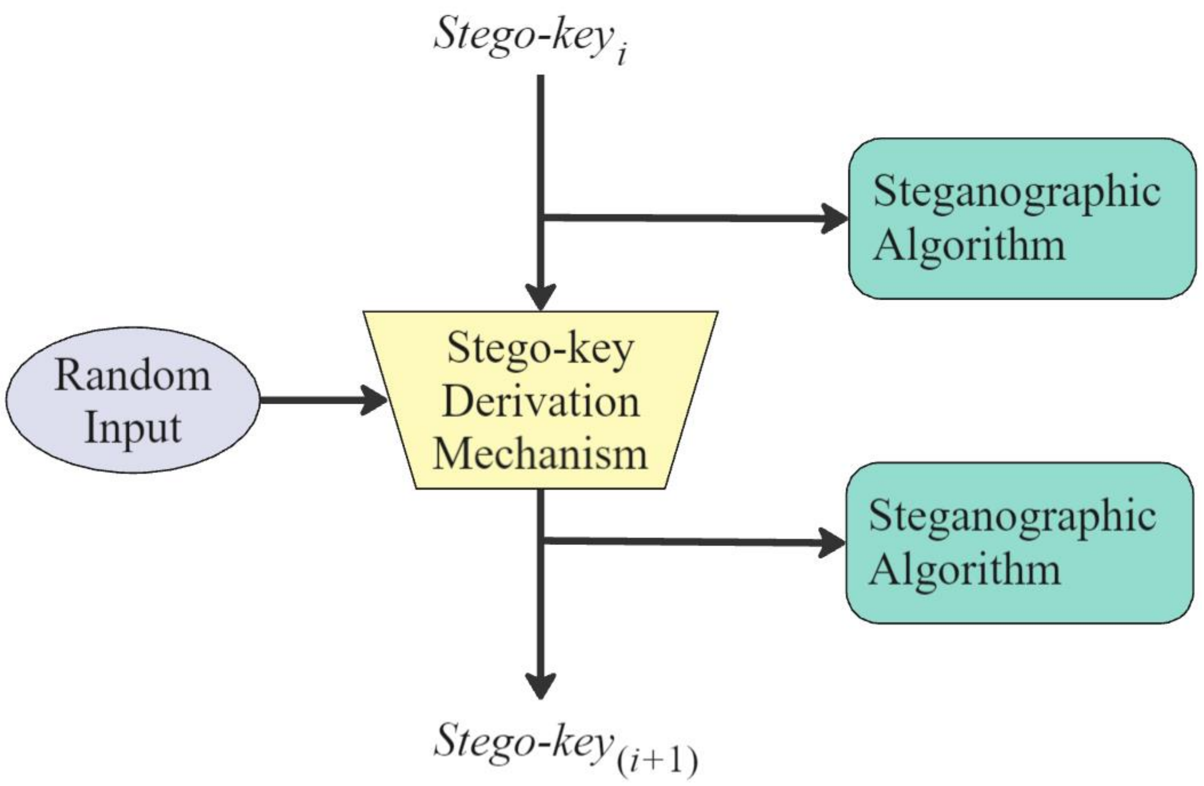


Key Derivation Function (KDF) – One way, deterministic function, derives new keys.

Double Ratchet

- Symmetric Key
- Diffie Hellman (DH)

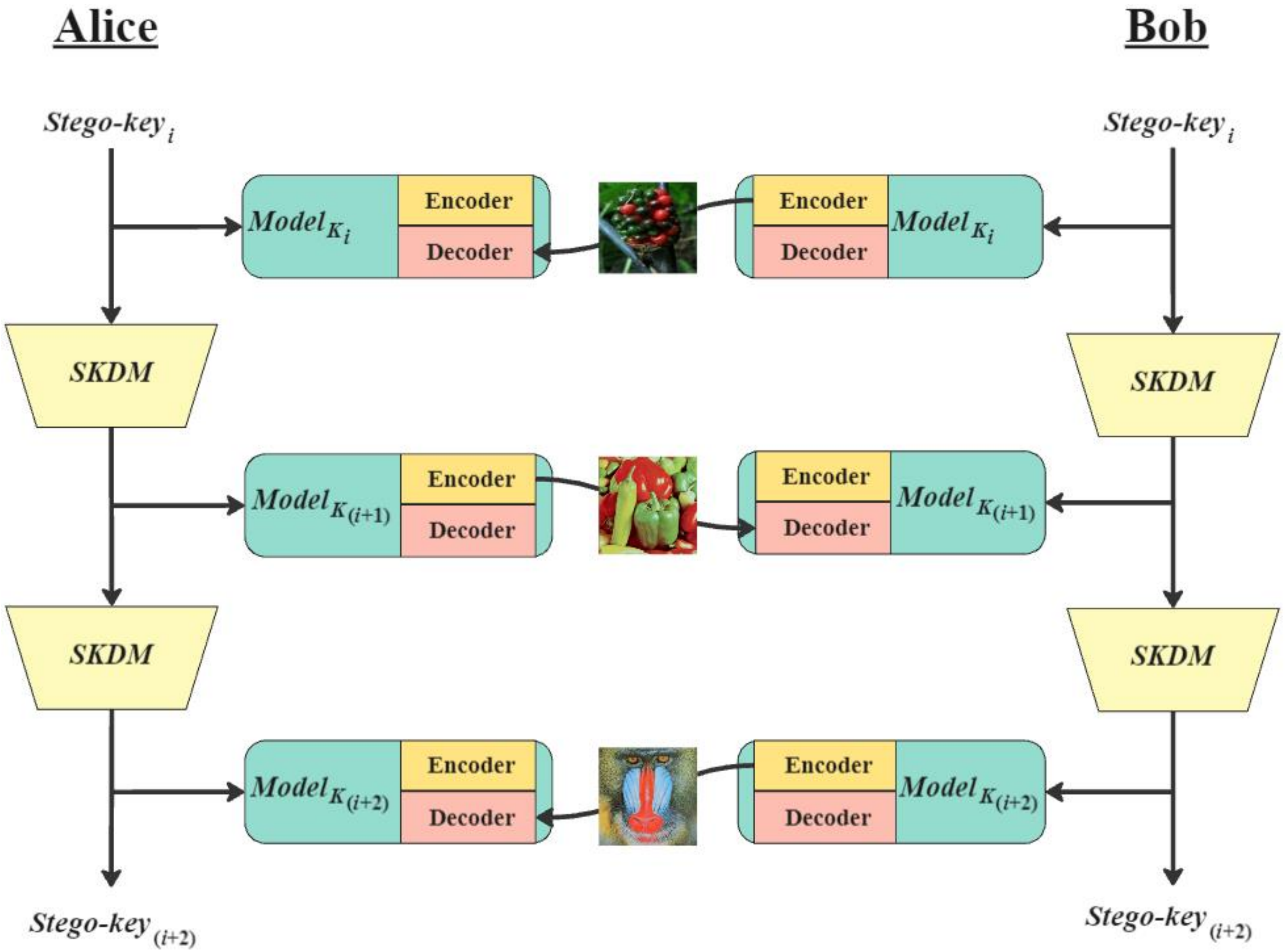
Definitions



***Stego-key*:** A set of machine learning model attributes held secret by the sender and receiver which define the model.

***Stego-key Derivation Mechanism* (*SKDM*):** a deterministic, one-way algorithm that takes as input a *stego-key* and outputs a new *stego-key*.

General Ratcheted Steganography Model with Machine Learning



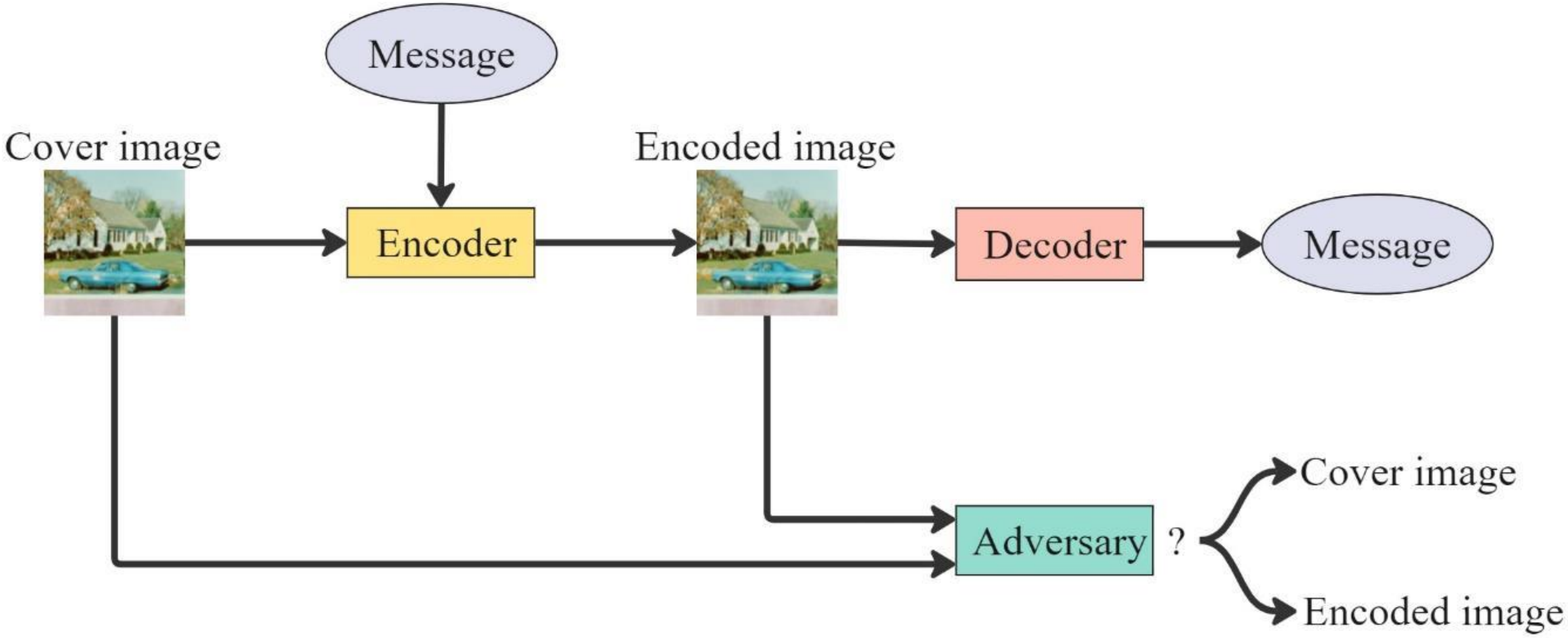
Machine Learning Steganography

Three neural networks:

- Encoder
- Decoder
- Adversary

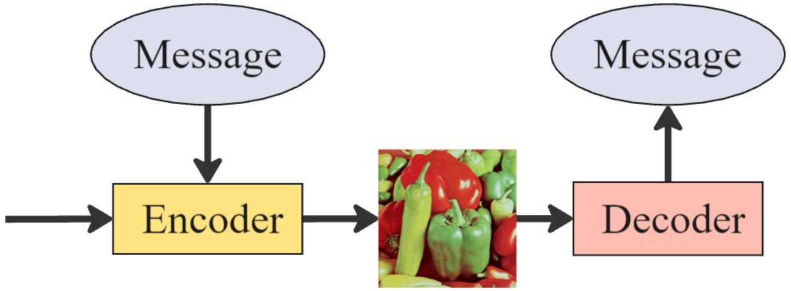
Minimize:

- Image Distortion
- Message Distortion
- Detectability



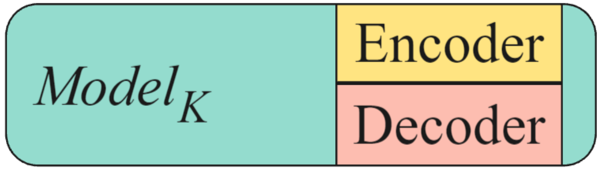
Definitions

Bit Error Rate (BER): Measures the accuracy of a decoder in a model. The total decoding errors divided by the total encoded bits. We have that $BER = n_e/n_b$.



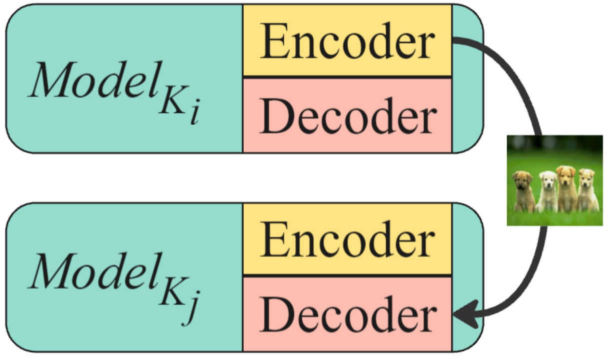
$$BER = \frac{n_e}{n_b}$$

Fully trained model: A model is fully trained if a decoder $Model_K^{dec}$ will extract a message msg with a $BER < .05$.



$$BER < .05$$

Model independency: $Model_{K_i}^{enc}$ is said to be independent of model $Model_{K_j}^{enc}$ if, for $Model_{K_j}^{dec}(stxt_i)$, we have average $BER > .45$.



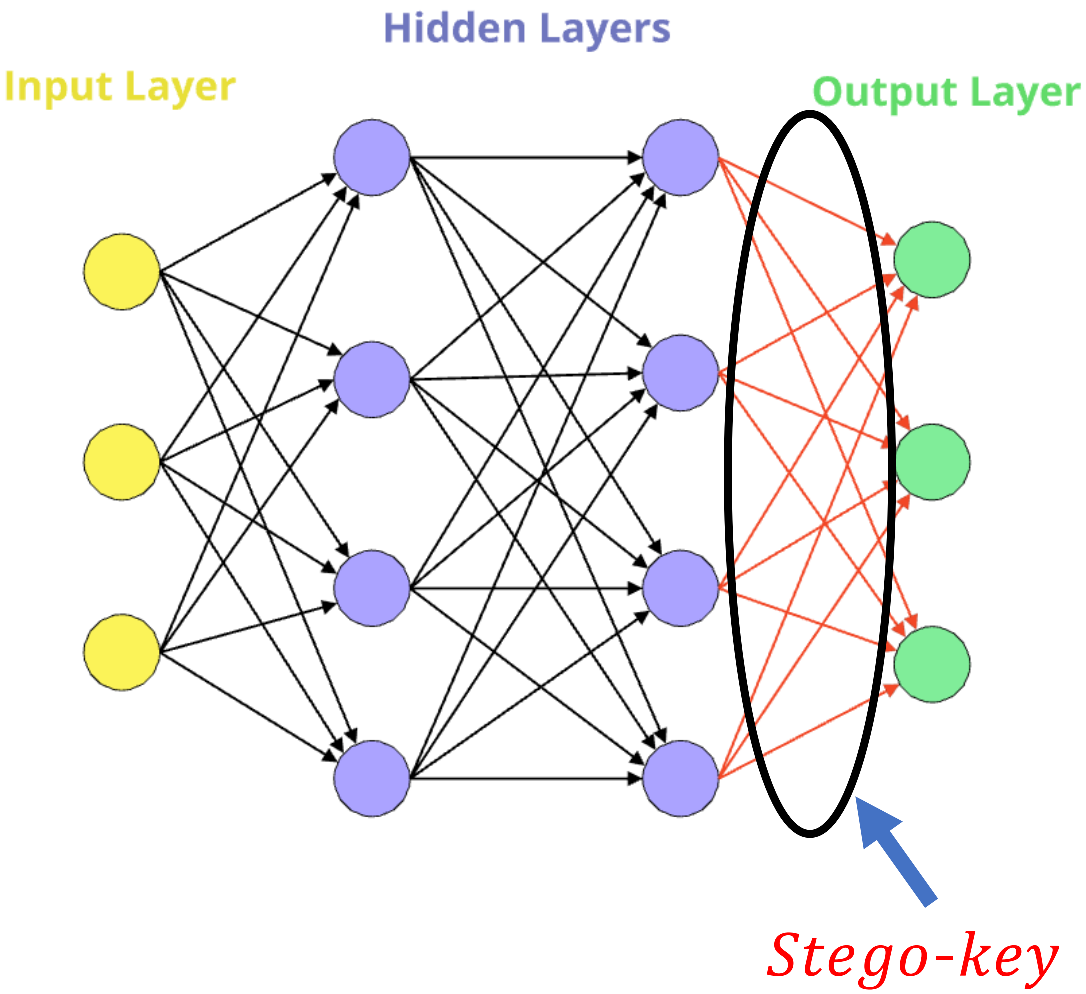
$$BER > .45$$

Randomizer Ratchet - Construction

HiDDeN Model: Generative steganography framework created by Zhu et al*, consisting of an encoder $Model^{enc}$ and decoder $Model^{dec}$

Randomizer Ratchet *Stego-key*: The set of weights w that feed into the output layer of the encoder $Model_K^{enc}$ and decoder $Model_K^{dec}$ neural networks.

Randomizer Ratchet *SKDM*: Select new initial weights randomly within a margin of the weight average $\overline{w_i}$ of weights in $stego\text{-}key_i$.



* Zhu, J., Kaplan, R., Johnson, J., Fei-Fei, L.: Hidden: Hiding Data With Deep Networks. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 657–672 (2018)

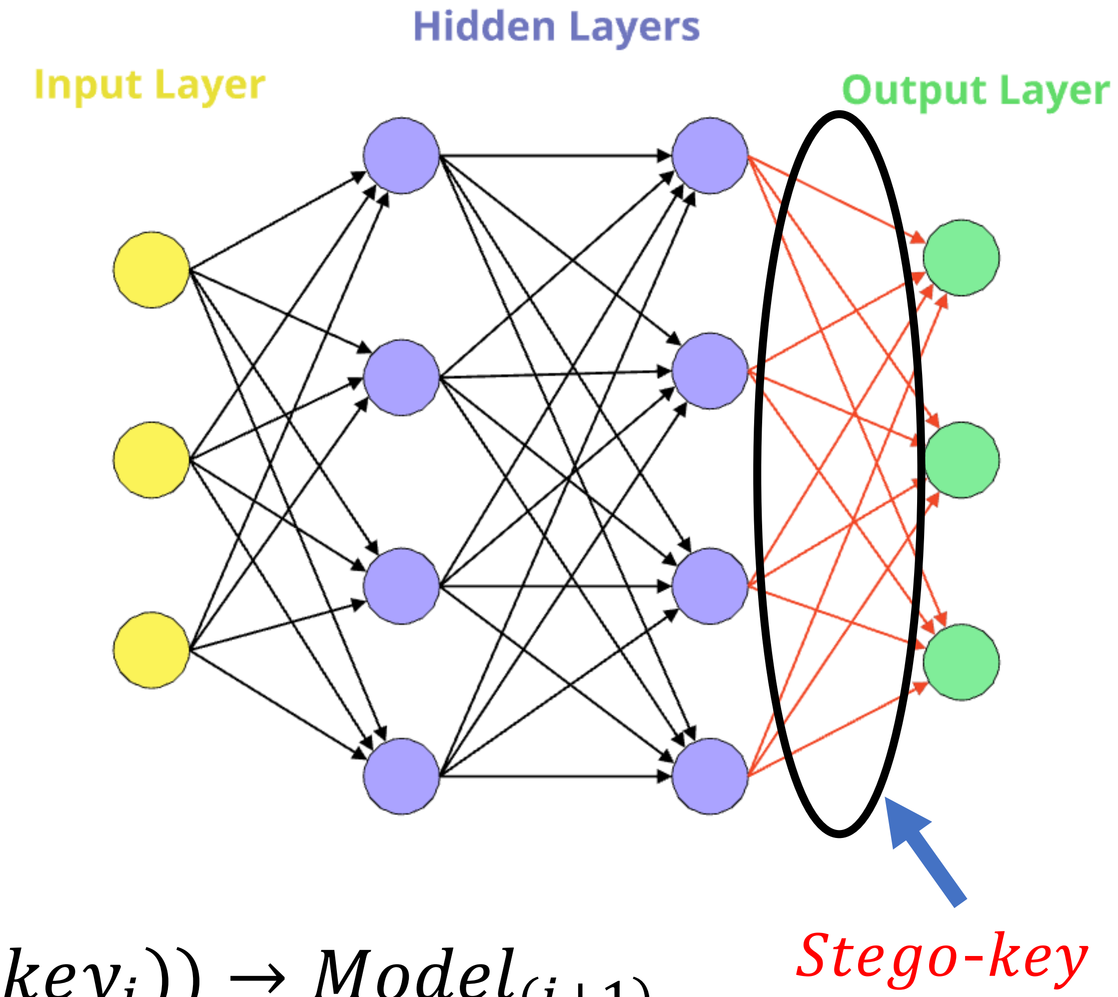
Randomizer Ratchet - Construction

1. Begin with trained model, $Model_{K_i}$.

2. Permanently set all weights in $Model_{K_i}^{enc}$ and $Model_{K_i}^{dec}$ as constant and immutable, except for the $stego\text{-}key$ weights.

3. Pass $stego\text{-}key_i$ through the Randomizer $SKDM$ to obtain new initial weights.

4. Perform additional training with on $Model_{K_i}$ with new weights, modifying only weights that feed into the output layer to obtain $Model_{K_{i+1}}$.



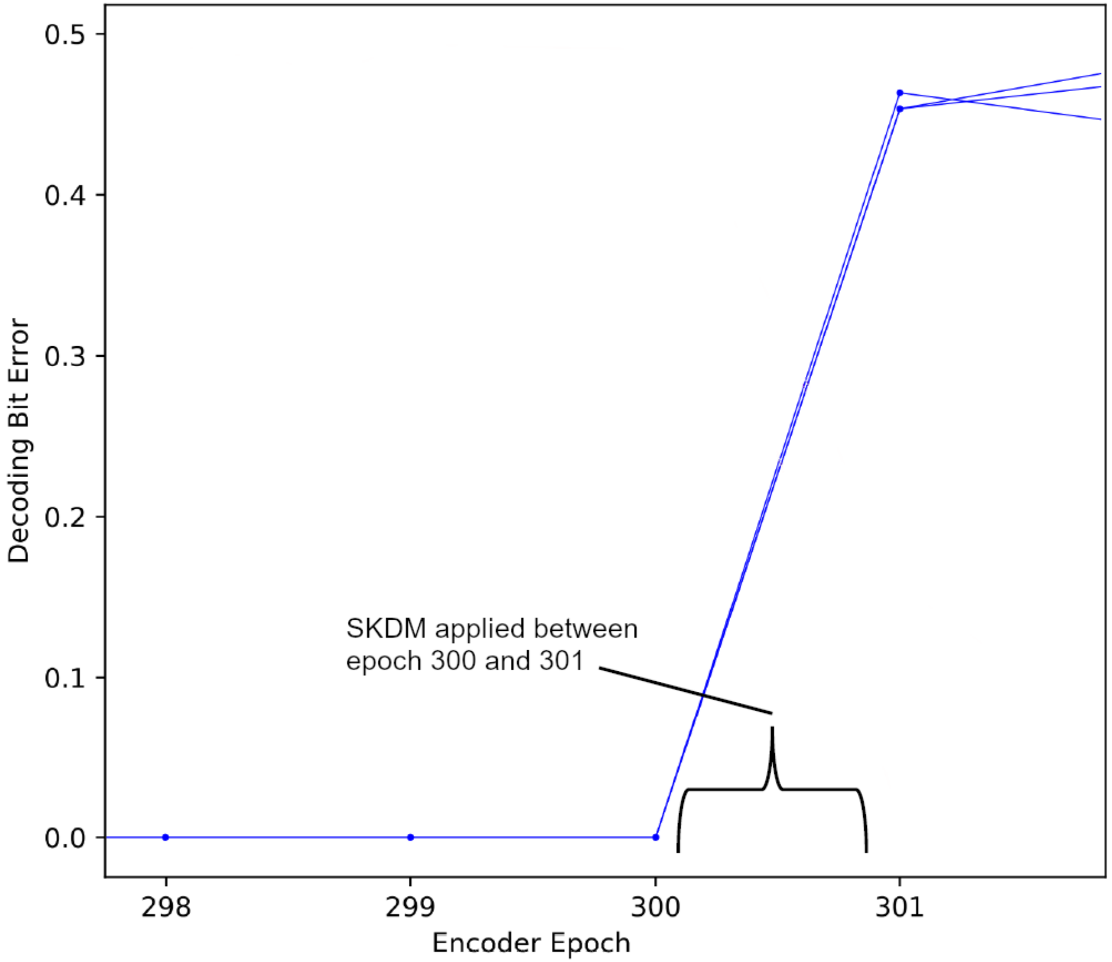
$$\mathrm{F}(Model_i, SKDM(stego\text{-}key_i)) \rightarrow Model_{(i+1)}$$

Randomizer Ratchet - Experiments

#	Experiment	Description	Observation
1.1	Ratcheting Feasibility	Apply a Randomizer $SKDM(stego\text{-}key)$ to a fully trained model	Observe if shifted out of model, i.e., $BER > .45$
1.2	Model Independence	Apply a Randomizer Ratchet to the same single base model 100 times: $\mathrm{F}(Model_0, SKDM(stego\text{-}key_0)) \rightarrow Model_1$	Measure model independence, i.e. Using $Model_{K_i}^{enc}$ and $Model_{K_j}^{dec}$, we have an average $BER > .45$
1.3	Ratcheting Limits	Sequentially apply a Randomizer Ratchet: $\mathrm{F}(Model_i, SKDM(stego\text{-}key_i)) \rightarrow Model_{i+1}$	Observe BER decay if any, i.e. if the $\overline{\triangle\, BER} > 0$

Randomizer Ratchet – Ratcheting Feasibility (1.1)

Apply a Randomizer $SKDM(stego\text{-}key)$ to a fully trained model

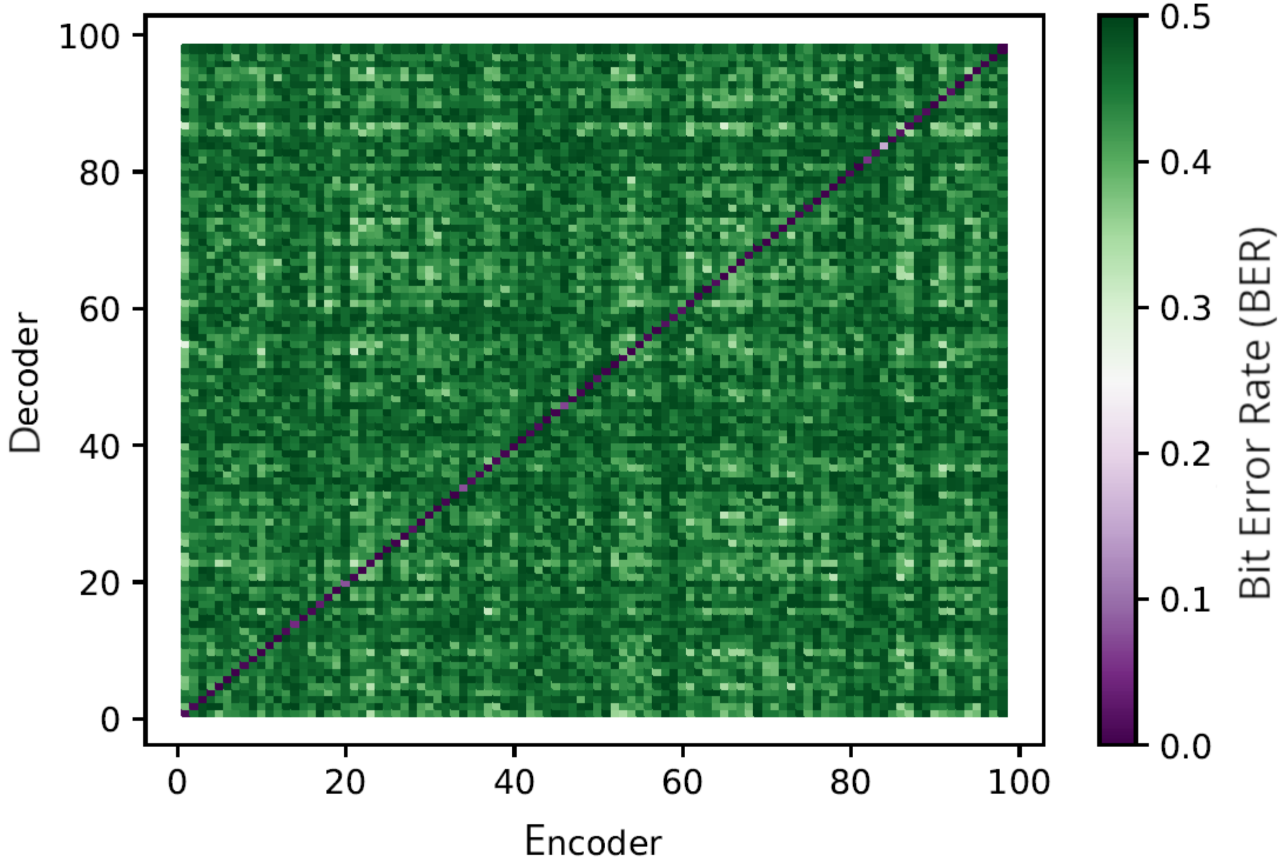


Randomizer Ratchet – Ratcheted Model Independence (1.2)

Apply a Randomizer Ratchet to the same single base model 100 times.

$$\mathrm{F}(Model_0, SKDM(stego\text{-}key_0)) \rightarrow Model_1$$

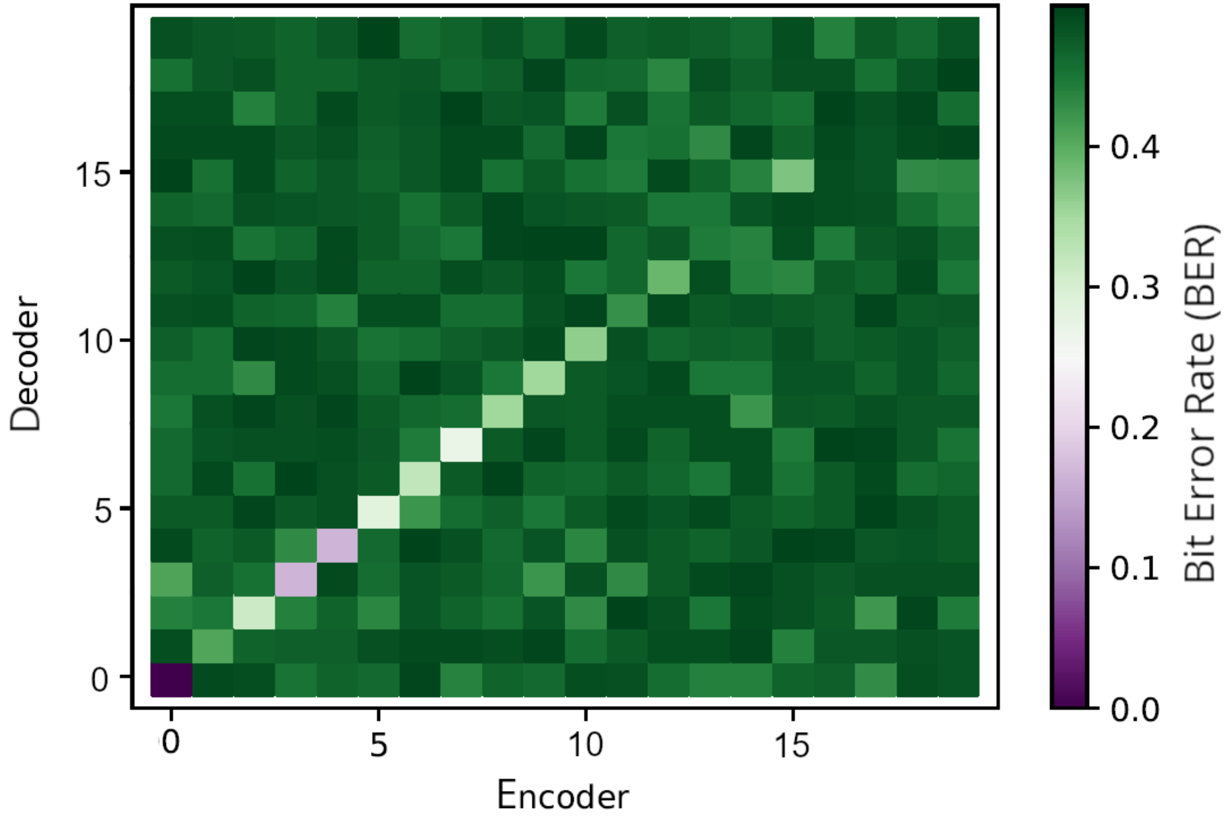
repeat for 100 tests – stego-key generation is non-deterministic



Randomizer Ratchet – Ratcheting Limits (1.3)

Sequentially apply a Randomizer Ratchet.

$$\mathrm{F}(Model_i, SKDM(stego\text{-}key_i)) \rightarrow Model_{i+1}$$



Steganalysis - Experiments

Purpose: Discover if certain bit positions are more likely to have errors.

Setup: Fully train **three** separate models.

Evaluation Metric - $BitErrors_l$: The number of decoding errors at bit position l is denoted by $BitErrors_l(Model_i^{enc}, Model_j^{dec})$, where the bit string was encoded with $Model_i^{enc}$ and decoded with $Model_j^{dec}$.

#	Experiment	Description	Observation
2.1	Bit error distribution baseline	Decode 1000 random messages ($msg \in \{0,1\}^{30}$) on all three decoders ($Model_i^{dec}$), repeated three times.	Sum of all decoding errors at each bit position, $BitErrors_l(\perp, Model_i^{dec})$.
2.2	Bit error distribution actual	For all three Encoders, encode 1000 random messages ($msg \in \{0,1\}^{30}$) with $Model_i^{enc}$ and then decode with all three decoders ($Model_j^{dec}$).	Sum of all decoding errors at each bit position, $BitErrors_l(Model_i^{enc}, Model_j^{dec})$.

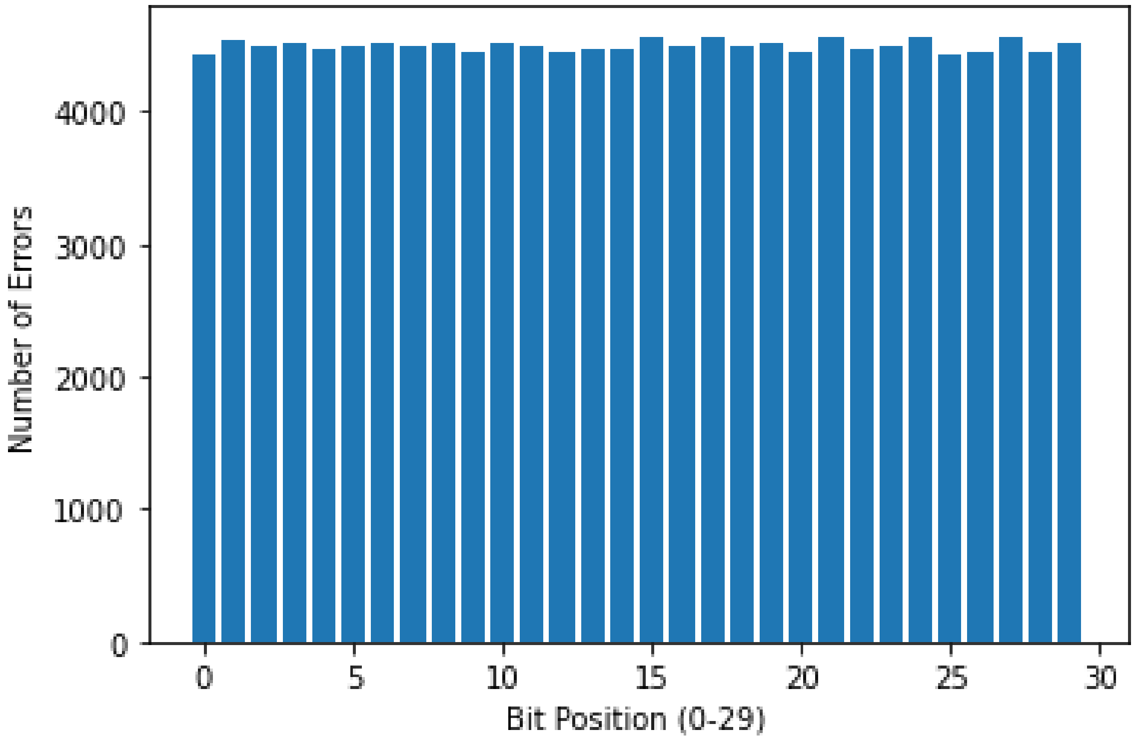
Steganalysis – Bit error distribution baseline (2.1)

Bit error distribution baseline across all l:

$$|sample| = 1000$$

$$NumErrors\ at\ bit\ position\ l =$$

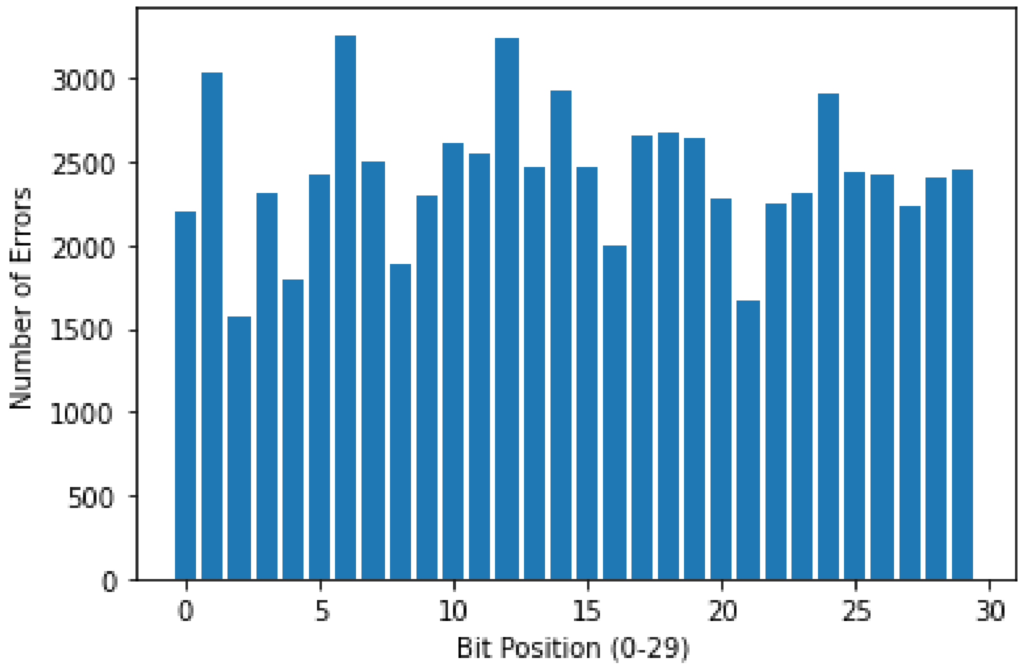
$$BitErrors_l(\perp, Model_0^{dec,\, sample\ 1}) +$$
$$BitErrors_l(\perp, Model_1^{dec,\, sample\ 1}) +$$
$$BitErrors_l(\perp, Model_2^{dec,\, sample\ 1}) +$$
$$BitErrors_l(\perp, Model_0^{dec,\, sample\ 2}) +$$
$$BitErrors_l(\perp, Model_1^{dec,\, sample\ 2}) +$$
$$BitErrors_l(\perp, Model_2^{dec,\, sample\ 2}) +$$
$$BitErrors_l(\perp, Model_0^{dec,\, sample\ 3}) +$$
$$BitErrors_l(\perp, Model_1^{dec,\, sample\ 3}) +$$
$$BitErrors_l(\perp, Model_2^{dec,\, sample\ 3})$$



Steganalysis – Bit error distribution actual (2.2)

Actual bit error distribution across all l:

$$
\begin{aligned}
&NumErrors\ at\ bit\ position\ l = \\
&\quad BitErrors_l\left(Model_0^{enc}, Model_0^{dec}\right) + \\
&\quad BitErrors_l\left(Model_0^{enc}, Model_1^{dec}\right) + \\
&\quad BitErrors_l\left(Model_0^{enc}, Model_2^{dec}\right) + \\
&\quad BitErrors_l\left(Model_1^{enc}, Model_0^{dec}\right) + \\
&\quad BitErrors_l\left(Model_1^{enc}, Model_1^{dec}\right) + \\
&\quad BitErrors_l\left(Model_1^{enc}, Model_2^{dec}\right) + \\
&\quad BitErrors_l\left(Model_2^{enc}, Model_0^{dec}\right) + \\
&\quad BitErrors_l\left(Model_2^{enc}, Model_1^{dec}\right) + \\
&\quad BitErrors_l\left(Model_2^{enc}, Model_2^{dec}\right)
\end{aligned}
$$

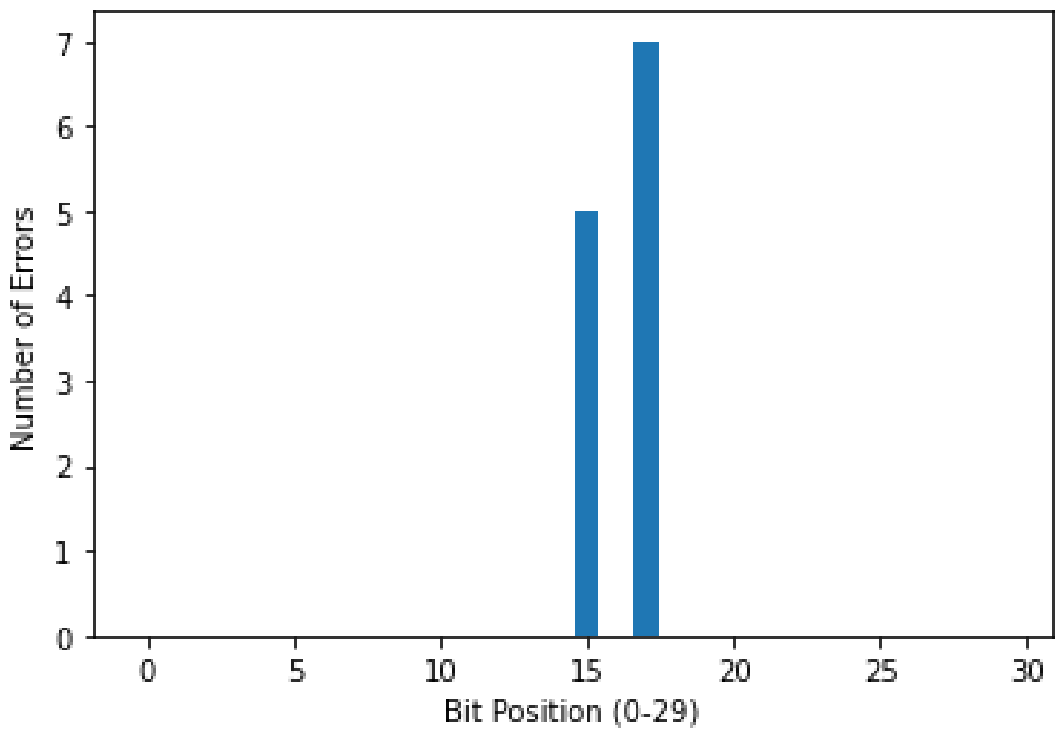


Steganalysis – Bit error distribution actual (2.2)

Actual bit error distribution across all l, showing only corresponding encoders/decoders:

$$NumErrors\ at\ bit\ position\ l =$$

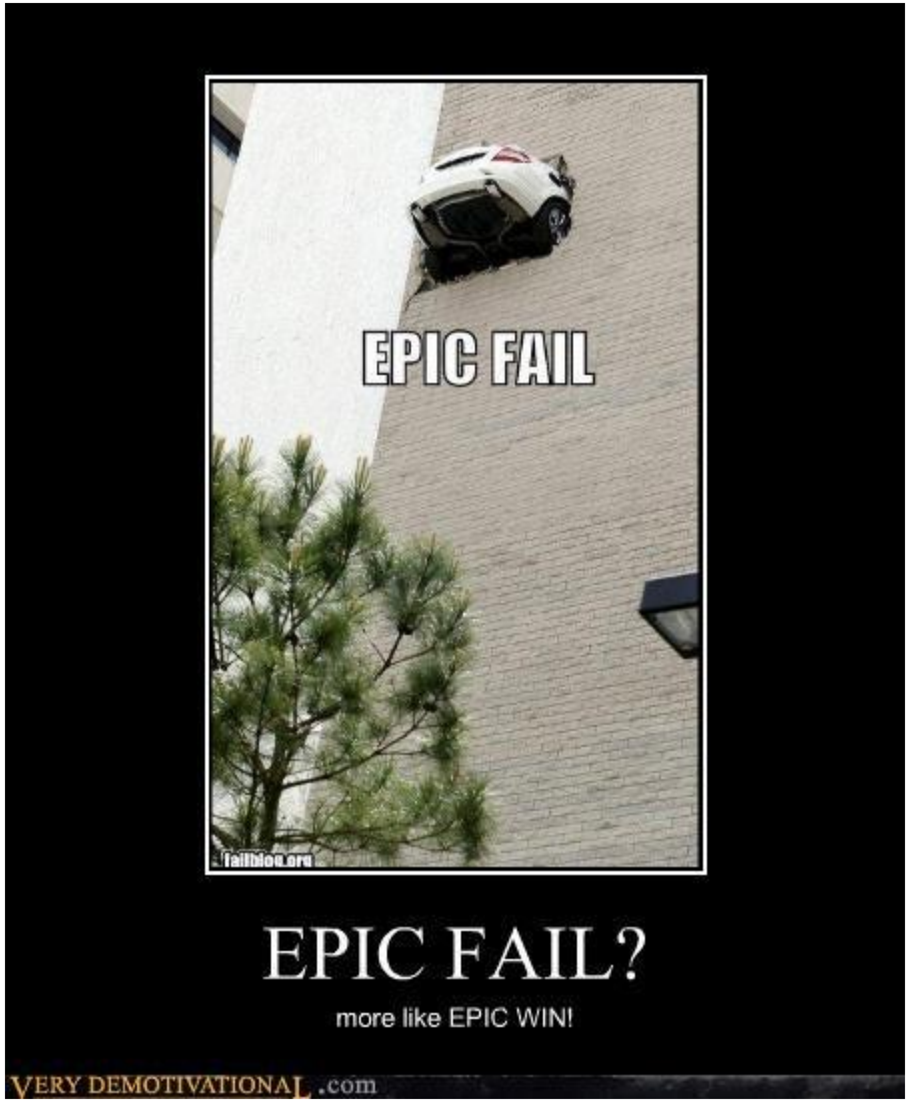
$$BitErrors_l\left(Model_0^{enc}, Model_0^{dec}\right) +$$
$$BitErrors_l\left(Model_1^{enc}, Model_1^{dec}\right) +$$
$$BitErrors_l\left(Model_2^{enc}, Model_2^{dec}\right)$$



Takeaways

- The Randomizer Ratchet was not effective:
  - Ratcheted models were not independent per experiment 1.2
  - Sequentially applying the ratchet resulted in BER decay per experiment 1.3
- An open question if ratcheted steganography with AI is practical.
- Cost function for training AI steganographic models should consider bit position error.

Questions?


EPIC FAIL
failblog.org
EPIC FAIL?
more like EPIC WIN!
VERY DEMOTIVATIONAL .com