UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DAINF - DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

GUSTAVO YUDI BIENTINEZI MATSUZAKE,
PAULO OTAVIO PANICHEK RIBEIRO,
WYVERSON BONASOLI DE OLIVEIRA

ESTEGANÁLISE DE IMAGENS DIGITAIS: METODOLOGIA PARA COMPARAÇÃO DE MÉTODOS TRADICIONAIS E APRENDIZAGEM PROFUNDA

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2017

GUSTAVO YUDI BIENTINEZI MATSUZAKE,
PAULO OTAVIO PANICHEK RIBEIRO,
WYVERSON BONASOLI DE OLIVEIRA

ESTEGANÁLISE DE IMAGENS DIGITAIS: METODOLOGIA PARA COMPARAÇÃO DE MÉTODOS TRADICIONAIS E APRENDIZAGEM PROFUNDA

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Sistemas de Informação da Universidade Tecnológica Federal do Paraná, como requisito parcial para a obtenção do título de Bacharel.

Orientador: Prof. Dr. Rodrigo Minetto
Universidade Tecnológica Federal do Paraná

Orientadora: Prof.ª Dra. Leyza Baldo Dorini
Universidade Tecnológica Federal do Paraná

CURITIBA
2017

Ministério da Educação
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Câmpus Curitiba
Diretoria de Graduação e Educação Profissional
Departamento Acadêmico de Informática
Coordenação do Curso de Bacharelado em Sistemas de Informação

TERMO DE APROVAÇÃO

“ESTEGANÁLISE DE IMAGENS DIGITAIS: METODOLOGIA PARA COMPARAÇÃO DE MÉTODOS TRADICIONAIS E APRENDIZAGEM PROFUNDA”

por

“GUSTAVO YUDI BIENTINEZI MATSUZAKE,

PAULO OTAVIO PANICHEK RIBEIRO e

WYVERSON BONASOLI DE OLIVEIRA”

Este Trabalho de Conclusão de Curso foi apresentado como requisito parcial à obtenção do grau de Bacharel em Sistemas de Informação na Universidade Tecnológica Federal do Paraná - UTFPR - Câmpus Curitiba. O(a)(s) aluno(a)(s) foi(ram) arguido(a)(s) pelos membros da Banca de Avaliação abaixo assinados. Após deliberação a Banca de Avaliação considerou o trabalho _____.

_____ **Profa. Leyza Elmeri Baldo Dorini** (Presidente - UTFPR/Curitiba)	_____ **Prof. Rodrigo Minetto** (Avaliador 1 – UTFPR/Curitiba)
_____ **Prof. André Eugênio Lazzaretti** (Avaliador 2 – UTFPR/Curitiba)	_____ **Prof. Fabio Antonio Dorini** (Avaliador 3 – UTFPR/Curitiba)
_____ **Prof. Ricardo Dutra da Silva** (Avaliador 2 - Instituição)	_____ **Profa. Leyza Elmeri Baldo Dorini** (Professor Responsável pelo TCC – UTFPR/Curitiba)
_____ **Prof. Leonelo Dell Anhol Almeida** (Coordenador(a) do curso de Bacharelado em Sistemas de Informação – UTFPR/Curitiba)	

“A Folha de Aprovação assinada encontra-se na Coordenação do Curso.”

Agradecimentos

Agradecimentos especiais são direcionados à NVIDIA pela doação da placa de vídeo GeForce GTX Titan Xp.

RESUMO

Matsuzake, G. Y. B., Ribeiro, P. O. P, Oliveira, W. B.. Esteganálise de Imagens Digitais: Metodologia para Comparação de Métodos Tradicionais e Aprendizagem Profunda. 2017. 62 f. Trabalho de Conclusão de Curso – Bacharelado em Sistemas de Informação, Universidade Tecnológica Federal do Paraná. Curitiba, 2017.

Tradicionalmente a esteganálise é feita utilizando descritores criados através do conhecimento específico da área. Estudos recentes mostraram a eficácia do uso da aprendizagem de características em esteganálise. Esse trabalho propõe uma metodologia de comparação dessas duas abordagens em diferentes ambientes estegoanalíticos. Com essa metodologia foi possível chegar em discernimentos acerca da natureza das abordagens em relação a ambientações de esteganálise. A abordagem tradicional obteve resultados melhores em ambientes de esteganálise direcionada, por outro lado, a abordagem com aprendizagem profunda (*deep learning*) aparenta ter um poder maior de generalização, assim, tendo resultados melhores em ambientes de esteganálise cega.

Palavras-chave: Esteganografia. Esteganálise. Aprendizagem Profunda.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AUC	***A**rea **U**nder the **C**urve*
BN	***B**atch **N**ormalization*
BOSS	***B**reak **O**ur **S**teganographic **S**ystem*
bpp	***b**its **p**er **p**ixel*
CNN	***C**onvolutional **N**eural **N**etwork*
FLD	***F**isher **L**inear **D**iscriminant*
HPF	***H**igh-**P**ass **F**ilter*
HUGO	***H**ighly **U**ndetectable Ste**go***
LSB	***L**east **S**ignificant **B**it*
MiPOD	***Mi**nimizing the **P**ower of **O**ptimal **D**etector*
OOBe	***O**ut-**O**f-**B**ag **e**rror*
POV	***P**air **o**f **V**alues*
PRNG	***P**seudo-**R**andom **N**umber **G**enerator*
ReLU	***Re**ctified **L**inear **U**nit*
ROC	***R**eceiving **O**perating **C**haracteristic*
SPAM	***S**ubtractive **P**ixel **A**djacency **M**atrix*
SRM	***S**patial **R**ich **M**odels*
STC	***S**yndrome-**T**rellis **C**ode*
S-UNIWARD	***S**patial **Uni**versal **Wa**velet **R**elative **D**istortion*
SVM	***S**upport **V**ector **M**achine*
TanH	**T**angente **H**iperbólica
UNIWARD	***Uni**versal **Wa**velet **R**elative **D**istortion*

Sumário

1 Introdução

Esteganografia pode ser definida como a comunicação de uma mensagem secreta escondida em uma mídia, a qual pode ser qualquer objeto cuja estrutura permita que pequenas alterações sejam realizadas para codificar a mensagem dificultando sua detecção (FRIDRICH, 2009).

Suponha que duas partes, *Alice* e *Bob*, queiram trocar mensagens através do uso de esteganografia, i.e., se comunicar secretamente de tal forma que *Eve* não detecte que tal comunicação está em curso. São reconhecidas cinco etapas fundamentais para uma comunicação secreta utilizando esteganografia (FRIDRICH, 2009):

1. *Alice* e *Bob* escolhem os objetos que irão ocultar a mensagem. Tais objetos devem ser comuns no canal que *Eve* está monitorando, de tal forma que eles não sejam identificados como estranhos e, consequentemente, relacionados facilmente a uma mensagem secreta.
2. *Alice* e *Bob* precisam desenvolver ou escolher implementações de algoritmos para embutir e extrair os dados secretos dos objetos.
3. Opcionalmente, para aumentar a segurança da comunicação, *Alice* e *Bob* podem estipular uma senha previamente, sendo que somente quem a possui consegue extrair a mensagem secreta do objeto.
4. *Alice* deve embutir a mensagem nos objetos escolhidos.
5. *Alice* envia os objetos com a mensagem embutida através do canal de comunicação potencialmente não confiável, para finalmente, *Bob* extrair a mensagem.

Para conseguir detectar que mensagens secretas estão sendo trocadas, *Eve* deverá utilizar técnicas de esteganálise nos objetos aparentemente inofensivos. Esteganálise é o estudo de métodos para a extração de atributos da esteganografia dada uma mídia suspeita. Esses atributos podem ser, por exemplo, se existe ou não uma mensagem escondida, o tamanho da mensagem, possíveis informações sobre a chave, localização da mensagem na mídia, dedução da mensagem, entre outros.

Na década passada, ocorreu um interesse crescente na área de esteganografia e esteganálise devido à disseminação da internet e de mídias digitais, as quais criam um ambiente favorável para a comunicação oculta (dado que são comumente representadas por uma grande quantidade de amostras e ruídos, os quais podem ser modificados imperceptivelmente com o objetivo de codificar uma mensagem secreta) (FRIDRICH, 2009). A Figura 1 ilustra o gradual interesse na área, tomando-se como base artigos publicados anualmente pelo *Institute of Electrical and Electronics Engineers* (IEEE).

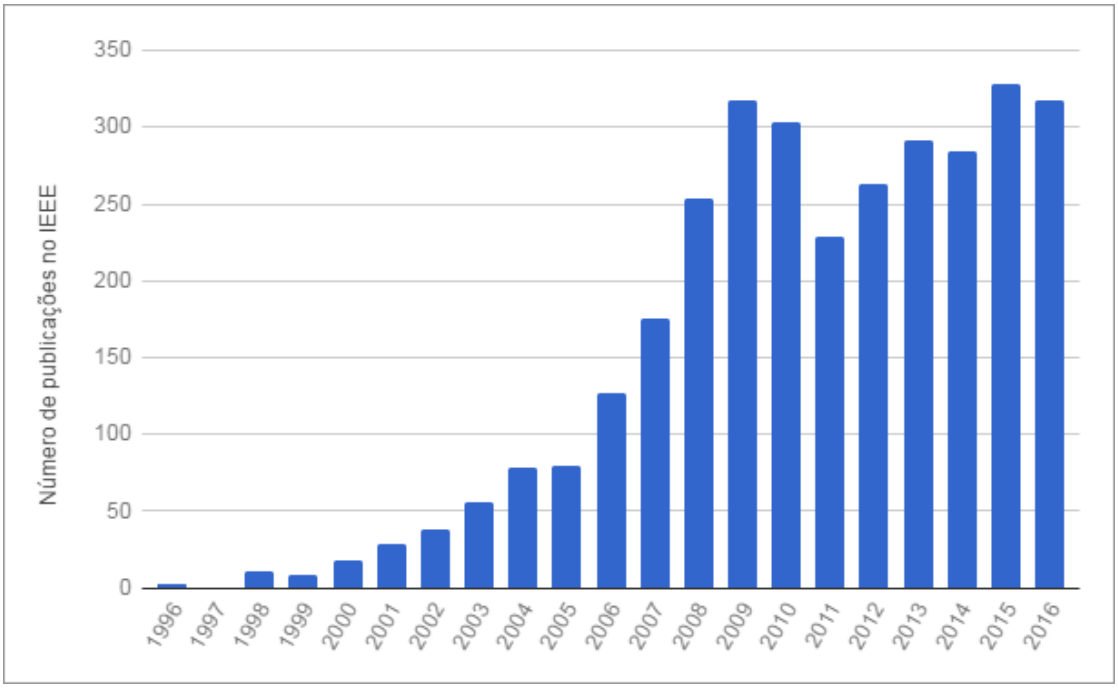


Figura 1 – O crescimento de artigos publicados anualmente pelo IEEE contendo as palavras "steganography" ou "steganalysis" (Autoria própria).

No contexto deste trabalho, serão consideradas imagens como mídias para comunicação. A maior parte dos métodos de esteganálise em imagens possuem uma formulação que baseia-se em um problema de classificação binária (se tem ou não tem a mensagem escondida). É importante ressaltar que, em contraste com a criptoanálise, não é preciso conseguir ler a mensagem escondida para quebrar o sistema esteganográfico - tarefa de uma área chamada Esteganálise Forense (FRIDRICH, 2009).

No caso da classificação binária, os métodos são comumente construídos em dois passos: extração de características e classificação. A primeira parte é geralmente feita com algoritmos já bem estabelecidos, incluindo *Subtractive Pixel Adjacency Matrix* (*SPAM*) (PEVNY; BAS; FRIDRICH, 2010) e o *Steganalysis Rich Models* (SRM) (FRIDRICH; KODOVSKY, 2012). A classificação pode ser feita utilizando, por exemplo, *Support Vector Machines* (SVM).

O projeto eficiente dos descritores de esteganálise tem uma ligação direta com a acurácia da classificação e consiste em uma tarefa não trivial, exigindo um vasto conhecimento nas áreas de esteganografia e esteganálise. Com o objetivo de contornar esses problemas, soluções baseadas em abordagens de aprendizagem profunda foram propostas (TAN; LI, 2014; QIAN et al., 2015; XU; WU; SHI, 2016; XU; WU; SHI, 2016). Neste contexto, as características utilizadas no processo de classificação são definidas (aprendidas) pela própria rede de aprendizagem (WU; ZHONG; LIU, 2017).

Este trabalho propõe uma metodologia para comparação de duas diferentes abordagens de esteganálise em imagens digitais: (a) utilizando o descritor denominado *Spatial Rich Models* (SRM) com um conjunto de classificadores do tipo *Fisher Linear Discriminant* e (b) utilizando aprendizagem profunda, do inglês *Deep Learning*, mais especificamente uma rede neural convolucional (do inglês, Convolutional Neural Network - CNN) inspirada naquela implementada por Xu, Wu e Shi (2016).

Para criar as bases de treinamento e teste, são considerados três algoritmos de esteganografia espacial adaptativos: *Highly Undetectable Stego* (HUGO) (PEVNỲ; FILLER; BAS, 2010), *Minimizing the Power of Optimal Detector* (MiPOD) (SEDIGHI; COGRANNE; FRIDRICH, 2016) e *Spatial Universal Wavelet Relative Distortion* (S-UNIWARD) (HOLUB; FRIDRICH; DENEMARK, 2014). Eles são aplicados nas imagens da base BOSSBase (BAS; FILLER; PEVNỲ, 2011) utilizando diferentes parâmetros, visando criar cenários para avaliação.

Para cada cenário criado, foi gerado uma instância do classificador da abordagem (a) e uma instância do classificador da abordagem (b). Todos os classificadores são testados com todos os cenários gerados. Através dessa metodologia que é análoga a criação de múltiplos cenários estegoanalíticos, foi possível avaliar as abordagens de esteganálise estudadas, tanto ao comparar o desempenho da combinação de diferentes algoritmos de esteganografia nas etapas de treinamento e teste quanto ao avaliar o impacto da utilização de diferentes proporções de mensagens escondidas nas imagens (*payloads*).

O uso dos descritores SRM com um conjunto de classificadores apresentou melhores resultados quando se tem informação prévia do algoritmo utilizado, i.e., quando foi utilizado o mesmo cenário para treinamento e teste. Já para a abordagem baseada em aprendizagem profunda os resultados mostraram uma capacidade de generalização maior, dado que a CNN obteve resultados com pouca diferença significativa para diferentes cenários (apenas a diferença de acurácia foi percebida no treinamento para diferentes algoritmos). Esses resultados sugerem que a CNN, por ter maior poder de generalização, pode obter mais sucesso para o uso em esteganálise cega — onde não há informações sobre o algoritmo de esteganografia utilizado.

Esse trabalho está dividido na seguinte forma: no Capítulo 2 são apresentados conceitos básicos e uma introdução aos métodos de esteganografia, esteganálise e aprendizagem profunda. No Capítulo 3 estão detalhados a base utilizada, a forma como foram realizados os experimentos e as métricas de comparação. O Capítulo 4 mostra os resultados obtidos com os diversos conjuntos de treinamento. Por fim, são apresentadas as considerações finais e algumas sugestões de trabalhos futuros.

1.1 Notação

Será utilizada a anotação da margem com a Figura 2 para localização rápida de definições de termos utilizados no trabalho. Logo após a figura na anotação de margem, estarão os termos definidos no texto. No texto, os termos estarão em negrito.

Figura 2 – Ícone utilizado para a fácil identificação de definições.

Exemplo

Este é um **exemplo** da utilização da anotação de margem.

2 Revisão da Literatura

Para um melhor entendimento do presente trabalho, as Seções 2.1 a 2.4 apresentam os principais conceitos de esteganografia, esteganálise e redes neurais (com foco em abordagens de aprendizagem profunda). Também serão apresentados os conceitos dos algoritmos de esteganálise e esteganografia relevantes para a avaliação dos resultados.

2.1 Esteganografia

Esteganografia

O principal objetivo da **esteganografia** é a transmissão de mensagens sem aparentar que uma comunicação secreta está em curso. Esse resultado pode ser obtido escondendo mensagens em objetos ordinários no canal de comunicação — no meio digital objetos muito disseminados são imagens, áudios, vídeos e textos (FRIDRICH, 2009). Nesta seção serão apresentados conceitos básicos e as principais técnicas de esteganografia em imagens digitais.

2.1.1 Conceitos

Dado embutido, mídia de cobertura e estego objeto

Em esteganografia, o conteúdo a ser escondido denomina-se *embedded data* (**dado embutido**). Essa mensagem será inserida em um objeto de cobertura qualquer, cuja denominação é *cover object* (**mídia de cobertura**). A saída resultante da fusão dos dados embutidos com o objeto de cobertura é chamado de *stego object* (**estego objeto**) (PETITCOLAS; ANDERSON; KUHN, 1999). O tipo da mídia de cobertura pode ser explicitado, utilizando-se, no caso de imagens digitais, os termos *cover image* (**imagem de cobertura**) e *stego image* (**estego imagem**).

Chave esteganográfica e sistema esteganográfico

A chave, quando utilizada para esconder a mensagem, é denominada como *stego key* (**chave esteganográfica**). Podemos definir o conjunto desses artefatos (dado embutido, mídia de cobertura, estego objeto e chave esteganográfica) como *stegosystem* (**sistema esteganográfico** ou canal esteganográfico), cujo esquema está ilustrado na Figura 3.

Quando a mídia de cobertura é uma imagem digital, a esteganografia pode atuar no **domínio espacial** ou no **domínio da frequência** (FRIDRICH, 2009), sendo o primeiro o foco deste trabalho.

Domínio espacial

No **domínio espacial** a imagem é representada por uma grade retangular densa de pixels, gerando arquivos grandes e que permitem mensagens maiores em-

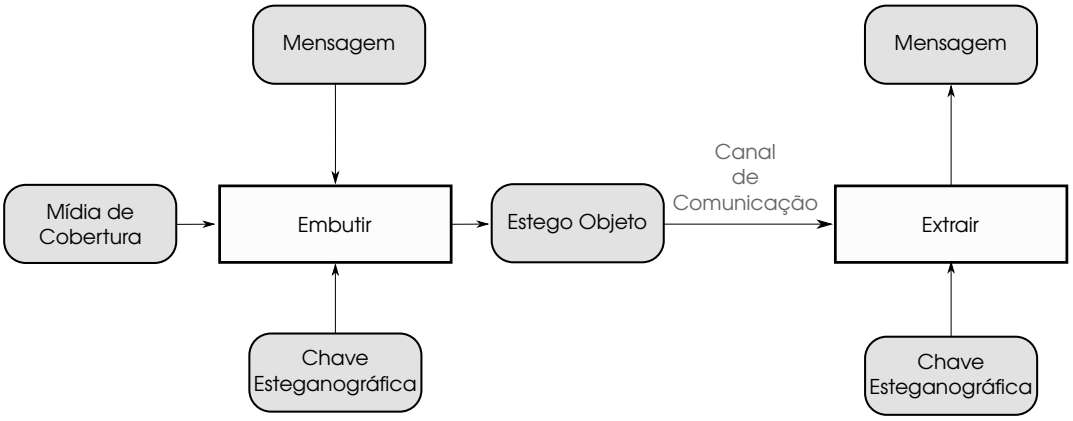


Figura 3 – Elementos de um sistema esteganográfico (Autoria própria).

butidas (FRIDRICH, 2009). No armazenamento utilizando mapa de bits, o qual será considerado nesse trabalho, a imagem é armazenada linha por linha, cada uma composta por pixels representados de acordo com o alcance de cores (BROWN, 1995). Por exemplo, em imagens coloridas cada pixel é descrito por três bytes, em imagens em tons de cinza por um byte e em imagens binárias apenas um bit representa cada pixel.

2.1.2 Métodos de Esteganografia

Não-adaptativos e adaptativos

Há diferentes formas de escolher quais pixels da imagem de cobertura devem ser utilizados para esconder a mensagem. Esses métodos são divididos em **não-adaptativos** e **adaptativos**. Os métodos não-adaptativos não levam em consideração o conteúdo da imagem para escolher os pixels onde a mensagem será embutida, se baseiam na hipótese que os ruídos da imagem são independentes de pixel para pixel. Algoritmos **adaptativos** são os algoritmos que levam em consideração o conteúdo da imagem de cobertura para esconder a mensagem.

Sequencial e pseudo-aleatório

Os principais métodos não-adaptativos para escolher os pixels da imagem são o **sequencial** e o **pseudo-aleatório**. O primeiro simplesmente esconde a mensagem sequencialmente na imagem, sendo que o primeiro pixel pode ser escolhido com base em um deslocamento arbitrário. O modo pseudo-aleatório utiliza a sequência de pixels definida por um gerador de números pseudo-aleatórios (PRNG, *Pseudo-Random Number Generator*). A implementação desse método geralmente é feita derivando uma chave para um número inteiro que é utilizado como semente do PRNG. Por fim, no formato adaptativo, a escolha dos pixels depende do conteúdo da imagem de cobertura e é

realizada por meio de heurísticas, as quais buscam minimizar a detecção.

Substituição e combinação

Após a seleção dos pixels, a inserção da mensagem na imagem é realizada por métodos que podem ser classificados como *replacement* (**substituição**) ou *matching* (**combinação**) (KER, 2014). Nos algoritmos por substituição, o bit menos significativo de cada um dos pixels selecionados é substituído pelo bit correspondente da mensagem. Algoritmos por combinação verificam se o bit da mensagem é o mesmo que o bit menos significativo do pixel. Se for igual, não é realizada nenhuma modificação. Caso contrário, aleatoriamente soma ou subtrai 1 ao valor do pixel. Existem também alguns métodos diferentes de combinação propostos na literatura como, por exemplo, utilizar uma função que correlaciona dois pixels com o objetivo de realizar menores distorções na imagem (MIELIKAINEN, 2006).

Modelo de imagem e codificador

Sistemas adaptativos começaram a chamar a atenção através do princípio *minimal impact embedding*. *Minimal impact embedding* são sistemas esteganográficos que separam seu design em **modelo de imagem** (*image model*) e **codificador** (*coder*) — no que se refere ao modelo (e suas heurísticas) utilizado para escolher os pixels e a maneira que ele irá codificar a mensagens neles, respectivamente (FRIDRICH; FILLER, 2007). Os modos sequencial e pseudo-aleatório são modelos de imagem não-adaptativos e os métodos substituição e combinação são codificadores não-adaptativos. Os modelos de imagem adaptativos tentam modelar os custos de modificação de cada pixel da imagem e os codificadores tentam embutir a informação na imagem minimizando a função de distorção do modelo de imagem.

Payload

O ***payload*** α determina a razão entre mensagem embutida na imagem e pixels com custo menor que infinito de acordo com o modelo de imagem, ou seja, seja m o número de bits de uma mensagem e k o número de elementos com o custo de distorção $\delta < \infty$ no modelo da imagem, então $\alpha = m/k$ (FILLER; JUDAS; FRIDRICH, 2010). Em relação aos algoritmos não-adaptativos, todos os custos de distorção são menores que infinito, então, o *payload* pode ser interpretado como a proporção da imagem $0 \leq p \leq 1$ que será utilizada para esconder a mensagem, sendo medido em bits por pixel (bpp). Neste trabalho todos os algoritmos utilizam o parâmetro *payload*.

Um sistema esteganográfico é dito *perfeitamente seguro* se a distribuição de probabilidade da mídia de cobertura corresponde exatamente àquela do estego-objeto (CACHIN, 2004). Esse problema é de difícil resolução com mídias digitais reais (naturais) porque exige o conhecimento exato de tal distribuição da mídia de cobertura. Embora métodos denominados *geração de mídia de cobertura* (ANDERSON, 1996; ADAMS, 2008; RYABKO; RYABKO, 2009) proponham uma solução para o problema, eles acabam gerando objetos tendenciosos e com muitos artefatos, e que podem ser

facilmente detectados. Na prática esse problema não é resolvido na sua forma ótima, sendo que as soluções mais utilizadas buscam minimizar as perturbações da distribuição de probabilidade, na esperança de que sejam encobertas pelos ruídos naturais da mídia de cobertura.

2.1.2.1 LSB e LSB *Matching*

LSB

Os algoritmos baseados na abordagem ***L**east **S**ignificant **B**its* (**LSB**) substituem os bits menos significativos de diversos pixels da imagem para embutir a mensagem, i.e., o LSB utiliza o codificador de substituição. A implementação mais trivial é utilizando o modelo de imagem sequencial não-adaptativo, mas o modelo pseudo-aleatório com uma chave esteganográfica também é utilizado.

Seja uma imagem em escala de cinza I formada por n pixels $p_1, p_2, \ldots, p_n$, tal que cada pixel $p_i = (b_{i,1}, b_{i,2}, \ldots, b_{i,8})$ é representado por um byte, em que $b_{i,j}$ é um bit mais significativo do que $b_{i,j+1}$. Seja também uma mensagem, $M = \{m_1, m_2, \ldots, m_k\}$, de tamanho $k \leq n$. A função de embutir do LSB sequencial pode ser definida como:

$$LSB(I, M, l) \implies p_l = (b_{l,1}, b_{l,2}, \ldots, b_{l,7}, m_l), \tag{1}$$

em que l é a posição do bit da mensagem que está sendo embutido. Exemplos do LSB são ilustrados na Figura 4.

A abordagem LSB é muito empregada devido a sua simplicidade de implementação. Apesar disso, por atuar nos bits menos significativos, a modificação realizada é visualmente imperceptível (LI et al., 2011).

Desde 2004, o método de LSB por substituição se mostrou fraco a ataques estatísticos (KER, 2014) e resultados posteriores melhoraram sua detecção em algoritmos não-adaptativos (PEVNY; BAS; FRIDRICH, 2010).

LSB *Matching*

O **LSB *Matching***, como o próprio nome diz, utiliza o processo de *matching* para esconder a mensagem, i.e., seu codificador é por combinação. Para ampliar sua segurança geralmente é utilizado o modelo de imagem pseudo-aleatório não-adaptativo para escolha da sequência de pixels a ser utilizada. Entretanto, também pode ser implementado utilizando a abordagem sequencial não-adaptativo.

Das várias abordagens existentes na literatura, a mais utilizada é a proposta por Mielikainen (2006), conforme detalhada no Algoritmo 1. Ela permite que a mesma quantidade de dados seja escondida com menos modificações na imagem de cobertura.

Seja $M = \{m_1, m_2, \ldots, m_k\}$ uma mensagem de tamanho k a ser escondida e $Y = \{y_0, y_1, \ldots, y_n\}$ a estego imagem. Para todo m_i em que i é par, $m_i = lsb(y_i)$,

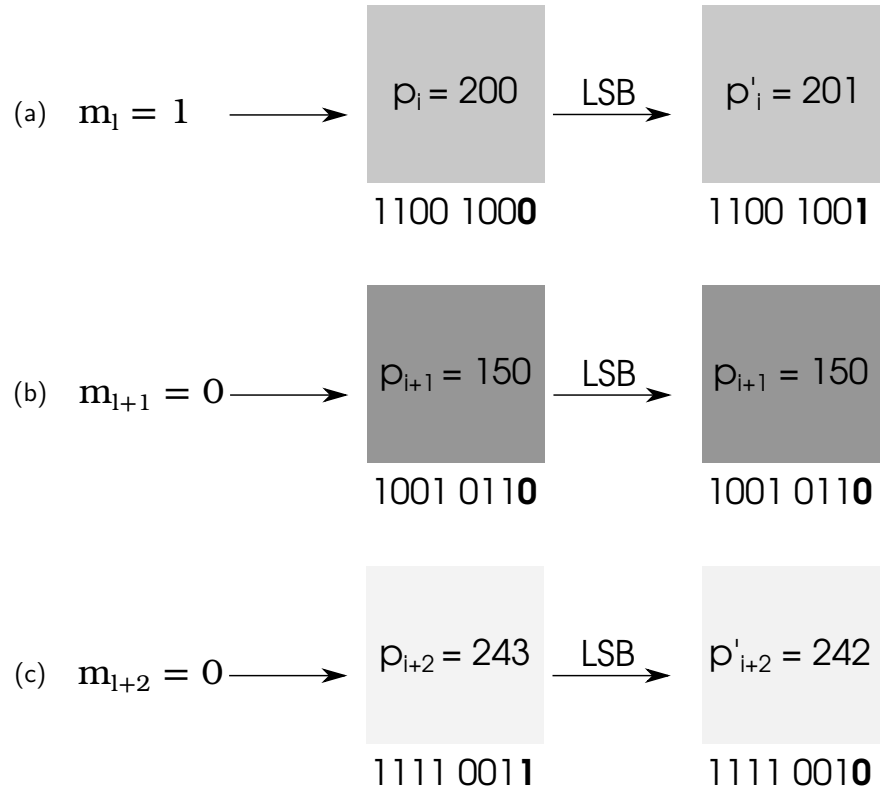


Figura 4 – Codificação da mensagem $m = (1, 0, 0)$ nos pixels $p_i = 200$, $p_{i+1} = 150$ e $p_{i+2} = 243$. Em negrito está destacado o bit menos significativo de cada pixel. (a) Representação da codificação do bit 1 no pixel p_i com valor 200, resultando no pixel $p'_{i+1} = 201$; (b) codificação do bit 0 da mensagem no pixel $p_{i+1} = 150$ não resultando em modificação; (c) codificação do bit 0 no pixel $p_{i+2} = 243$ resultando no pixel $p'_{i+2} = 242$ (Autoria própria).

onde $lsb(x)$ retorna o bit menos significativo de x. Para todo m_i onde i é impar $m_i = f(y_{i-1}, y_i)$, em que $f(a, b)$ é uma função binária de correlação de a e b.

Considere as seguintes propriedades:

Propriedade 1. $f(l-1, n) \neq f(l+1, n), \quad \forall_{l,n} \in \mathbb{Z}$

Propriedade 2. $f(l, n) \neq f(l, n+1), \quad \forall_{l,n} \in \mathbb{Z}$

Caso elas sejam atendidas, é possível que o processo esteganográfico carregue uma informação a mais para o próximo pixel, resultando em menos modificações na imagem de cobertura. Um exemplo que atende estas propriedades é $f(a, b) = LSB(\lfloor a/2 \rfloor + b)$.

Algoritmo 1: LSB Matching

Input: Um par de pixels x_i, x_j da imagem de cobertura e dois bits m_i, m_j da mensagem
Output: Um par de pixels y_i, y_j da estego imagem

if $m_i == lsb(x_i)$ **then**
  if $m_j == f(m_i, m_j)$ **then**
    $y_j \leftarrow x_j$
  else
    /* Soma ou subtrai x_j */
    $y_j \leftarrow \pm x_j$
  end
  $y_i \leftarrow x_i$
else
  if $m_j == f(x_i - 1, x_j)$ **then**
    $y_i \leftarrow x_i - 1$
  else
    $y_i \leftarrow x_i + 1$
  end
  $y_j \leftarrow x_j$
end

2.1.2.2 HUGO

HUGO

H**ighly **U**ndetectable Ste**go (**HUGO**) é um algoritmo para gerar um modelo de imagem adaptativo *ad-hoc* para minimizar o impacto de detecção do descritor *SPAM* (será visto com mais detalhes na Subseção 2.2.2.2). Em implementações práticas, o HUGO é utilizado como modelo de imagem e o *Syndrome-Trellis Code (STC)* (FILLER; JUDAS; FRIDRICH, 2010) como codificador. O algoritmo foi apresentado através de um novo modelo para definir custos de embutir através de descritores para esteganálise (PEVNỲ; FILLER; BAS, 2010) (descritores para esteganálise podem ser vistos na Subseção 2.2.2). Nesse modelo, os custos do HUGO são calculados a partir das matrizes de correlação do SPAM. A função de custo de embutir do HUGO é representada pela Equação 2:

$$D(X,Y) = \sum_{d_1,d_2,d_3=-T}^{T} \left(w(d_1,d_2,d_3) \left| \sum_{k\in\{\rightarrow,\leftarrow,\uparrow,\downarrow\}} C^{X,k}_{d_1,d_2,d_3} - C^{Y,k}_{d_1,d_2,d_3} \right| + w(d_1,d_2,d_3) \left| \sum_{k\in\{\searrow,\nwarrow,\swarrow,\nearrow\}} C^{X,k}_{d_1,d_2,d_3} - C^{Y,k}_{d_1,d_2,d_3} \right| \right), \tag{2}$$

em que:

- T é um limiar entre 0 e 255;
- X é a imagem de cobertura;
- Y é a estego imagem;
- $C^{I,k}_{d_1,d_2,d_3}$ é a matriz de correlação da imagem I na direção k;
- $w(d_1, d_2, d_3)$ é uma função de peso que tem a forma da Equação 3.

$$w(d_1, d_2, d_3) = \frac{1}{\left(\sqrt{d_1^2 + d_2^2 + d_3^2} + \sigma\right)^{\gamma}}, \tag{3}$$

onde $\sigma, \gamma > 0$, são parâmetros que podem ser ajustados para minimizar a detecção.

O Algoritmo 2 apresenta o pseudo-código do HUGO.

2.1.2.3 S-UNIWARD

Historicamente, após bons resultados da função de distorção denominada *Wavelet Obtained Weights* (WOW) (HOLUB; FRIDRICH, 2012), melhorias e generalizações no método deram origem uma função similar, o *Uniward* (*Universal Wavelet Relative Distortion*) (HOLUB; FRIDRICH; DENEMARK, 2014). Ela representa a distorção na forma de uma soma de mudanças relativas entre a imagem de cobertura e a estego imagem representadas no domínio *wavelet*. Este trabalho abordará apenas o ***S**patial **Uni**versal **Wa**velet **R**elative **D**istortion* (***S-UNIWARD***), que trata da versão no domínio espacial da função *Uniward*.

S-UNIWARD

Em resumo, o S-UNIWARD é um algoritmo que gera um modelo de imagem adaptativo através de uma função de distorção para o domínio espacial, a qual calcula as mudanças relativas da estego imagem e da imagem de cobertura no domínio wavelet.

Na prática, após a utilização da função de distorção *Uniward*, o esteganógrafo terá que utilizar uma função de codificação para minimizar a função de distorção. No domínio espacial geralmente é utilizado o STC (assim como no HUGO).

No *S-UNIWARD*, a distorção depende da escolha de um banco de filtros direcionais e um parâmetro escalar para estabilizar os cálculos numéricos. Banco de filtros direcionais consiste em um conjunto de três filtros lineares, $\beta = \{K^{(1)}, K^{(2)}, K^{(3)}\}$, invariantes a deslocamento (HOLUB; FRIDRICH; DENEMARK, 2014). Esses kernels são utilizados basicamente para avaliar a suavidade da imagem nas direções horizontal, vertical e diagonal computando os resíduos direcionais $W^{(k)} = K^{(k)} \star X$, em que W são os resíduos, X é a imagem de cobertura e $\star$ é uma convolução *mirror-padded* (HOLUB; FRIDRICH, 2012). Embora a direção dos filtros $K^{(1)}, K^{(2)}, K^{(3)}$ podem ser escolhidos de forma arbitrária, Holub, Fridrich e Denemark (2014) utilizaram kernels construídos a partir de filtros de decomposições *wavelet* 1-dimensional passa-baixa H e passa-alta G

Algoritmo 2: HUGO (PEVNÝ; FILLER; BAS, 2010)

Input: Todos os PIXELS da imagem de cobertura X
Output: A estego imagem Y

```
for (i, j) in PIXELS do
    // Calcula o impacto da codificação de +1
    stego_plus ← X.copy()
    stego_plus[i][j] + +
    embed_impact_plus[i][j] ← D(X, stego_plus)
    // Calcula o impacto da codificação de -1
    stego_minus ← X.copy()
    stego_minus[i][j] − −
    embed_impact_minus[i][j] ← D(X, stego_minus)
end
// Mínimo, elemento a elemento
embed_impact_min ← min(embed_impact_plus, embed_impact_minus)
// Syndrome Trellis-Code
pixels_to_change ←
  minimize_embed_impact(LSB(X), embed_impact_min, message)
Y ← X
if model_correction_step_enabled then
    for (i, j) in pixels_to_change do
        correction_plus ← Y
        correction_plus[i][j] + +
        dp ← D(X, correction_plus)
        correction_minus ← Y
        correction_minus[i][j] − −
        dm ← D(X, correction_minus)
        if dp < dm then
            Y[i][j] + +
        else
            Y[i][j] − −
        end
    end
else
    for (i, j) in pixels_to_change do
        if embed_impact_plus[i][j] < embed_impact_minus[i][j]
          then
            Y[i][j] + +
        else
            Y[i][j] − −
        end
    end
end
```

da Equação 4.

$$K^{(1)} = H \cdot G^T, K^{(2)} = G \cdot H^T, K^{(3)} = G \cdot G^T \tag{4}$$

Seja X e Y um par de imagens de dimensão $n \times m$, e $W_{i,j}^{(k)}(X)$ e $W_{i,j}^{(k)}(Y)$ os i,j-ésimos coeficientes *wavelet* correspondente as imagens X e Y. A função de distorção *Uniward* é definida pela Equação 5.

$$D(X,Y) \triangleq \sum_{k=1}^{3} \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{|W_{i,j}^{(k)}(X) - W_{i,j}^{(k)}(Y)|}{\sigma + |W_{i,j}^{(k)}(X)|}, \tag{5}$$

sendo $\sigma > 0$ a constante de estabilização dos cálculos numéricos.

2.1.2.4 MiPOD

Historicamente, o design de sistemas esteganográficos para imagens digitais era muito dependente de heurísticas *ad-hoc*. Essa abordagem deu origem a diversos métodos esteganográficos (PEVNỲ; FILLER; BAS, 2010; HOLUB; FRIDRICH, 2012; HOLUB; FRIDRICH; DENEMARK, 2014; LI et al., 2014) que são realizados definindo uma função de custo para modificar cada pixel para, posteriormente, a mensagem ser escondida minimizando esse custo. Porém, todo esse paradigma é questionado, uma vez que não existe nenhuma conexão formal entre essas funções de distorção e a detecção estatística. É argumentado por Böhme (2010) que essa conexão pode nunca ser encontrada.

MiPOD

O **MiPOD**, acrônimo para ***Mi**nimizing the **P**ower of **O**ptimal **D**etector*, foi desenvolvido modelando a imagem de cobertura como um vetor N-dimensional, onde $X = \{x_1, x_2, \ldots, x_n\}$ e cada x_i é constituído através de uma distribuição gaussiana independente $x_i \sim \mathcal{N}(\mu_i, \omega_i^2)$ e quantizados em pontos discretos $k \in \mathbb{Z}$. Aqui, μ_i denota o conteúdo da imagem de cobertura sem ruído de captação, ω_i^2 é a variância do ruído da captura. Nesse modelo, $z_i = x_i - \mu_i$ também é gerado por uma gaussiana independente $z_i \sim \mathcal{N}(0, \sigma_i^2)$.

Dada a imagem de cobertura $X = \{x_1, x_2, \ldots, x_n\}$ a estego imagem $Y = \{y_1, y_2, \ldots, y_n\}$ é obtida aplicando independentemente as seguintes regras de probabilidade:

$$\begin{aligned}
\mathbb{P}(y_i = x_i + 1) &= \beta_i, \\
\mathbb{P}(y_i = x_i - 1) &= \beta_i, \\
\mathbb{P}(y_i = x_i) &= 1 - 2\beta_i,
\end{aligned}$$

tal que $0 \leq \beta \leq 1/3$.

Com essas suposições sobre a imagem de cobertura e a estego imagem, é possível chegar em um detector ótimo que tem conhecimento de todas os desvios originais da imagem de cobertura $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$. Tal detector é representado pela Equação 6:

$$\Phi = \sqrt{2 \sum_{i=1}^{N} \sigma_i^{-4} \beta_i^2}, \tag{6}$$

que é o coeficiente de deflexão das duas hipóteses do detector (se é ou não uma estego imagem).

Para o estado da arte ser obtido, é preciso estimar a variância da imagem de cobertura para minimizar o poder de detecção da Equação 6 (SEDIGHI; COGRANNE; FRIDRICH, 2016). O papel do esteganógrafo é, nesse caso, estimar a variância da imagem de cobertura; calcular a detectabilidade utilizando a Equação 6; criar o modelo de imagem adaptativo e os custos de codificar a mensagem. Uma ilustração desse modelo pode ser visto na Figura 5.

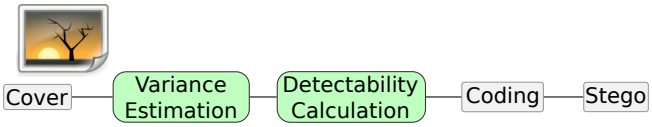
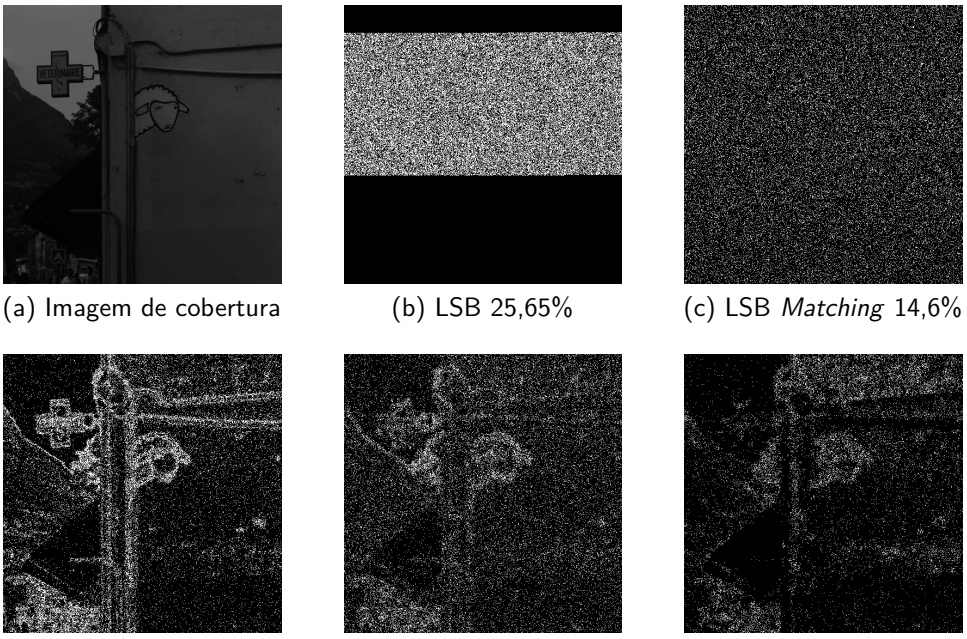


Figura 5 – Etapas do MiPOD (SEDIGHI; COGRANNE; FRIDRICH, 2016).

2.1.3 Comparação dos Métodos de Esteganografia

Como discutido nas subseções anteriores, os algoritmos de esteganografia LSB e LSB *Matching* são chamados de não-adaptativos pela aceitação da hipótese que os custos de embutir o *payload* na imagem de cobertura são os mesmos independente do conteúdo da imagem. Por outro lado, os algoritmos HUGO, S-UNIWARD e MiPOD utilizam o princípio do *Minimal Impact Embedding*, que separa o processo de esteganografia em modelo de imagem e codificador — sendo o modelo de imagem os custos de se modificar cada pixel de acordo com o conteúdo da imagem de cobertura.

A Figura 6 ilustra o impacto dessa diferença de abordagem para a criação do modelo de imagem de cada algoritmo. Para uma mesma mensagem, é possível observar que os algoritmos adaptativos HUGO, S-UNIWARD e MiPOD modificam, respectivamente, 14,76%, 12,69% e 6,9% da imagem, exemplificando o ganho de eficiência — relacionado ao número de modificações — entre os algoritmos adaptativos. Para os não adaptativos, a proporção da imagem alterada é consideravelmente maior.



(a) Imagem de cobertura (b) LSB 25,65% (c) LSB *Matching* 14,6%

(d) HUGO 14,75% (e) S-UNIWARD 12,68% (f) MiPOD 6,9%

Figura 6 – Comparação das modificações feitas por diferentes algoritmos de esteganografia. (a) é a amostra 5008 da base BOSSBase (BAS; FILLER; PEVNỲ, 2011). As modificações estão representadas por imagens binárias, em branco as modificações realizadas para embutir um *payload* de 0.6 bpp, em preto são os pixels não modificados de (a). Ao lado do nome de cada algoritmo estão as porcentagens de modificação da imagem. (Autoria própria).

Além disso, algumas características dos algoritmos podem ser observadas. Ao passo que o LSB inclui a mensagem de forma não-controlada, os algoritmos adaptativos buscam por regiões (tais como bordas) em que a alteração é menos perceptível.

2.2 Esteganálise

O uso excessivo de algoritmos de esteganografia atrai estegoanalistas com o intuito de elaborar ataques para constatar a presença ou ausência de dados escondidos em imagens. Nesta seção, os métodos para atacar os algoritmos serão explicados mais detalhadamente.

2.2.1 Conceitos

Esteganálise

Esteganálise nada mais é do que o processo de detectar a esteganografia, ou seja, enquanto a esteganografia busca ocultar a existência informações, a esteganálise visa atacar essa forma de comunicação. Um estegoanalista tem sucesso em seu

ataque caso consiga distinguir uma estego imagem de uma imagem de cobertura com probabilidade maior que o limiar de 50% de acurácia, equivalente a um simples chute aleatório. Um ataque também pode ser planejado de modo a aproximar a quantidade de informação supostamente escondida na estego imagem (FRIDRICH, 2009).

Esteganálise cega e esteganálise direcionada

A esteganálise pode ser subdividida em duas classes principais: *blind steganalysis* (**esteganálise cega**) e *targeted steganalysis* (**esteganálise direcionada**). A primeira classe compreende os algoritmos de esteganálise que não levam em consideração o conhecimento prévio de um método de esteganografia específico, e a segunda consiste em planejar um ataque em um domínio de algoritmos pré-definido (FRIDRICH, 2009). Dado que o enfoque do trabalho é no domínio espacial, a seção terá foco na descrição dos algoritmos de esteganálise, cega e direcionada, no domínio espacial.

2.2.2 Métodos de Esteganálise

Ataque visual

A maneira mais simples de ataque esteganográfico é por meio aural, também chamado de **visual**. Esse ataque consiste em deslocar os bits menos significativos de cada pixel da imagem para as primeiras posições e preencher os demais com o mesmo valor. Dessa forma, os bits que menos influenciavam na imagem passam a ser os únicos relevantes. Por conseguinte, pixels que foram manipulados passam a divergir dos demais tornando visíveis rastros de informações escondidas (WAYNER, 2009).

Um exemplo pode ser visto na Figura 7, onde a aplicação do ataque à imagem com uma mensagem oculta gerou, na parte superior, um comportamento discrepante em relação ao restante da imagem. Tal região corresponde aos pixels alterados pelo LSB.

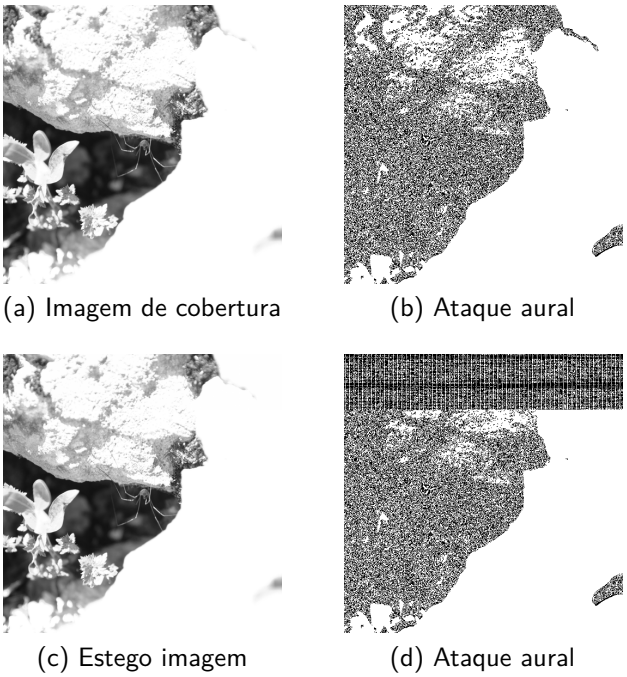
Ataques estatísticos

Verificar visualmente alterações na imagem tipicamente não é suficiente quando um método robusto de esteganografia é utilizado. Pode-se então aplicar **ataques estatísticos**, os quais analisam a distribuição dos pixels e seus bits menos significativos em uma imagem. Dessa forma, a partir de padrões frequentes, busca-se calcular a probabilidade de haver uma mensagem oculta na mesma (FRIDRICH; GOLJAN, 2002).

Esse tipo de ataque é muito empregado por depender somente da disposição dos bits na imagem. Nas seções subsequentes serão explicados os algoritmos estatísticos relevantes na área de esteganálise, bem como os que compõe o estado da arte atual.

2.2.2.1 Esteganálise sem Uso de Descritores

O primeiro e mais simples método de esteganálise estatística, proposto por Westfeld e Pfitzmann (1999), é baseado no conceito de **par de valores** (POV, *pair*



(a) Imagem de cobertura (b) Ataque aural

(c) Estego imagem (d) Ataque aural

Figura 7 – Detecção do LSB por ataque visual. (a) Imagem 269 do BOSSBase (BAS; FILLER; PEVNỲ, 2011). (b) Ataque visual à imagem (a). (c) Estego imagem com LSB. (d) Ataque visual à estego imagem (Autoria própria).

of values), que são valores que diferem apenas no bit menos significativo. O ataque baseia-se na suposição de que, quando dados são escondidos em uma imagem, existe uma tendência à equalização dos valores de cada POV no histograma.

Devido às limitações da análise dos POVs, foi proposto por Fridrich e Goljan (2002) um método de esteganálise mais robusto, denominado *RS-Attack*. A sua ideia principal é avaliar a correlação espacial entre os pixels da estego imagem a partir de uma função discriminante.

Contudo, o *RS-Attack* tem um bom desempenho somente quando a esteganografia é feita com algoritmos muito simples, como o LSB. Com algoritmos como o LSB *Matching* o desempenho é mediano.

Ambos os algoritmos são de implementação simples, mas apresentam complicações principalmente em imagens com grande quantidade de ruído, nas quais a quantidade de informação pressuposta pelo algoritmo para uma imagem de cobertura sem nenhuma informação é extremamente alta. Esse fenômeno é denominado ***initial bias*** e é tratado de maneira muito mais concisa quando se extrai características relevantes das imagens previamente.

Initial bias

2.2.2.2 Descritor SPAM

Com o intuito de quebrar eficientemente o LSB *Matching*, uma análise empírica no banco de imagens BOWS2[1] revelou que, em uma imagem natural, a probabilidade de pixels adjacentes terem valores parecidos é elevada. A representatividade dessa probabilidade pode ser vista na Figura 8, na qual a intensidade da escala de cinza é inversamente proporcional a probabilidade $Pr(I_{i,j} = x \wedge I_{i,j+1} = y)$ (PEVNY; BAS; FRIDRICH, 2010).

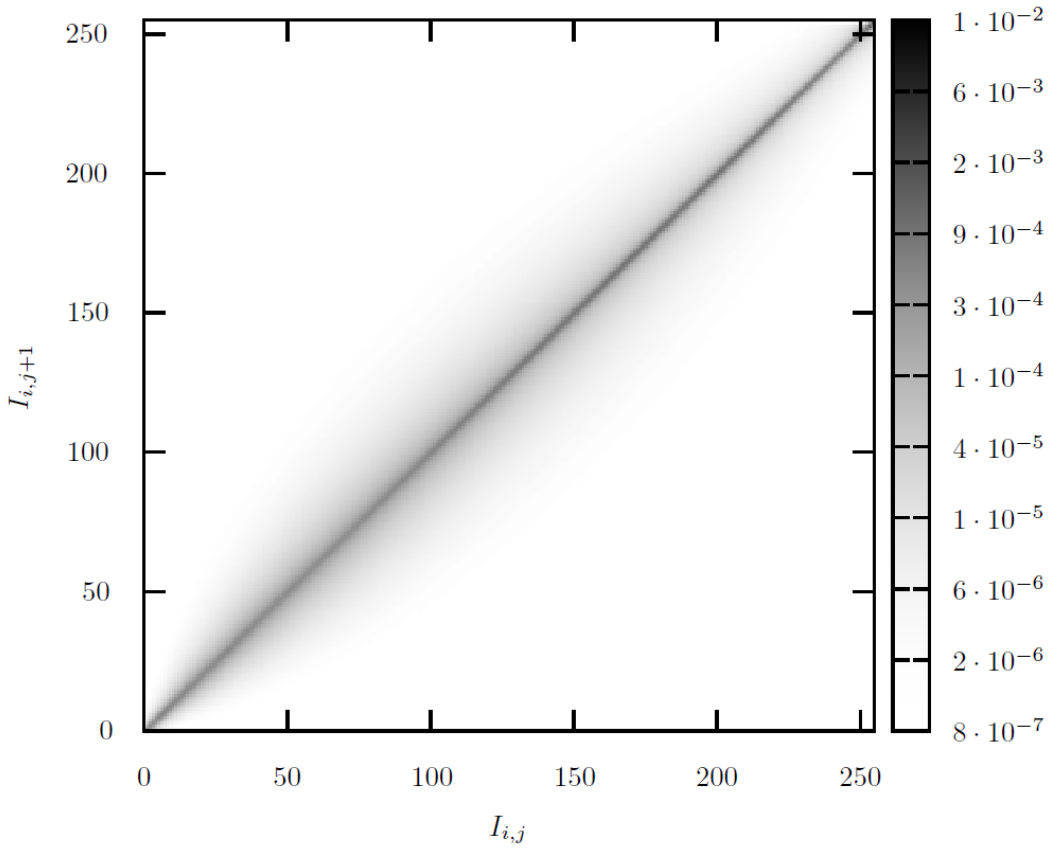


Figura 8 – Distribuição da probabilidade do valor de dois pixels horizontalmente adjacentes possuírem os valores descritos no eixo x e y, em escala logarítmica (PEVNY; BAS; FRIDRICH, 2010).

Essa ideia abre a possibilidade de descrever a imagem a partir de uma matriz de probabilidades e selecionar estego imagens via um algoritmo de classificação. O primeiro método que utiliza uma matriz de probabilidades para a descrição de imagens é denominado ***S**ubtractive **P**ixel **A**djacency **M**atrix* (**SPAM**) e se baseia nos gradientes direcionais da imagem de cobertura. A derivada direcional para a direita do pixel $I_{i,j}$,

SPAM

[1] http://bows2.gipsa-la.inpg.fr

por exemplo, é dada pela Equação 7 e pode ser vista como um filtro linear espacial $[-1,1]$.

$$D^{\rightarrow}_{i,j} = I_{i,j} - I_{i,j+1} \tag{7}$$

O primeiro passo do algoritmo para gerar o descritor é computar as derivadas em todas as direções e sentidos possíveis: $\{\leftarrow, \rightarrow, \uparrow, \downarrow, \nearrow, \swarrow, \nwarrow, \searrow\}$, resultando em um vetor $\mathbf{D}_{i,j}$ para cada gradiente possível. Para diminuir a dimensionalidade do descritor, o escopo é limitado a considerar somente os valores de $\mathbf{D}_{i,j} \in [-T, T]$, onde T é um parâmetro passado previamente ao SPAM.

A matriz resultante é gerada a partir de uma cadeia de Markov (LIPSCHUTZ; SCHILLER, 1998), podendo ser de primeira ou segunda ordem. Nas Equações 8 e 9 pode-se observar como cada posição (u, v) e (u, v, w) da matriz para a derivada à direita é gerada para ambas as ordens, respectivamente.

$$\mathbf{M}^{\rightarrow}_{u,v} = Pr(\mathbf{D}^{\rightarrow}_{i,j+1} = u | \mathbf{D}^{\rightarrow}_{i,j} = v) \tag{8}$$

$$\mathbf{M}^{\rightarrow}_{u,v,w} = Pr(\mathbf{D}^{\rightarrow}_{i,j+2} = u | \mathbf{D}^{\rightarrow}_{i,j+1} = v | \mathbf{D}^{\rightarrow}_{i,j} = w) \tag{9}$$

O descritor SPAM é a matriz de probabilidades resultante da cadeia de Markov. Um esquema que ilustra o processo do SPAM pode ser visto na Figura 9.

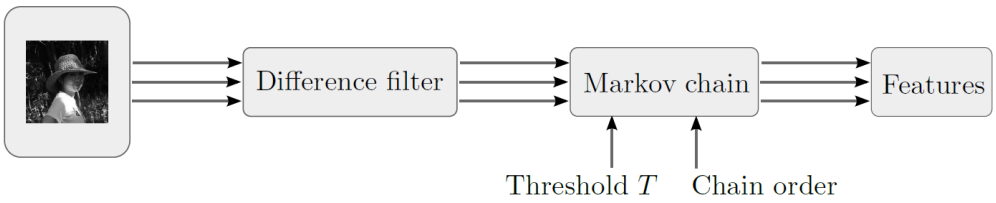


Figura 9 – Esquema de extração de características do SPAM (PEVNY; BAS; FRIDRICH, 2010)

A contribuição do SPAM na área de esteganálise é fundamental, dado que muitos algoritmos de esteganografia, como o HUGO, tentam minimizar custos diretamente relacionados ao filtro espacial do SPAM. Não somente isso, ele é a base do SRM, atual estado da arte que não utiliza técnicas de aprendizagem profunda.

2.2.2.3 Descritor SRM

SRM

Proposto inicialmente para detectar eficientemente o HUGO, a esteganálise baseada em ***S**patial **R**ich **M**odels* (**SRM**) tem processo semelhante ao introduzido na

Figura 9, sendo que a maior mudança ocorre na parte da filtragem a partir do *kernel* de gradiente simples (FRIDRICH; KODOVSKY, 2012). Da mesma maneira que no SPAM, análises empíricas da correlação entre pixels próximos em uma imagem natural levaram ao uso de 45 matrizes residuais para o cálculo dos descritores.

As matrizes de correlação podem ser observadas na Figura 10. Os *kernels* do tipo *spam* são filtros lineares simples, enquanto os *kernels* do tipo *minmax* são calculados a partir de operações de mínimo e máximo de múltiplos filtros lineares diferentes. O *kernel* 1d, por exemplo, gera dois resíduos a partir das equações: $min\{I_{i-1,j} - I_{i,j}, I_{i,j-1} - I_{i,j}, I_{i+1} - I_{i,j}\}$ e $max\{I_{i-1,j} - I_{i,j}, I_{i,j-1} - I_{i,j}, I_{i+1} - I_{i,j}\}$.

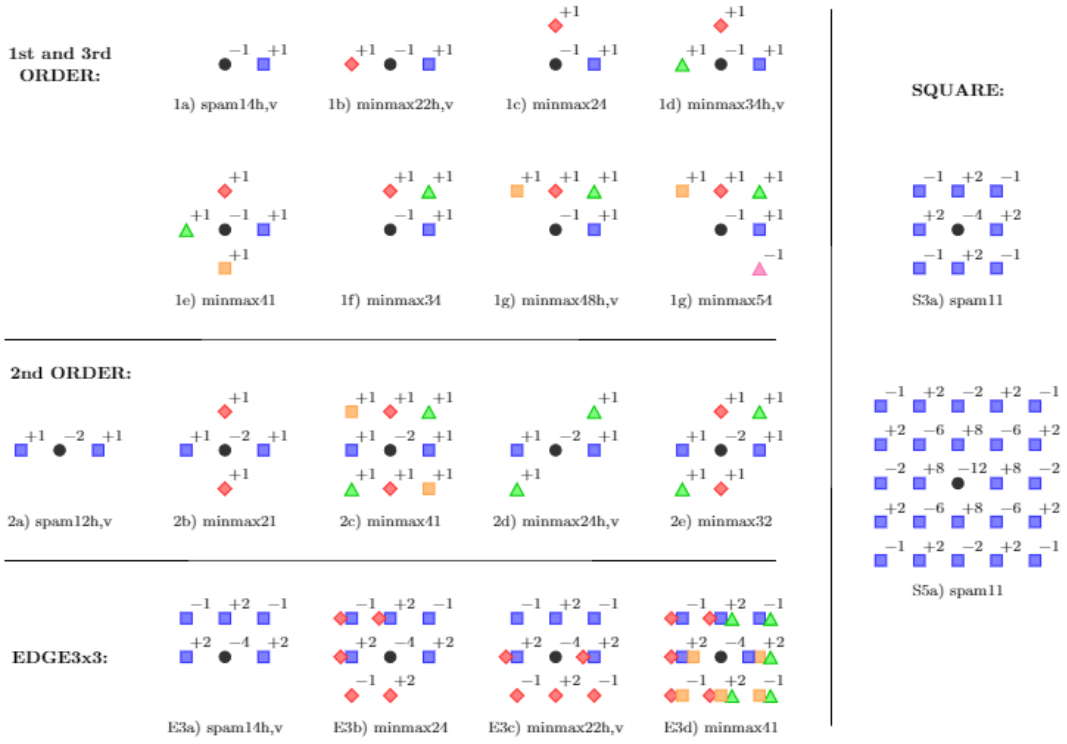


Figura 10 – *Kernels* de extração dos modelos SRM (FRIDRICH; KODOVSKY, 2012)

Assim como no SPAM, são geradas as matrizes de correlação a partir de cadeias de Markov. O parâmetro T, no entanto, serve para truncar valores fora do intervalo $[-T, T]$, ao contrário de ignorar esses valores, como ocorria no SPAM.

O descritor SRM e suas variações se consolidaram na área de esteganálise como estado da arte até os modelos de redes de aprendizagem profunda. Apesar disso a classificação via *Ensemble of Classifiers* e SRM ainda é muito utilizada devido a alta eficiência e o tempo reduzido de treino e teste com comparação métodos com aprendizagem profunda e é base para comparação de todos os algoritmos novos de

esteganálise.

2.2.2.4 Classificadores

Apesar da possibilidade da classificação de uma imagem de cobertura ser realizada com qualquer classificador, o uso de um *Ensemble of Classifiers* específico para esteganografia com SRM mostrou-se mais eficiente (KODOVSKY; FRIDRICH; HOLUB, 2012).

Ensemble of Classifiers

Ensemble of Classifiers é a nomenclatura para o uso de diversos classificadores ao invés de somente um, chegando à classificação final a partir de um indicador formado pelo resultado individual de cada classificador (como uma votação, por exemplo).

Essa abordagem teve sua introdução e disseminação no âmbito esteganográfico devido ao baixo custo computacional aliado a bons resultados para um conjunto de características de alta dimensionalidade.

A escalabilidade com o número de amostras para treino sem perda de performance na classificação, se igualando a classificadores bem estabelecidos, como SVM, em conjunto com as outras características já listadas tornam o *Ensemble of Classifiers* de FLDs ideal para o experimento proposto.

O modelo é composto pelo algoritmo de *bootstrapping* para seleção de características e de imagens de treino e teste com classificadores do tipo *Fisher Linear Discriminant* (FLD).

A quantidade de classificadores e o tamanho da dimensão de características são definidos pelo algoritmo de otimização proposto por Kodovsky, Fridrich e Holub (2012), que busca minimizar o ***Out-of-bag error*** (OOBe), que é uma estimativa imparcial do erro de teste para *Ensemble of Classifiers*. O valor do erro é calculado para cada sub classificador a partir da Equação 10 (FRIDRICH; KODOVSKY, 2012).

Out-of-bag error

$$E_{OOB}^{(L)} = \frac{1}{2N^{trn}} \sum_{m=1}^{N^{trn}} (B^{(L)}(\boldsymbol{x}^{(m)}) + 1 - B^{(L)}(\bar{\boldsymbol{x}}^{(m)})), \tag{10}$$

onde:

- N^{trn} é o número de amostras definida para cada classificador.
- $B^{(L)}(x)$ é o resultado da votação (imagem de cobertura = 0, estego imagem = 1) do classificador L para o conjunto de características x.
- $\boldsymbol{x}^{(m)}$ é o vetor de características da amostra de imagem de cobertura m do conjunto de treino relativo a um classificador contendo imagens de cobertura sem informação escondida.

- $\bar{\boldsymbol{x}}^{(m)}$ é recíproco a $\boldsymbol{x}^{(m)}$, porém para o conjunto de estego imagens.

2.3 Redes Neurais

Partindo da hipótese que a atividade mental consiste de reações eletroquímicas de uma rede de células chamadas neurônios, alguns trabalhos pioneiros focaram em criar redes neurais artificiais, dando origem a área da inteligencia artificial chamada de *conexionismo* ou *computação neural* (RUSSELL; NORVIG, 2009).

Um modelo matemático simples foi proposto em McCulloch e Pitts (1943) e pode ser visto na Figura 11. De forma simplificada, o modelo propõe um cálculo que realiza uma combinação linear das entradas conectadas e seus devidos pesos $w_{0,j}, w_{1,j}, \ldots, w_{n,j}$. O neurônio é então ativado através de uma função de ativação baseada em um limiar.

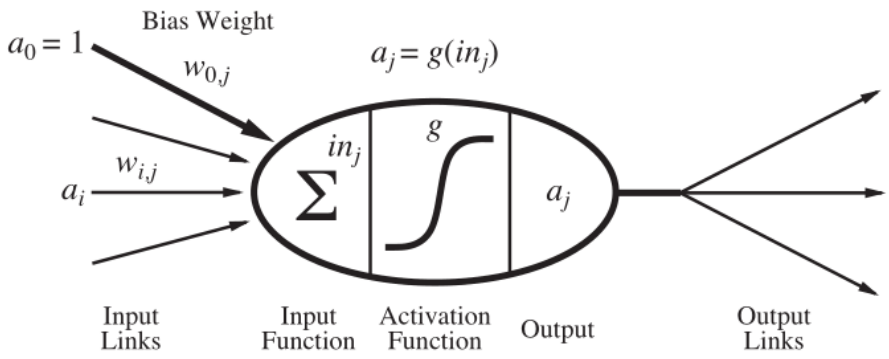


Figura 11 – Modelo matemático proposto em McCulloch e Pitts (1943), a imagem pode ser encontrada em (RUSSELL; NORVIG, 2009).

Rede neural

Uma **rede neural** é simplesmente um conjunto de unidades neuronais conectadas, sendo que suas propriedades e funções são determinadas pela topologia ou pelas funções de combinação linear e ativação.

Na fase de treinamento de uma rede neural, cada entrada previamente rotulada é classificada pela rede e, caso a saída da rede seja diferente do esperado, os pesos das ligações entre os neurônios são reajustados a partir do vetor gradiente computado pelo algoritmo de aprendizagem. O gradiente indica se o erro iria aumentar ou diminuir caso o peso da ligação aumentasse levemente. O processo de reajuste dos pesos é denominado *backpropagation*, pois se propaga entre as arestas de camadas cada vez mais superficiais até os neurônios iniciais da rede (LECUN; BENGIO; HINTON, 2015).

As funções de ativação abordadas neste trabalho são a tangente hiperbólica (TanH) e a unidade linear retificada (ReLU, *rectified linear unit*), definidas pelas

Equações 11 e 12.

$$\text{TanH}(x) = \frac{1 - e^{-\lambda x}}{1 + e^{-\lambda x}} \tag{11}$$

$$\text{ReLU}(x) = \max(0, x) \tag{12}$$

2.3.1 Aprendizagem Profunda

Aprendizagem profunda

Uma rede de **aprendizagem profunda** (em inglês, *Deep Learning*) é resumida estruturalmente como uma rede neural de diversas camadas intermediárias. Diversos trabalhos mostram que a sua performance é superior a métodos preliminares, fazendo com que estudos de redes neurais profundas sejam cada vez mais utilizados em diversas áreas do conhecimento (LECUN et al., 1990; LECUN et al., 1998; CHEN et al., 2015; LU et al., 2016; ZHONG; LIU; HUA, 2016).

Em um modelo tradicional de algoritmos de aprendizagem de máquina, o primeiro passo para a classificação é a extração de descritores condizentes com o problema, seguido da rotulagem do componente a partir de um classificador genérico. No caso da aprendizagem profunda as próprias camadas internas são responsáveis pela extração das características e pela classificação (LECUN et al., 1998).

A principal consequência da rede ser responsável pelo processo inteiro de classificação é a rede operar no modelo *black box*, no qual há muito pouco controle sobre o seu treinamento. Quanto mais profunda é a camada, maior o nível de abstração dos dados e, consequentemente, menor é o controle sobre a respectiva etapa do processo.

O modelo de arquitetura utilizado neste trabalho é o de Redes Neurais Convolucionais (CNN, *Convolutional Neural Network*), o qual é muito empregado quando a entrada é uma matriz, como no caso de imagens digitais.

2.3.2 *Convolutional Neural Networks*

CNN

A principal diferença da **CNN** para uma rede neural convencional é o fato de que ela faz uso de camadas de convolução e subamostragem (LECUN; BENGIO; HINTON, 2015). As camadas de convolução são filtros lineares cujos *kernels* tem os pesos ajustados no processo de *backpropagation* e as camadas de subamostragem servem para reduzir a imagem para uma resolução menor. Normalmente o processo de subamostragem é realizado a partir de operações de *maxpooling*, *minpooling* e *averagepooling*, que consistem respectivamente na seleção do maior, menor e média

dos valores em uma janela de pixels. Um exemplo de *averagepooling* pode ser visto na Figura 12.

A união dos subprocessos de modo a formar uma CNN completa pode ser vista na Figura 13.

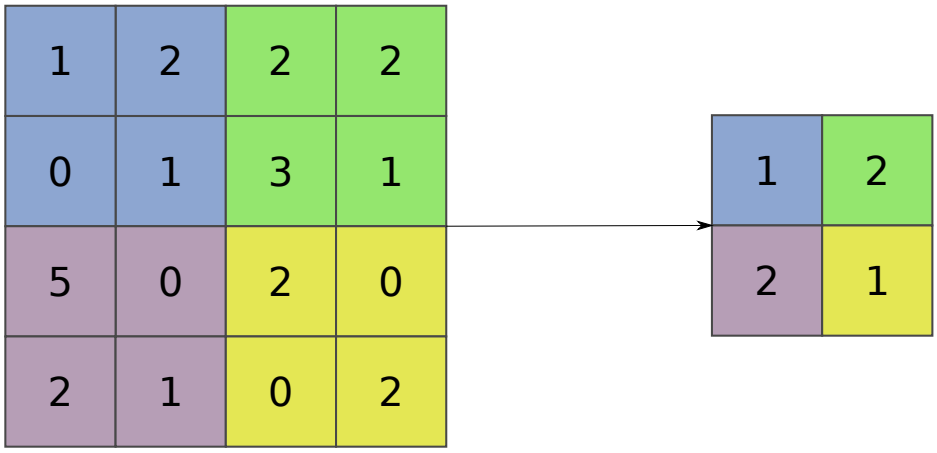


Figura 12 – Exemplo de *averagepooling*, com uma janela de 2x2 e deslocamento (*stride*) de 2, onde o a subamostragem é feita pela média dos valores em cada janela (Autoria própria).

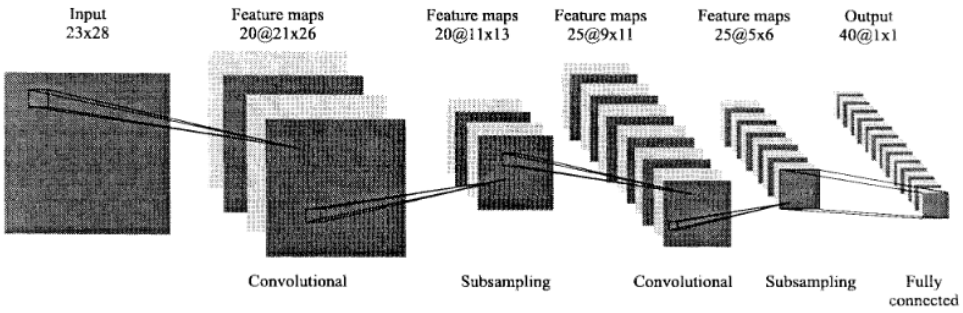


Figura 13 – Exemplo de uma CNN convencional (LAWRENCE et al., 1997).

2.4 Esteganálise via Aprendizagem Profunda

Um modelo de esteganálise que faz uso de aprendizagem profunda independe do uso de qualquer descritor, logo todos os dados para treino e teste são as próprias imagens. A partir disso, a própria rede gera os descritores para a imagem e a classifica como imagem de cobertura ou estego imagem.

O uso de aprendizagem profunda para esteganálise é recente, sendo que o primeiro modelo utilizando CNN foi proposto em 2014 por Tan e Li (2014), uma rede convolucional empilhada com *auto-encoder* e os resultados foram significativamente inferiores ao SRM. Em 2015, Qian et al. (2015) apresentaram resultados tão bons quanto os resultados do maxSRM (variação do SRM) com uma rede convolucional utilizando uma função ativação nova chamada de ativação gaussiana. Xu, Wu e Shi (2016) desenvolveram uma arquitetura de CNN que incorpora um conhecimento do domínio da esteganálise, utilizando o *kernel* do SPAM como pré-processamento. Na sequência, Xu, Wu e Shi (2016) propuseram um modelo de *ensemble of CNN* que melhorou a acurácia do modelo anterior. O atual estado da arte provém de uma CNN residual extremamente complexa com 50 camadas de convolução (WU; ZHONG; LIU, 2017). A rede errou em 4.9% dos casos para o algoritmo MiPOD com *payload* de 0.4 bpp, resultado muito acima de qualquer outro na área.

3 Metodologia

A metodologia empregada neste trabalho está organizada nos seguintes passos principais:

Etapa 1 Aquisição da base de dados BOSSBase (BAS; FILLER; PEVNỲ, 2011).

Etapa 2 Aplicação dos algoritmos HUGO, MiPOD e S-UNIWARD nas imagens da base BOSSBase para criar as estego imagens, as quais irão compor as bases de treinamento e teste.

Etapa 3 Realização do procedimento de esteganálise utilizando a abordagem tradicional: extração de descritores (SRM) e classificação (*Ensemble of Classifiers*).

Etapa 4 Realização do procedimento de esteganálise utilizando-se uma CNN.

Etapa 5 Análise comparativa dos resultados obtidos a partir das Etapas 3 e 4. Os resultados desta etapa serão discutidos em detalhes no próximo capítulo.

As seções a seguir detalham como esses passos serão realizados.

3.1 Base de Dados

Neste trabalho foi utilizada a base de imagens denominada BOSSBase (v1.01)[1], a qual foi criada para a competição de esteganálise *Break Our Steganographic System* (BOSS) (BAS; FILLER; PEVNỲ, 2011). É composta por 10000 imagens sem compressão, em escala de cinza, com dimensões 512x512 e provenientes de 7 câmeras diferentes, produzindo assim, padrões de ruídos de captação diferentes para não gerar especializações em algoritmos de esteganografia que utilizam abordagens adaptativas e exploram esses ruídos.

A competição BOSS buscava inicialmente incentivar a criação de algoritmos de esteganálise para quebrar o HUGO. A sua importância na área de esteganografia se consolidou após o grupo vencedor, denominado *"HUGO breakers"*, propor o uso dos descritores SRM.

Alguns exemplos da base de dados BOSS podem ser vistos na Figura 14.

3.2 Criação das Estego Imagens

Visando criar diferentes cenários de treinamento e teste, os algoritmos de esteganografia HUGO (PEVNỲ; FILLER; BAS, 2010), S-UNIWARD (HOLUB; FRI-

[1] Disponível em: <http://dde.binghamton.edu/download/ImageDB/BOSSbase_1.01.zip> (acessado em 11/2017).

Figura 14 – Exemplos de amostras da base BOSS.

DRICH; DENEMARK, 2014) e MiPOD (SEDIGHI; COGRANNE; FRIDRICH, 2016) (Subseções 2.1.2.2 - 2.1.2.4) foram aplicados nas imagens da base de dados BOSSBase, considerando-se diferentes parâmetros. Tais cenários permitem avaliar o desempenho das abordagens de esteganálise consideradas neste trabalho, tanto ao comparar a combinação de diferentes algoritmos de esteganografia nas etapas de treinamento e teste quando ao avaliar o impacto da utilização de diferentes *payloads*.

Os cenários de treinamento são listados na Tabela 1. Em resumo, as 5000 imagens iniciais da BOSSBase foram utilizadas para aplicação dos algoritmos HUGO, S-UNIWARD e MiPOD para quatro diferentes *payloads*: 0.1, 0.2, 0.4 e 0.6. Portanto, foram criados 12 diferentes conjuntos de treinamento, os quais serão referenciados no decorrer deste trabalho pela sigla presente na primeira coluna (formada pela primeira letra do nome do algoritmo seguida por um valor que representa o *payload*).

As estego imagens para os testes foram geradas a partir das 5000 últimas imagens da BOSSBase, criando os cenários de teste listados na Tabela 2. Observe que, além da variação do *payload*, foram utilizados também diferentes valores de STC. Os

Nomenclatura	Algoritmo	*Payload*
H1	HUGO	0.1
H2		0.2
H4		0.4
H6		0.6
S1	S-UNIWARD	0.1
S2		0.2
S4		0.4
S6		0.6
M1	MiPOD	0.1
M2		0.2
M4		0.4
M6		0.6

Tabela 1 – Cenários de treinamento.

parâmetros para o HUGO (STC, γ e σ) foram extraídos de Pevnỳ, Filler e Bas (2010)[2]. Para o S-UNIWARD, seguiu-se a recomendação disponível no executável do programa, que dizia para variar a altura STC de 7 a 12 para melhores resultados. O MiPOD foi executado com valores *default* dos parâmetros e com o codificador padrão (sem STC).

Para facilitar a discussão dos resultados, serão consideradas as siglas definidas na primeira coluna, as quais são formadas pela primeira letra do nome do algoritmo seguida pelos valores do STC e do *payload*.

As implementações dos algoritmos de esteganografia HUGO, S-UNIWARD e MiPOD estão disponíveis na página da universidade de Binghamton[3].

3.3 Esteganálise Baseada no Descritor SRM e Classificação com *Ensemble of Classifiers*

Um conjunto (*ensemble*) de classificadores é um paradigma de aprendizado de máquina em que vários classificadores são treinados para resolver o mesmo problema. Como detalhado na Seção 2.2.2.4, neste trabalho um conjunto de hipóteses é induzido separadamente usando *Fisher Linear Discriminant* (FLD) como classificadores base, e depois combinado através de um método de consenso.

Seguindo a abordagem proposta por Kodovsky, Fridrich e Holub (2012), a seleção das imagens de treinamento para extração de características (representadas pelo descritor SRM) foi realizada por *bootstrapping*, considerando uma amostragem

[2]Foi gerado também um cenário do HUGO com STC de altura 10, $\gamma = 1$ e $\sigma = 5$. Porém, como não houve uma variação significativa nos resultados, esse cenário foi descartado.

[3]Disponível em: <http://dde.binghamton.edu/download/stego_algorithms/>.

Nomenclatura	Algoritmo	STC	*Payload*
H.0.1	HUGO	0	0.1
H.0.2			0.2
H.0.4			0.4
H.0.6			0.6
H.10.1		10, $\gamma = 2$ e $\sigma = 5$	0.1
H.10.2			0.2
H.10.4			0.4
H.10.6			0.6
S.0.1	S-UNIWARD	0	0.1
S.0.2			0.2
S.0.4			0.4
S.0.6			0.6
S.7.1		7	0.1
S.7.2			0.2
S.7.4			0.4
S.7.6			0.6
S.10.1		10	0.1
S.10.2			0.2
S.10.4			0.4
S.10.6			0.6
S.12.1		12	0.1
S.12.2			0.2
S.12.4			0.4
S.12.6			0.6
M.-.1	MIPOD	Não se aplica	0.1
M.-.2			0.2
M.-.4			0.4
M.-.6			0.6

Tabela 2 – Cenários de teste.

uniforme com substituição. A classificação final é definida por uma votação majoritária do resultado de cada classificador base.

Foram utilizadas implementações prontas do SRM (em C++[4]) e também do *Ensemble of Classifiers* (em MATLAB[5]).

3.3.1 Experimentos

Para a etapa de treinamento, foram empregados os 12 conjuntos listados na Tabela 1, que englobam os três algoritmos de esteganografia para cada um dos

[4]Disponível em: <http://dde.binghamton.edu/download/feature_extractors/>
[5]Disponível em: <http://dde.binghamton.edu/download/ensemble/>

quatro *payloads*. Foram utilizadas 5000 imagens da BOSSBase e suas respectivas estego imagens para construção da rede de classificadores, a qual foi inicializada com os valores *default* dos parâmetros, os quais buscam minimizar o OOBe com a variação do número de características e de classificadores (FLDs) (ver detalhes na Seção 2.2.2.4).

Na etapa de testes, são usados os 5000 pares restantes de estego imagem e imagem de cobertura da BOSSBase, as quais são classificadas por uma votação majoritária dos FLDs. O indicador responsável pela classificação é estruturado a partir do somatório dos resultados apontados por cada classificador individualmente, que são limitados a -1 caso a classificação seja de imagem de cobertura e 1, caso o contrário. Dessa maneira, o valor 0 indica total incerteza, e os extremos $-N$, $+N$ indicam certeza máxima, onde N é o número de classificadores.

Para cada treinamento realizado, foram considerados os seguintes conjuntos de testes: H.0.x, H.10.x, S.0.x, S.10.x, M.-.x, onde x $\in \{1,2,4,6\}$ (Tabela 2). Os demais não foram utilizados por limitação de tempo.

3.4 Esteganálise Baseada em CNN

A abordagem de esteganálise com aprendizagem profunda empregada neste trabalho se baseia na proposta por Xu, Wu e Shi (2016), que utiliza uma rede neural convolucional (CNN) para predizer se uma imagem possui ou não uma mensagem escondida. A arquitetura da CNN utilizada pode ser vista na Figura 15.

Inicialmente, é realizado um pré-processamento nas imagens, com a aplicação de um filtro passa-alta (HPF) com o seguinte *kernel*:

$$\frac{1}{12}\begin{bmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{bmatrix},$$

o qual serve para destacar o ruído residual, ajudando na esteganálise visto que os dados embutidos na imagem tendem a gerar distorções. Tal *kernel* foi inicialmente empregado por Kodovsky, Fridrich e Holub (2011) para esteganálise do algoritmo HUGO. Porém, por ter apresentado bons resultados também com outros algoritmos de esteganografia, é amplamente utilizado em vários modelos de esteganálise.

Após esse pré-processamento as imagens passam pela CNN, que pode ser dividida em seis grupos, sendo os cinco primeiros grupos convolucionais, e o último grupo de classificação linear. Os grupos convolucionais iniciam com uma camada de

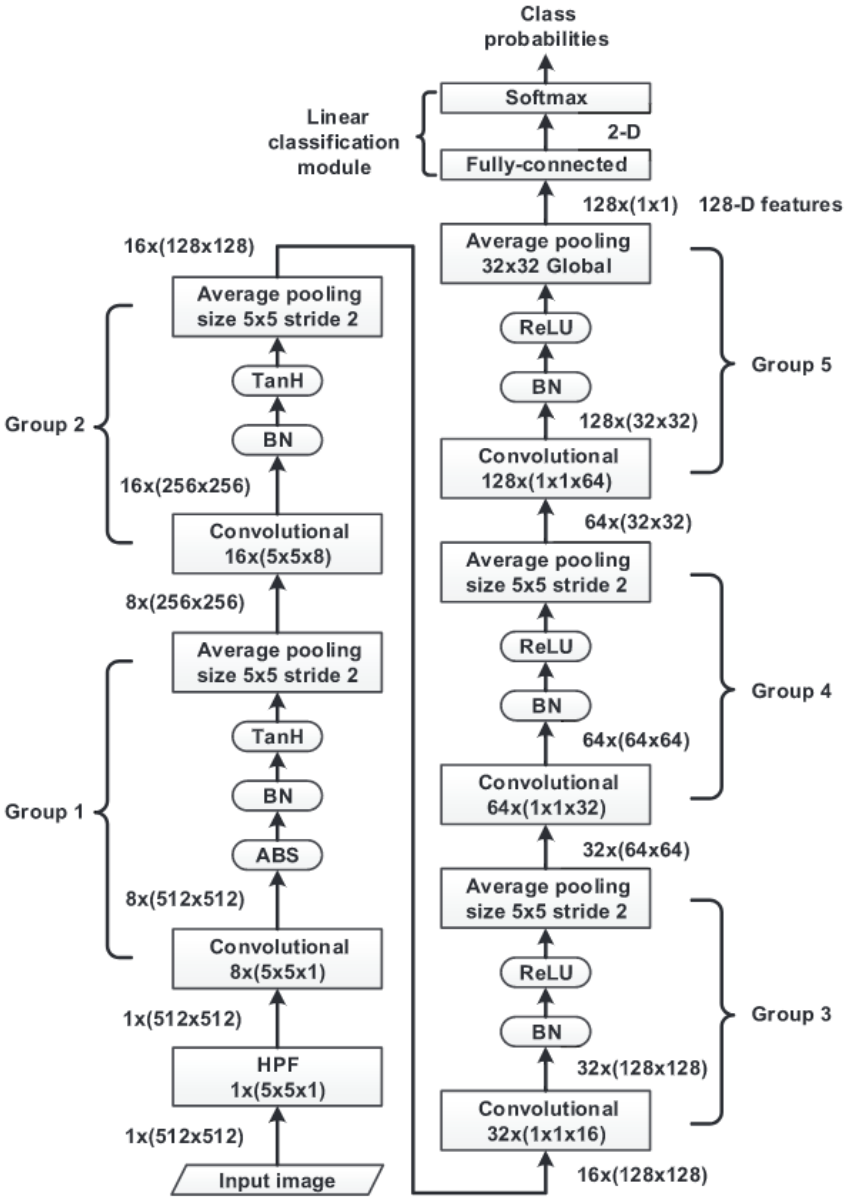


Figura 15 – Arquitetura da CNN utilizada. Dentro das caixas é mostrado o tipo da camada e seus parâmetros (XU; WU; SHI, 2016).

convolução e terminam com uma camada de média de subamostragem (*average pooling*) com um *kernel* de 5×5 - com exceção do quinto grupo, onde a subamostragem é global.

Nos grupos 1 e 2, é utilizada a função de ativação não-linear tangente hiperbólica (TanH), enquanto nos grupos 3, 4 e 5 é usada a Unidade Retificada Linear (ReLU). No grupo 1 também é inserida uma camada de ativação modular (ABS), a qual calcula o valor absoluto de cada valor do mapa de características. Após essa camada, e após as

camadas convolucionais nos outros grupos, é realizada uma normalização do conjunto (*Batch Normalization* - BN) (IOFFE; SZEGEDY, 2015), que objetiva impedir que o treinamento caia em um mínimo local.

Por fim, a parte de classificação linear recebe um mapa de 128 características em sua camada totalmente conectada, o que gera dois valores. Após a aplicação da função *softmax* (BISHOP, 2006), eles representam a probabilidade da imagem ser de cobertura ou estego.

3.4.1 Experimentos

Para o treinamento da rede, foram utilizados os 12 conjuntos listados na Tabela 1, gerados a partir da aplicação dos algoritmos HUGO, S-UNIWARD e MiPOD em 5000 imagens da base BOSSBase para diferentes *payloads*.

As 5000 imagens de cada conjunto de treinamento foram divididas igualmente em cinco subconjuntos sem sobreposição, a partir dos quais foram gerados cinco modelos de CNN. Em cada um deles, um dos subconjuntos foi utilizado para validação e os quatro restantes para treinamento. Isto é feito como forma controlada de *bootstrap aggregation* (BREIMAN, 1996), visando criar uma combinação de classificadores.

Na fase de testes, as imagens dos 28 cenários da Tabela 2 são classificadas pelas cinco CNNs treinadas. A predição final é calculada com base na média das probabilidades para cada imagem. Observe que esse procedimento é realizado para cada um dos 12 conjuntos de treinamento.

Cada uma das CNNs foi treinada por 30000 iterações, com lotes de 64 imagens (32 pares) para cada iteração. A taxa de aprendizagem (*learning rate*) é inicializada em 0.001 sendo reduzida em 10% a cada 5000 iterações. O *momentum* é fixado em 0.9. O *bias* foi desativado em todas as camadas convolucionais.

A implementação da CNN foi realizada em Python utilizando a biblioteca TensorFlow. Os experimentos foram executados em uma GeForce GTX Titan Xp, o tempo total de treinamento com cada conjunto foi de aproximadamente 22 horas e o tempo de uma bateria completa de testes, com os 28 cenários, foi de cerca de 2 horas.

3.5 Comparação dos Resultados

Para comparar o desempenho da abordagem proposta neste trabalho em relação ao uso de um *Ensemble of Classifiers* (com descritor SRM), serão utilizadas duas métricas bastante disseminadas no âmbito da esteganálise: a acurácia e a área sob a curva *Receiving Operating Characteristic* (ROC).

3.5.1 Acurácia

A acurácia de um classificador é uma medida simples dada pela quantidade de amostras classificadas de maneira correta divida pelo total de amostras. Essa medida é muito utilizada devido a sua simplicidade e objetividade.

3.5.2 *Area Under Curve* (AUC)

Uma curva ROC é uma representação gráfica do desempenho de um classificador binário. Ela consiste em um gráfico da taxa de verdadeiros positivos versus a taxa de falsos positivos. Ela é uma medida particularmente útil em domínios onde ocorre uma grande desproporção entre as classes ou quando se deve considerar diferentes custos/benefícios para os diferentes erros/acertos de classificação.

Dado que detectores diferentes podem ter intersecções entre suas curvas ROCs, causando dificuldades na comparação, pode-se considerar a área sob a curva (AUC - do inglês, *Area Under Curve*) como medida de comparação.

Uma vez que a área abaixo da curva ROC é uma fração da área de um quadrado de lado um, o seu valor, p, está sempre entre 0 e 1. Quando $p \approx 0{,}5$ a curva ROC está próxima à linha diagonal, ou seja, equivale ao classificador aleatório, e quando $p \approx 1$ a curva ROC está mais próxima dos detectores perfeitos.

4 Análise e Discussão dos Resultados

Este capítulo discute os resultados obtidos nas duas abordagens de esteganálise estudadas neste trabalho. Como descrito anteriormente, a primeira é baseada no descritor SRM - posteriormente utilizado em um conjunto (*ensemble*) de classificadores - e a segunda em CNNs.

Conforme detalhado no capítulo anterior, são considerados 12 conjuntos de treinamento, resultantes da aplicação dos algoritmos de esteganografia HUGO, S-UNIWARD e MiPOD para quatro diferentes *payloads*: 0.1, 0.2, 0.4 e 0.6 bpp. Visando facilitar a identificação de um conjunto específico, serão consideradas as siglas definidas na Tabela 1, compostas pela primeira letra do nome do algoritmo de esteganografia seguida do valor do *payload* (por exemplo, H6 representa HUGO com *payload* 0.6 bpp).

Além disso, também são definidos 28 conjuntos de teste, listados na Tabela 2. Cada um deles é identificado por uma sigla composta pela primeira letra do nome do algoritmo de esteganografia utilizado para gerar a estego imagem, seguida por dois valores: um que representa o STC e outro que identifica o *payload* (por exemplo, H.0.2 representa o HUGO com STC 0 e *payload* 0.2 bpp).

Cada um dos 12 conjuntos de treinamento é utilizado separadamente para treinamento das duas abordagens de esteganálise. Os modelos de classificação resultantes (*ensemble* de classificadores ou CNN) são então aplicados em diferentes conjuntos de teste (x denota a variação para os quatro *payloads*):

1. SRM + *Ensemble*: sete conjuntos - H.0.x, H.10.x, S.0.x, S.10.1, S.10.4, S.10.6, M.0.x[1].
2. CNN: todos os 28 conjuntos listados na Tabela 2 - H.0.x, H.10.x, S.0.x, S.7.x, S.10.x, S.12.x, M.-.x.

Com isso, tem-se um total de $12 \times 7 + 12 \times 28 = 420$ resultados.

Na sequência, serão discutidos os resultados obtidos, os quais serão divididos por abordagem de esteganálise e depois por conjunto de treinamento. Ao final, será realizada uma análise geral dos resultados. Os Apêndices A e B mostram tabelas das áreas sob a curva (AUC) obtidas nos testes com SRM e CNN respectivamente.

[1] Por limitação de tempo não foram realizados os demais testes.

4.1 Resultados com SRM e *Ensemble* de Classificadores

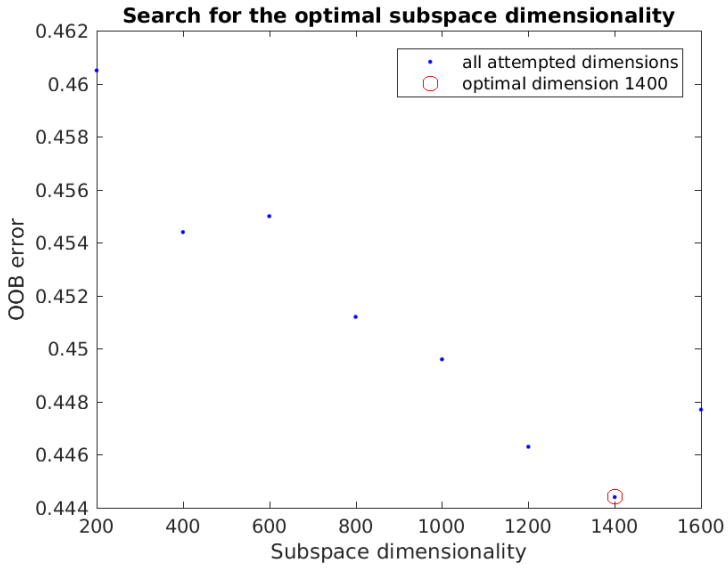
Em geral, o comportamento da seleção de características é semelhante para todos *payloads*de 1000 à 1400 características. No caso do número de classificadores, os dois *payloads* menores variam de 70 à 80 enquanto os maiores variam de 90 à 100. Tais comportamentos são ilustrado na Figura 16, para o caso do conjunto de treinamento M1.

Outro fato interessante é o comportamento dos classificadores base para os diferentes *payloads* de teste. A Figura 17(a) mostra a assertividade quando o *payload* de treinamento e teste são próximos, dado que existe uma tendência do indicador estar próximo da extremidade e com os tipos de imagens, estego ou cobertura, na posição desejada.

Além disso, quando o classificador treinado em um conjunto de treinamento com *payload* alto é aplicado em um conjunto de testes que possui *payload* baixo, o histograma tende a estar mais elevado para a esquerda tanto para as imagens de cobertura quanto para as estego imagens. Isso ocorre porque, como o *payload* de teste é muito menor que o de treino, a rede confunde quase todas as imagens com imagens de cobertura. Esse comportamento é recíproco para *payloads* de treino e teste inversos, porém a confusão ocorre com estego imagens (um exemplo pode ser visto na Figura 17(b)).

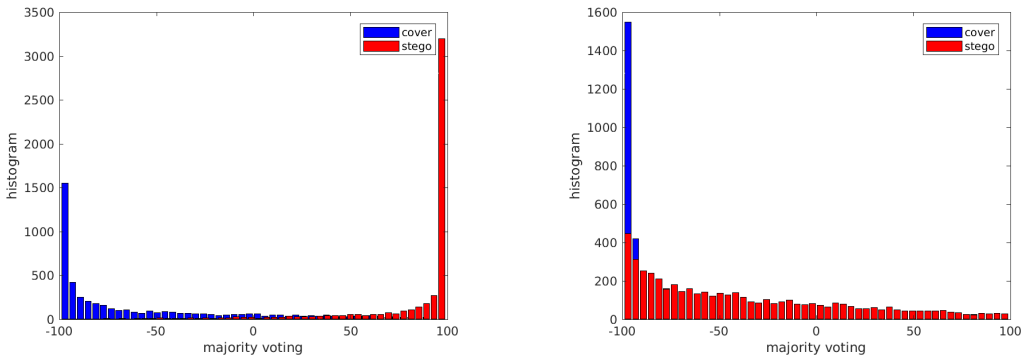
As seções a seguir apresentarão os resultados da classificação das imagens com o uso do descritor SRM para os 12 diferentes conjuntos de treino.

(a)



(b)

Figura 16 – (a) Variação do OOBe com relação ao número de classificadores para o cojunto de treinamento M1. (b) Variação do OOBe com relação ao número de características para o conjunto de treinamento M1 (Autoria própria).



(a) (b)

Figura 17 – (a) Histograma de votação para o conjunto de treino H4 e de teste H.0.6. (b) Histograma de votação para o conjunto de treino H6 e de teste H.0.1. (Autoria própria)

4.1.1 Conjunto de Treinamento H1

Os resultados para o conjunto de treinamento H1 podem ser vistos na Tabela 3. Os melhores resultados foram obtidos com o *payload* de teste igual ao de treino e resultados inferiores nos demais cenários de teste. Essas situações mostram as principais características observadas ao se usar o descritor SRM com *ensemble* de classificadores: melhores performances quando se usando *payload* próximo e mesmo algoritmo de treino e teste, baixa performance ao testar com *payloads* muito distintos do de teste, além da facilidade na detecção do algoritmo HUGO em relação aos outros dois algoritmos.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	66,23%	72,48%	77,57%	79,31%
H.10.x	**67,02%**	**73,25%**	**77,79%**	**79,42%**
S.0.x	58,78%	66,91%	74,68%	78,11%
S.10.x	59,78%	67,71%	75,20%	78,24%
M.-.x	57,86%	64,77%	71,92%	75,61%

Tabela 3 – Acurácias dos testes do SRM com o conjunto de treinamento H1.

4.1.2 Conjunto de Treinamento H2

Do mesmo modo que para o conjunto de treino anterior, os melhores resultados foram obtidos para os cenários de teste com uso do HUGO. Porém, os resultados para os conjuntos H.0.4, H.0.6, H.10.4, H.10.6 foram muito superiores. Os resultados podem ser vistos na Tabela 4.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	63,24%	73,89%	82,23%	84,47%
H.10.x	**65,03%**	**75,30%**	**82,70%**	**84,69%**
S.0.x	52,83%	64,02%	77,11%	82,51%
S.10.x	53,95%	65,86%	78,22%	82,89%
M.-.x	52,70%	61,50%	73,73%	79,95%

Tabela 4 – Acurácias dos testes do SRM com o conjunto de treinamento H2.

4.1.3 Conjunto de Treinamento H4

A Tabela 5 mostra que o conjunto apresentou, como esperado, os melhores resultados para o algoritmo HUGO com o *payload* de 0.4 bpp e valores acima de 89% para os cenários H.10.6 e H.0.6, os quais só não superaram o conjunto de treino H6.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	53,55%	60,80%	84,60%	89,11%
H.10.x	**54,85%**	**70,39%**	**85,43%**	**89,36%**
S.0.x	50,75%	53,75%	73,42%	84,48%
S.10.x	51,04%	55,46%	75,67%	85,36%
M.-.x	50,78%	52,90%	68,50%	80,17%

Tabela 5 – Acurácias dos testes do SRM com o conjunto de treinamento H4

4.1.4 Conjunto de Treinamento H6

Nos experimentos, o conjunto de treinamento H6 obteve os melhores resultados para os cenários H.10.6 e H.0.6 com 91,81% e 90,46% de acurácia respectivamente. Os resultados podem ser vistos na Tabela 6.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	51,13%	58,98%	**81,30%**	90,46%
H.10.x	**51,46%**	**61,14%**	77,79%	**91,18%**
S.0.x	50,41%	51,64%	64,44%	79,46%
S.10.x	50,46%	51,94%	66,66%	81,52%
M.-.x	50,38%	51,28%	60,05%	73,73%

Tabela 6 – Acurácias dos testes do SRM com o conjunto de treinamento H6.

4.1.5 Conjunto de Treinamento S1

Todos os melhores resultados para o algoritmo S-UNIWARD foram obtidos com os conjuntos treinados com o mesmo algoritmo e mesmo *payload*. No caso do S1, a Tabela 7 mostra que foram obtidas acurácias interessantes para os cenários H.0.1, H.10.1, S.0.1, S.10.1 e M.1, mesmo que com acurácia muito abaixo da obtida pelos conjuntos treinados e testados com o HUGO para o mesmo *payload*.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	59,54%	66,68%	74,59%	79,54%
H.10.x	61,19%	66,43%	75,22%	78,76%
S.0.x	**61,53%**	**69,97%**	77,10%	**79,87%**
S.10.x	61,09%	69,75%	**77,46%**	80,22%
M.-.x	59,85%	66,06%	73,50%	77,28%

Tabela 7 – Acurácias dos testes do SRM com o conjunto de treinamento S1.

4.1.6 Conjunto de Treinamento S2

A Tabela 8 apresenta os resultados para o conjunto de treino S2. Como já dito anteriormente, o conjunto obteve os melhores resultados para o cenário S.0.2, além de resultados bons para o *payload* de 0.4 bpp.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	58,65%	69,30%	79,94%	84,37%
H.10.x	59,67%	69,50%	79,63%	83,82%
S.0.x	59,20%	**70,40%**	81,24%	**84,90%**
S.10.x	**60,59%**	71,49%	**82,01%**	84,99%
M.-.x	57,60%	66,79%	76,84%	82,05%

Tabela 8 – Acurácias dos testes do SRM com o conjunto de treinamento S2.

4.1.7 Conjunto de Treinamento S4

A única exceção a regra, o conjunto S4 obteve tanto os melhores resultados dentre todos os conjuntos de treino tanto para o cenário S.0.4 quanto para o S.0.6. Seus resultados podem ser vistos na Tabela 9.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	53,98%	65,23%	79,86%	88,17%
H.10.x	**55,33%**	**66,08%**	81,29%	87,86%
S.0.x	52,81%	64,08%	81,89%	**89,64%**
S.10.x	53,68%	66,31%	**83,31%**	88,95%
M.-.x	52,42%	61,86%	77,99%	85,34%

Tabela 9 – Acurácias dos testes do SRM com o conjunto de treinamento S4.

Em comparação ao conjunto S2, os resultados para o *payload* 0.1 foram significativamente melhores, enquanto que o desempenho para o *payload* 0.6 aumentou. Isso sugere que a capacidade do classificador ao ser treinado com 0.4 o torna mais robusto a maiores alterações na imagem, mas prejudica os casos em que há pouca alteração.

4.1.8 Conjunto de Treinamento S6

O pior dos conjuntos de treino com o algoritmo S-UNIWARD, só obteve resultados satisfatórios para os cenários de teste H.0.6, H.10.6, S.0.6 e M.6. Os resultados para o conjunto de treino S6 podem ser vistos na Tabela 10.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	51,50%	58,89%	76,08%	87,21%
H.10.x	**51,86%**	**59,80%**	78,94%	88,31%
S.0.x	51,12%	56,29%	76,67%	**88,93%**
S.10.x	51,27%	58,51%	**78,98%**	89,71%
M.-.x	50,98%	55,92%	72,69%	84,05%

Tabela 10 – Acurácias dos testes do SRM com o conjunto de treinamento S6.

4.1.9 Conjunto de Treinamento M1

Conforme observado na Tabela 11, assim como em outros algoritmos listados na Tabelas 12, 13 e 14, o MiPOD também é sensível a cenários de teste que fazem uso de outro algoritmo no treinamento.

A Tabela 11 por exemplo mostra a melhor acurácia para o cenário M.1, 61,17%. Em compensação os resultados para todos os outros *payloads* foram ruins.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	60,02%	66,74%	74,31%	78,09%
H.10.x	**61,65%**	66,88%	74,78%	79,29%
S.0.x	60,94%	**67,93%**	76,36%	**79,42%**
S.10.x	61,49%	69,38%	**76,69%**	79,55%
M.-.x	61,17%	66,65%	73,55%	77,41%

Tabela 11 – Acurácias dos testes do SRM com o conjunto de treinamento M1.

4.1.10 Conjunto de Treinamento M2

Curiosamente, assim como no conjunto M1, os melhores resultados, com exceção do *payload* de 0.1 bpp foram obtidos utilizando o algoritmo S-UNIWARD para testes. O melhor resultado para o cenário M.2 foi com esse conjunto. Os resultados podem ser observados na Tabela 12.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	59,91%	68,41%	77,63%	81,99%
H.10.x	**61,03%**	68,94%	78,29%	82,22%
S.0.x	59,14%	**69,22%**	79,14%	**82,90%**
S.10.x	60,49%	70,22%	**79,96%**	83,11%
M.-.x	58,85%	67,91%	76,87%	81,44%

Tabela 12 – Acurácias dos testes do SRM com o conjunto de treinamento M2.

4.1.11 Conjunto de Treinamento M4

A Tabela 13 mostra os resultados para o conjunto de treinamento M4, que não teve nenhum resultado muito interessante além do melhor resultado para o conjunto M.-.4.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	54,06%	65,47%	79,80%	85,64%
H.10.x	**55,20%**	**66,45%**	80,66%	86,27%
S.0.x	52,50%	63,31%	80,50%	**86,57%**
S.10.x	53,33%	65,29%	**81,68%**	87,03%
M.-.x	52,58%	62,96%	78,31%	85,04%

Tabela 13 – Acurácias dos testes do SRM com o conjunto de treinamento M4.

4.1.12 Conjunto de Treinamento M6

O conjunto M6 tem comportamento semelhante ao conjunto anterior, com a exceção de que possui o melhor resultado para o conjunto de teste M.-.6. Os resultados estão dispostos na Tabela 14.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	51,11%	57,29%	76,52%	86,59%
H.10.x	**51,42%**	**59,04%**	**78,25%**	**87,60%**
S.0.x	50,75%	53,82%	73,91%	87,17%
S.10.x	51,11%	54,88%	76,73%	88,19%
M.-.x	51,11%	54,35%	74,60%	85,91%

Tabela 14 – Acurácias dos testes do SRM com o conjunto de treinamento M6.

4.2 Resultados com CNN

Como descrito anteriormente, as CNNs treinadas com cada um dos 12 conjuntos de treinamento são utilizadas para classificação de todas as 28 bases de teste. Para a maioria dos conjuntos os melhores resultados foram obtidos nos testes com o algoritmo HUGO com STC de altura 10. As CNNs treinadas para *payloads* de 0.2 e 0.4 bpp apresentaram uma boa capacidade de generalização. Enquanto as treinadas com os demais *payloads* se mostraram mais especializadas.

Nas seções a seguir são mostrados os resultados obtidos com cada conjunto de treinamento.

4.2.1 Conjunto de Treinamento H1

Os resultados obtidos com o conjunto de treinamento H1, ou seja, aquele onde foi utilizado o algoritmo HUGO com *payload* de 0.1 bpp, pode ser visto na Tabela 15. Mostrou resultados satisfatórios apenas quando testado em bases com o mesmo algoritmo usado no treinamento, especialmente nos cenários H.0.1 e H.10.1, onde apresentou a segunda maior taxa de acertos entre todas as CNNs.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	54,74%	63,72%	77,68%	83,97%
H.10.x	**56,61%**	**66,96%**	**79,74%**	**84,86%**
S.0.x	51,49%	55,68%	70,62%	81,53%
S.7.x	52,15%	57,70%	74,29%	82,93%
S.10.x	52,01%	57,07%	72,87%	82,31%
S.12.x	51,96%	56,90%	72,07%	82,35%
M.-.x	51,99%	57,81%	72,76%	81,76%

Tabela 15 – Acurácias dos testes da CNN com o conjunto de treinamento H1.

4.2.2 Conjunto de Treinamento H2

Assim como o anterior, este conjunto também apresentou melhores acurácias nos testes com o algoritmo HUGO. Fato que já era esperado, visto que este é o algoritmo mais simples dos testados, além de ser o usado no treinamento desta CNN. A Tabela 16 expõe os resultados dos testes.

Apesar da acurácia obtida nos cenários de teste H.0.1 e H.10.1 ser inferior àquela alcançada pelo conjunto de treinamento H1, a área sob a curva (AUC) foi superior. Tendo sido, dentre os 12 conjuntos de treinamento, a maior para esses cenários nos testes com CNN.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	52,44%	60,17%	78,25%	86,64%
H.10.x	**53,80%**	**63,53%**	**80,96%**	**87,38%**
S.0.x	50,96%	53,62%	68,92%	84,03%
S.7.x	51,22%	55,18%	73,30%	85,76%
S.10.x	51,10%	54,58%	71,56%	85,07%
S.12.x	51,10%	54,36%	70,51%	84,85%
M.-.x	51,11%	55,27%	71,93%	84,11%

Tabela 16 – Acurácias dos testes da CNN com o conjunto de treinamento H2.

4.2.3 Conjunto de Treinamento H4

Entre todos os conjuntos, este foi o que apresentou as melhores acurácias nos testes com os cenários H.0.4, H.10.4, H.0.6 e H.10.6, isto é, com o algoritmo HUGO e *payloads* de 0.4 e 0.6 bpp. Também conseguindo bons resultados nos outros algoritmos com esses *payloads*, como pode ser visto na Tabela 17.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	52,27%	60,82%	80,78%	88,01%
H.10.x	**53,29%**	**64,23%**	**82,63%**	**88,56%**
S.0.x	50,76%	53,80%	73,99%	86,73%
S.7.x	51,18%	55,69%	78,30%	87,80%
S.10.x	51,07%	54,80%	77,04%	87,33%
S.12.x	50,98%	54,51%	75,78%	87,23%
M.-.x	50,96%	55,16%	75,15%	85,50%

Tabela 17 – Acurácias dos testes da CNN com o conjunto de treinamento H4.

4.2.4 Conjunto de Treinamento H6

A Tabela 18 ilustra os resultados do conjunto de treinamento H6, que foi um dos piores nos testes realizados, tendo atingido acurácia de apenas 51.03% para *payload* de 0.1 bpp. Contudo a AUC para esses casos ficou entre as maiores obtidas, ficando acima de 0,5648.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	50,77%	56,26%	73,51%	85,50%
H.10.x	**51,03%**	**58,68%**	**76,02%**	**87,11%**
S.0.x	50,32%	51,87%	69,10%	83,14%
S.7.x	50,44%	53,21%	72,62%	86,16%
S.10.x	50,38%	52,74%	71,38%	84,67%
S.12.x	50,38%	52,48%	70,58%	84,43%
M.-.x	50,39%	53,06%	70,28%	82,14%

Tabela 18 – Acurácias dos testes da CNN com o conjunto de treinamento H6.

4.2.5 Conjunto de Treinamento S1

A Tabela 19 mostra as acurácias da rede treinada com o conjunto S1, ou seja, o S-UNIWARD com *payload* de 0.1 bpp. Embora tenha atingido taxas de acurácia superiores a 82% para o *payload* 0.6, este conjunto apresentou resultados insatisfatórios

para os demais *payloads*. Além disso esse conjunto atingiu os piores resultados ao se analisar a AUC.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	51,24%	53,92%	73,93%	84,99%
H.10.x	**51,52%**	**55,1%**	**76,7%**	**86,02%**
S.0.x	50,64%	52,59%	65,49%	83,57%
S.7.x	50,94%	53,32%	71,94%	85,43%
S.10.x	50,92%	52,96%	69,07%	84,71%
S.12.x	50,77%	52,8%	67,94%	84,47%
M.-.x	50,78%	52,64%	67,59%	82,96%

Tabela 19 – Acurácias dos testes da CNN com o conjunto de treinamento S1.

4.2.6 Conjunto de Treinamento S2

Dentre todos os conjuntos de treinamentos, esse foi o que apresentou os melhores resultados na detecção de todos os algoritmos com *payloads* de 0.1 e 0.2 bpp. Porém, obteve os piores resultados para *payload* de 0.6 bpp, não alcançando 75% de acurácia, apesar disso as AUCs nos testes com esse *payload* foi muito similar as obtidas com o conjunto de treinamento H4, um dos que teve as maiores acurácias. A Figura 18 ilustra essa situação com o cenário de teste H.10.6.

Essa disparidade entre as métricas avaliadas pode ser explicada ao se analisar as matrizes de confusão[2] dos testes. A Tabela 20 compara as matrizes de confusão, acurácia, precisão[3] e revocação[4] obtidas pelas CNNs treinadas com os conjuntos S2 e H4 quando testados no cenário H.10.6.

	Matriz de Confusão	**Acurácia**	**Precisão**	**Revocação**
H4	$\begin{bmatrix} 4156 & 844 \\ 300 & 4700 \end{bmatrix}$	88,56%	0.8477	0.94
S2	$\begin{bmatrix} 2548 & 2452 \\ 62 & 4938 \end{bmatrix}$	74,86%	0.6682	0.9876

Tabela 20 – Outras métricas dos testes realizados no cenário H.10.6 com as CNNs treinadas com os conjuntos H4 e S2.

[2]Matriz de confusão é uma tabela que mostra os resultados obtidos por um classificador. As linhas representam os rótulos reais, e as colunas o predito pelo classificador.

[3]Proporção de verdadeiros positivos em relação a todas predições positivas.

[4]Taxa de verdadeiros positivos.

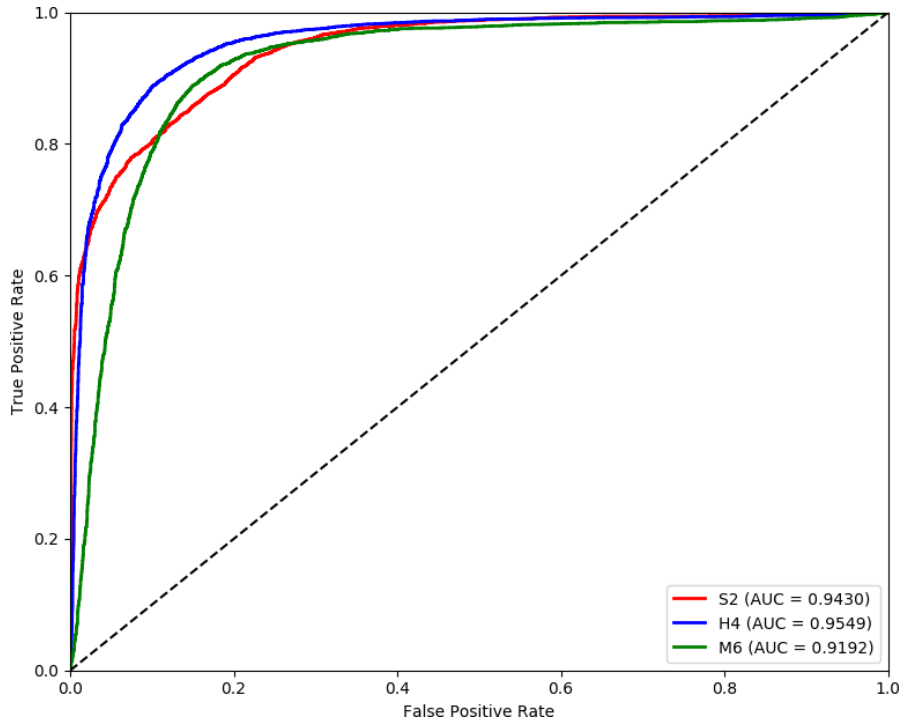


Figura 18 – Curvas ROC dos testes com o cenário H.10.6 para as CNNs treinadas com os conjuntos S2, H4 e M6 (Autoria própria).

Também é interessante notar a baixa variação das acurácias para o *payload* de 0.6 bpp, onde se tem um desvio padrão de apenas 0,093. A Tabela 21 ilustra os resultados obtidos ao utilizar diferentes cenários de teste.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	56,1%	67,59%	73,59%	74,76%
H.10.x	**57,94%**	**69,02%**	**73,87%**	**74,86%**
S.0.x	53,65%	63,21%	73,09%	74,73%
S.7.x	55,03%	66,78%	73,65%	74,83%
S.10.x	54,69%	65,59%	73,47%	74,76%
S.12.x	54,3%	65,36%	73,45%	74,79%
M.-.x	54,18%	64,17%	72,88%	74,55%

Tabela 21 – Acurácias dos testes da CNN com o conjunto de treinamento S2.

4.2.7 Conjunto de Treinamento S4

Os testes com a CNN treinada neste conjunto apresentaram bons resultados, ficando entre os melhores para *payloads* de 0.1, 0.2 e 0.4 bpp para todos algoritmos. A Tabela 22 apresenta as acurácias para todos os cenários de teste.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	53,78%	66,74%	80,25%	84,43%
H.10.x	**55,21%**	**69,04%**	**81,07%**	84,50%
S.0.x	52,28%	59,95%	79,15%	84,56%
S.7.x	53,08%	64,30%	80,94%	**84,89%**
S.10.x	52,82%	62,80%	80,43%	84,77%
S.12.x	52,64%	62,39%	80,00%	84,65%
M.-.x	52,58%	61,82%	78,03%	83,30%

Tabela 22 – Acurácias dos testes da CNN com o conjunto de treinamento S4.

4.2.8 Conjunto de Treinamento S6

Este foi o conjunto de treinamento que obteve a maior acurácia para todos os cenários de teste do algoritmo S-UNIWARD com *payload* de 0.6 bpp. Tendo mostrado também boa acurácia para outros algoritmos com o mesmo *payload*. Porém foi um dos piores nos testes com *payload* de 0.1 bpp, como pode ser visto na Tabela 23.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	51,12%	59,21%	75,72%	87,01%
H.10.x	**51,67%**	**61,38%**	77,58%	87,86%
S.0.x	50,68%	55,24%	74,52%	87,24%
S.7.x	50,98%	58,51%	**77,67%**	**89,12%**
S.10.x	50,72%	57,16%	76,50%	88,37%
S.12.x	50,81%	56,80%	75,89%	88,12%
M.-.x	50,73%	55,52%	72,75%	83,85%

Tabela 23 – Acurácias dos testes da CNN com o conjunto de treinamento S6.

4.2.9 Conjunto de Treinamento M1

Este conjunto, treinado com a base contendo imagens esteganografadas com o algoritmo MiPOD, apresentou resultados medianos em todos os testes. A Tabela 24 exibe as acurácias obtidas.

4.2.10 Conjunto de Treinamento M2

Assim como o conjunto S4, o M2 também ficou entre os melhores nos testes com *payloads* de 0.1, 0.2, e 0.4 bpp, porém foi um dos piores nos testes com 0.6. A Tabela 25 apresenta os resultados obtidos. Assim como na maioria dos testes, cenário H.10.x foi o que apresentou as maiores taxas de acurácia.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	52,87%	61,84%	77,76%	83,15%
H.10.x	**53,80%**	**64,64%**	**79,00%**	**83,56%**
S.0.x	51,40%	55,88%	74,84%	82,09%
S.7.x	51,98%	58,11%	77,29%	83,08%
S.10.x	51,72%	57,42%	76,28%	82,85%
S.12.x	51,65%	57,07%	76,02%	82,63%
M.-.x	51,98%	58,18%	76,14%	82,20%

Tabela 24 – Acurácias dos testes da CNN com o conjunto de treinamento M1.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	53,80%	63,09%	78,96%	82,46%
H.10.x	**54,94%**	**65,94%**	**79,93%**	**82,59%**
S.0.x	52,25%	57,86%	75,74%	81,92%
S.7.x	52,92%	60,29%	78,40%	82,46%
S.10.x	52,72%	59,49%	77,58%	82,27%
S.12.x	52,64%	58,94%	76,90%	82,10%
M.-.x	52,94%	60,59%	77,46%	82,00%

Tabela 25 – Acurácias dos testes da CNN com o conjunto de treinamento M2.

4.2.11 Conjunto de Treinamento M4

A Tabela 26 apresenta os resultados obtidos com este conjunto de treinamento, que teve boas taxas de acerto nos testes com *payloads* de 0.4 e 0.6 bpp. Foi o melhor desempenho obtido em todos os testes para o cenário M.-.6, onde alcançou 85.94% de acurácia, superando inclusive a abordagem com SRM + *ensemble of classifiers*.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	51,61%	57,49%	79,67%	86,44%
H.10.x	**52,17%**	**60,59%**	**81,55%**	**87,05%**
S.0.x	51,09%	54,10%	73,14%	85,51%
S.7.x	51,36%	55,54%	78,52%	86,48%
S.10.x	51,28%	54,83%	76,12%	86,25%
S.12.x	51,24%	54,73%	75,43%	86,08%
M.-.x	51,57%	55,77%	77,14%	85,94%

Tabela 26 – Acurácias dos testes da CNN com o conjunto de treinamento M4.

Observe que o desempenho para os *payloads* 0.1 e 0.2 não foi bom, indicando que o treinamento para o *payload* 0.4 deixa o classificador mais habilitado para identificar maiores quantidades de mensagens escondidas. Ao passo que o treinamento realizado

com M1 e M2 obtém 64,64% e 65,94% de acurácia para o teste com *payload* 0.2, respectivamente, para o M4 o valor é de 60,59%.

4.2.12 Conjunto de Treinamento M6

Os resultados do conjunto M6, mostrados na Tabela 27, foram os piores em comparação com os obtidos pelos outros conjuntos em grande parte dos testes. Apresentando resultados satisfatórios apenas quando testado com *payload* de 0.6 bpp.

	x = 0.1	**x = 0.2**	**x = 0.4**	**x = 0.6**
H.0.x	50,52%	51,86%	65,65%	84,16%
H.10.x	**50,63%**	**52,29%**	**69,96%**	**86,12%**
S.0.x	50,37%	51,35%	57,59%	80,20%
S.7.x	50,45%	51,63%	61,46%	84,46%
S.10.x	50,44%	51,59%	59,65%	82,80%
S.12.x	50,39%	51,60%	59,30%	82,28%
M.-.x	50,48%	51,48%	63,61%	83,87%

Tabela 27 – Acurácias dos testes da CNN com o conjunto de treinamento M6.

4.3 Análise dos Resultados

Os experimentos realizados com o uso de SRM e *ensemble* de classificadores revelaram que essa abordagem é inadequada para esteganálise cega, isto é, detectar a presença de uma mensagem oculta sem conhecimento prévio do algoritmo e *payload* utilizados. Os resultados obtidos só foram bons quando o treinamento e teste foi feito com o mesmo algoritmo, e com uma taxa de bits por pixel igual ou parecida.

Já a CNN, apesar de não ter alcançado os mesmos valores de acurácia que o *ensemble* de classificadores em esteganálise direcionada, se mostrou muito mais eficaz para esteganálise cega, especialmente quando treinada com *payloads* de 0.2 e 0.4 bpp. As únicas exceções a isso são as CNNs treinadas com MiPOD e S-UNIWARD com 0.2 bpp quando testadas com *payload* de 0.6 bpp, onde a acurácia obtida foi muito baixa.

Uma das primeiras hipóteses consideradas no início desse trabalho era que os kernels da CNN poderiam se aproximar dos filtros SRM. Porém, isso não pode ser confirmado com os experimentos, uma vez que os dois detectores se comportam de forma diferente — a CNN tendo poder de generalização maior enquanto o SRM se mostra mais especializado.

Outro comportamento pertinente ao se usar a CNN foi a obtenção de piores resultados ao se usar os *payloads* de 0.1 bpp e 0.6 bpp para o treinamento. Isso

provavelmente ocorre porque quanto mais extremo o *payload*, mais especializada tende a ficar a rede.

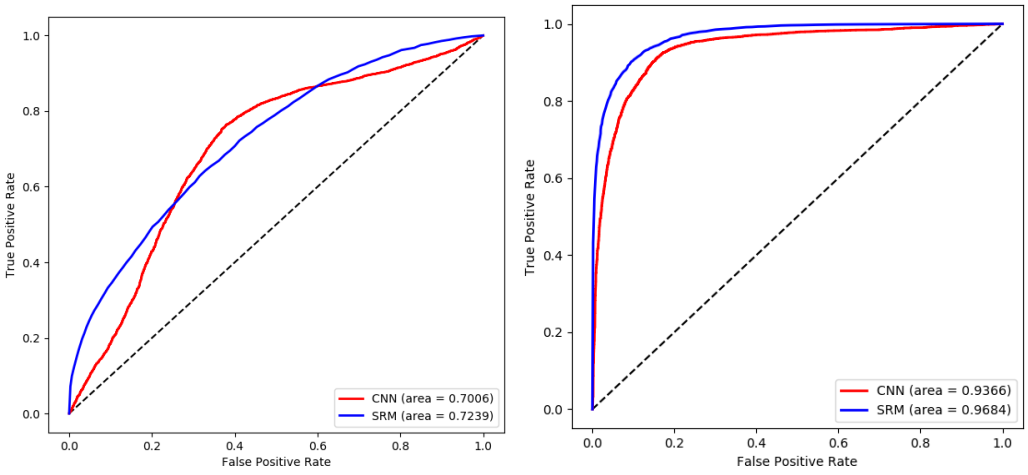
É possível notar também que, nos cenários onde foi utilizado o STC, a acurácia dos detectores foi maior, tanto do SRM e *ensemble* de classificadores quanto da CNN.

Como já esperado, o algoritmo com o uso do descritor SRM teve desempenho notável quando se testando com o algoritmo HUGO, principalmente pelo fato de ter sido estruturado para a esteganálise do mesmo. O que acabou sendo surpreendente é que o seu melhor desempenho é muito superior ao da CNN, diferença discrepante com relação ao resultados dos outros algoritmos. Como pode ser observado na Tabela 28, nos cenários H.0.1 e H.10.1 a diferença chegou a um máximo de 10%.

	H.0.1	H.0.2	H.0.4	H.0.6	H.10.1	H.10.2	H.10.4	H.10.6
SRM	66,23%	73,89%	84,60%	90,46%	67,02%	75,30%	85,43%	91,18%
CNN	56,10%	67,59%	80,25%	88,01%	57,94%	69,04%	81,55%	88,56%

Tabela 28 – Comparação entre os melhores resultados para o algoritmo HUGO com uso do SRM e com uso da CNN

Apesar das acurácias diferirem bastante, principalmente para o algoritmo HUGO, as AUCs para o *payload* de 0.1 bpp não tem um contraste tão evidente, como mostrado na Figura 19(a), na qual a CNN chegou, inclusive, a ter a curva superior ao SRM em vários pontos. Esse mesmo comportamento não se repete para o *payload* de 0.6 bpp, como pode ser visto na Figura 19(b), onde a curva do SRM é sempre superior a da CNN.



(a) HUGO 0.1 (b) HUGO 0.6

Figura 19 – Curvas ROC da CNN e do SRM para os cenários de teste e treino iguais (Autoria própria)

A Figura 20 ilustra um conjunto de treino no qual o SRM teve desempenho inferior à CNN em todos os cenários de teste que fazem uso do MiPOD, tanto com relação as acurácias quanto as AUCs. Isso ocorre porque o treino foi feito com um algoritmo de menor complexidade que o de teste, situação que piora consideravelmente a performance do SRM.

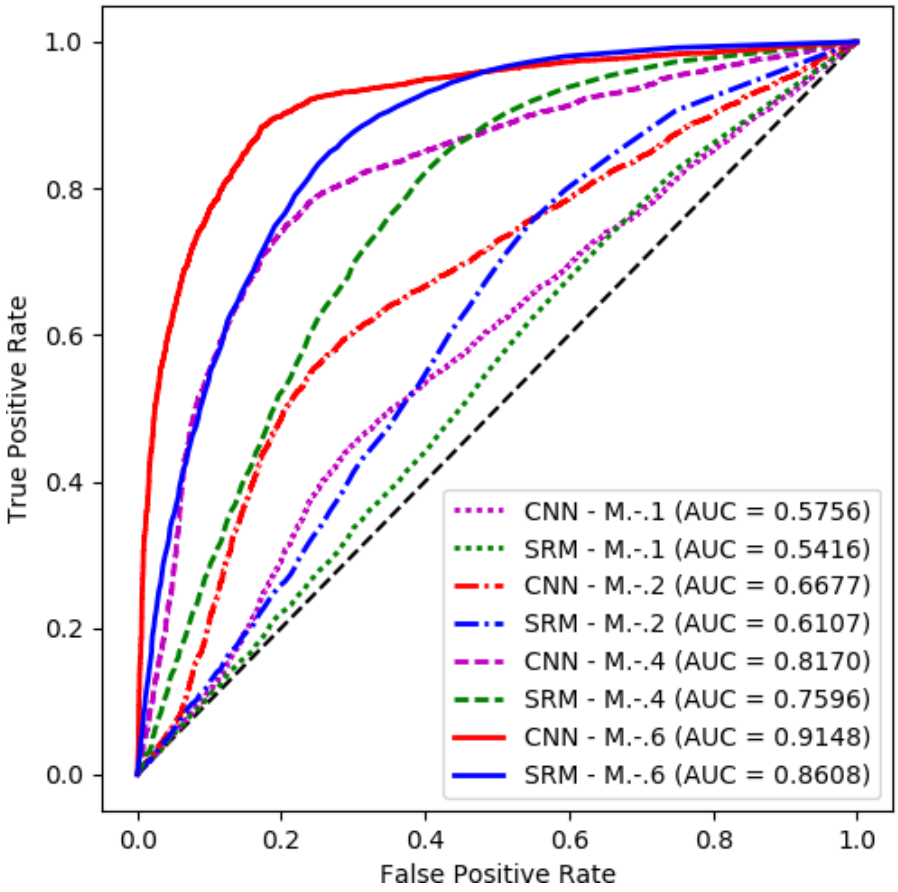


Figura 20 – Curvas ROC dos testes com o MiPOD e treinos com o cenário H6. (Autoria própria).

Uma característica em comum às duas arquiteturas foi a maior dificuldade de detectar a presença do algoritmo MiPOD. No caso do SRM, isso é ainda mais notável quando o classificador é treinado com um algoritmo diferente do MiPOD.

Com relação ao desempenho computacional, foi possível notar que o tempo de treino do *ensemble* de classificadores com FLD é muito menor que o da CNN, porém o descritor SRM tem alta complexidade computacional pela quantidade de convoluções que são realizadas para cada filtro do algoritmo. Mesmo com a alta complexidade do descritor, a CNN é mais custosa computacionalmente no processo como um todo.

Os Apêndices A e B apresentam os valores de AUC para as duas abordagens.

5 Conclusão

Uma das contribuições importantes do trabalho é ser uma literatura moderna sobre esteganografia e esteganálise, visto que há certa defasagem desses assuntos na literatura em português. Esses temas tem ganhado notoriedade na comunidade científica e são áreas de estudo complexas e multidisciplinares que tocam, por exemplo, as áreas de Processamento de Sinais, Segurança da Informação, Teoria da Informação e Teoria dos Códigos. Além disso, também foi utilizada uma abordagem de esteganálise com *deep learning* que é um dos métodos que estão sendo investigados atualmente por estegoanalistas.

Nos resultados, a utilização dos descritores SRM em um *ensemble* de classificadores FLD obteve resultados melhores quando o treinamento e teste foram feitos no mesmo cenário, ou seja, quando o classificador foi treinado e testado em estego imagens geradas pelo mesmo algoritmo de esteganografia e com o mesmo *payload*.

A CNN se comportou de maneira distinta, tendo resultados semelhantes mesmo quando testada em cenários diferentes. Ela generaliza os casos de teste de tal forma que funciona similarmente em diversos cenários, enquanto a abordagem utilizando SRM é mais especializada e funciona melhor para um cenário.

Estes resultados sugerem que a CNN é mais apropriada para a utilização em ambientes de esteganálise cega e o *ensemble* de classificadores se comporta melhor em ambientes de esteganálise direcionada. Essa diferença de resultados também indica que a CNN não está aprendendo algo semelhante aos filtros do SRM.

Essas observações acerca da natureza das duas abordagens foi possível devido a uma metodologia onde, para cada conjunto de treinamento (que consideraram a combinação de diferentes algoritmos de esteganografia e *payloads*), foram utilizados dois classificadores: CNN e *ensemble* de classificadores (FLD como base). Eles foram aplicados para classificar diversos cenários de teste gerados nesse trabalho. Assim, podemos analisar a diferença das duas abordagens em diversos contextos de esteganálise.

Abordagens de *deep learning* estão sendo aplicadas com sucesso em diversos problemas difíceis de Processamento de Sinais. Em esteganálise não é diferente, com bons resultados recentes utilizando CNNs (TAN; LI, 2014; QIAN et al., 2015; XU; WU; SHI, 2016; XU; WU; SHI, 2016). Portanto, o estudo de esteganálise com CNN é bastante promissor, ainda tendo lacunas de possíveis metodologias, arquiteturas e camadas relacionadas a otimização da rede especificamente para problemas de esteganálise há serem exploradas.

5.1 Trabalhos Futuros

Os resultados desse trabalho mostraram as diferenças de generalização e especificação dos detectores de esteganografia. Uma continuação desse trabalho, utilizando a mesma metodologia poderia ser a criação de um *Ensamble of CNNs*, com o fim de aumentar a acurácia sem perder a generalização. Além dessa mudança da forma em que as CNNs são utilizadas, é possível também alterar a estrutura interna de cada uma das CNNs.

Há diferentes e novas estruturas de CNN que estão sendo investigadas para resolver diferentes problemas. Alguns exemplos são as *Recurrent Neural Networks*, *Residual Neural Networks*, *Region Based CNN* entre outras. Essas arquiteturas podem ser exploradas com a finalidade de solucionar problemas de esteganálise, como o trabalho de Wu, Zhong e Liu (2017) onde é utilizado uma *Residual Neural Network* com ≈ 60 camadas. Algumas dessas arquiteturas favorecem intuitivamente alguns problemas da esteganálise forense como, por exemplo, uma *Region Based CNN* poderia ser treinada para descobrir a localização dos dados embutidos e, com isso, obter dicas do algoritmo que foi utilizado. *Recurrent Neural Networks* criam relações entre as camadas o que poderia ser utilizado para a estimativa de *payloads*.

Como o SRM para esteganálise direcionada obteve mais sucesso nos testes realizados, os kernels da CNN poderiam ser inicializados com os filtros lineares desse descritor (o que seria uma forma de explorar o comportamento desses detectores). Também, trabalhos como os de Denemark et al. (2014) e Denemark, Fridrich e Comesaña-Alfaro (2016) buscam aprimoramentos, alterações e acréscimos nos descritores SRM. Esses trabalhos podem também ser explorados em conjunto com CNNs.

Sedighi, Cogranne e Fridrich (2016) apresentaram uma definição rigorosa de modelos de imagens de cobertura e estego imagens, conseguindo chegar em uma fórmula fechada para um detector ótimo de esteganografia que elevou o compreendimento sobre os limites da esteganálise. Esses resultados podem ser utilizados para a criação de novos métodos e arquiteturas de CNNs para esteganálise.

Recentemente, Sedighi e Fridrich (2017) publicaram resultados sobre a implementação de uma nova camada para CNN chamada de *Histogram Layer*, criada especificamente para melhorar resultados em esteganálise. Apesar do estado da arte não ser atingido, esses resultados foram importantes como prova de conceito de uma rede com informações e métodos esteganográficos previamente conhecidos, deixando a dúvida da existência de outros métodos similares misturando conhecimentos de esteganálise com aprendizado das CNNs.

Referências

SEDIGHI, V.; COGRANNE, R.; FRIDRICH, J. Content-adaptive steganography by minimizing statistical detectability. **IEEE Transactions on Information Forensics and Security**, IEEE, v. 11, n. 2, p. 221–234, 2016.

BAS, P.; FILLER, T.; PEVNỲ, T. "break our steganographic system": The ins and outs of organizing boss. In: SPRINGER. **International Workshop on Information Hiding**. [S.l.], 2011. p. 59–70.

PEVNY, T.; BAS, P.; FRIDRICH, J. Steganalysis by subtractive pixel adjacency matrix. **IEEE Transactions on information Forensics and Security**, IEEE, v. 5, n. 2, p. 215–224, 2010.

FRIDRICH, J.; KODOVSKY, J. Rich models for steganalysis of digital images. **IEEE Transactions on Information Forensics and Security**, IEEE, v. 7, n. 3, p. 868–882, 2012.

MCCULLOCH, W. S.; PITTS, W. H. A logical calculus of the ideas immanent in nervous activity. **Bulletin of Mathematical Biophysics**, v. 5, p. 115–133, 1943.

RUSSELL, S. J.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. Third. [S.l.]: Prentice Hall, 2009. (Prentice Hall series in artificial intelligence).

LAWRENCE, S. et al. Face recognition: A convolutional neural-network approach. **IEEE transactions on neural networks**, IEEE, v. 8, n. 1, p. 98–113, 1997.

XU, G.; WU, H.-Z.; SHI, Y.-Q. Structural design of convolutional neural networks for steganalysis. **IEEE Signal Processing Letters**, IEEE, v. 23, n. 5, p. 708–712, 2016.

FRIDRICH, J. **Steganography in Digital Media: Principles, Algorithms, and Applications**. 1st. ed. New York, NY, USA: Cambridge University Press, 2009. ISBN 0521190193, 9780521190190.

TAN, S.; LI, B. Stacked convolutional auto-encoders for steganalysis of digital images. In: IEEE. **Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA)**. [S.l.], 2014. p. 1–4.

QIAN, Y. et al. Deep learning for steganalysis via convolutional neural networks. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. **SPIE/IS&T Electronic Imaging**. [S.l.], 2015. p. 94090J–94090J.

XU, G.; WU, H.-Z.; SHI, Y. Q. Ensemble of cnns for steganalysis: an empirical study. In: ACM. **Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security**. [S.l.], 2016. p. 103–107.

WU, S.; ZHONG, S.; LIU, Y. Deep residual learning for image steganalysis. **Multimedia Tools and Applications**, Springer, p. 1–17, 2017.

PEVNỲ, T.; FILLER, T.; BAS, P. Using high-dimensional image models to perform highly undetectable steganography. p. 161–177, 2010.

HOLUB, V.; FRIDRICH, J.; DENEMARK, T. Universal distortion function for steganography in an arbitrary domain. **EURASIP Journal on Information Security**, Springer, v. 2014, n. 1, p. 1, 2014.

PETITCOLAS, F. A.; ANDERSON, R. J.; KUHN, M. G. Information hiding-a survey. **Proceedings of the IEEE**, IEEE, v. 87, n. 7, p. 1062–1078, 1999.

BROWN, C. W. **Graphics file formats: reference and guide**. [S.l.]: Manning Publications, 1995. ISBN 1884777007.

KER, A. D. Improved detection of lsb steganography in grayscale images. **Lecture Notes in Computer Science**, Springer, v. 3200, 2014.

MIELIKAINEN, J. Lsb matching revisited. **IEEE signal processing letters**, IEEE, v. 13, n. 5, p. 285–287, 2006.

FRIDRICH, J.; FILLER, T. Practical methods for minimizing embedding impact in steganography. **SPIE 6505, Security, Steganography, and Watermarking of Multimedia Contents IX**, v. 6505, 2 2007.

FILLER, T.; JUDAS, J.; FRIDRICH, J. Minimizing embedding impact in steganography using trellis-coded quantization. **Media Forensics and Security II**, v. 7541, 1 2010.

CACHIN, C. An information-theoretic model for steganography. **Information and Computation**, Elsevier, v. 192, n. 1, p. 41–56, 2004.

ANDERSON, R. J. Stretching the limits of steganography. Springer-Verlag, London, UK, UK, p. 39–48, 1996. Disponível em: <http://dl.acm.org/citation.cfm?id=647594.731520>.

ADAMS, P. Ying wang and pierre moulin. **IEEE Transactions on Information Theory**, v. 54, 6 2008.

RYABKO, B. Y.; RYABKO, D. B. Asymptotically optimal perfect steganographic systems. **Probl. Inf. Transm.**, Plenum Press, New York, NY, USA, v. 45, n. 2, p. 184–190, jun. 2009. ISSN 0032-9460. Disponível em: <http://dx.doi.org/10.1134/S0032946009020094>.

LI, B. et al. A survey on image steganography and steganalysis. **Journal of Information Hiding and Multimedia Signal Processing**, v. 2, n. 2, p. 142–172, 2011.

HOLUB, V.; FRIDRICH, J. Designing steganographic distortion using directional filters. In: IEEE. **Information Forensics and Security (WIFS), 2012 IEEE International Workshop on**. [S.l.], 2012. p. 234–239.

LI, B. et al. A new cost function for spatial image steganography. In: IEEE. **Image Processing (ICIP), 2014 IEEE International Conference on**. [S.l.], 2014. p. 4206–4210.

BÖHME, R. **Advanced statistical steganalysis**. [S.l.]: Springer Science & Business Media, 2010.

WAYNER, P. **Disappearing cryptography: information hiding: steganography & watermarking**. [S.l.]: Morgan Kaufmann, 2009.

FRIDRICH, J.; GOLJAN, M. Practical steganalysis of digital images: state of the art. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. **Electronic Imaging 2002**. [S.l.], 2002. p. 1–13.

WESTFELD, A.; PFITZMANN, A. Attacks on steganographic systems. In: SPRINGER. **Information Hiding**. [S.l.], 1999. p. 61–76.

LIPSCHUTZ, S.; SCHILLER, J. J. **Schaum's Outline of Introduction to Probability and Statistics**. [S.l.]: McGraw Hill Professional, 1998.

KODOVSKY, J.; FRIDRICH, J.; HOLUB, V. Ensemble classifiers for steganalysis of digital media. **IEEE Transactions on Information Forensics and Security**, IEEE, v. 7, n. 2, p. 432–444, 2012.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, Nature Research, v. 521, n. 7553, p. 436–444, 2015.

LECUN, Y. et al. Handwritten digit recognition with a back-propagation network. In: **Advances in neural information processing systems**. [S.l.: s.n.], 1990. p. 396–404.

LECUN, Y. et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, IEEE, v. 86, n. 11, p. 2278–2324, 1998.

CHEN, H. et al. Standard plane localization in fetal ultrasound via domain transferred deep neural networks. **IEEE journal of biomedical and health informatics**, IEEE, v. 19, n. 5, p. 1627–1636, 2015.

LU, H. et al. Wound intensity correction and segmentation with convolutional neural networks. **Concurrency and Computation: Practice and Experience**, Wiley Online Library, 2016.

ZHONG, S.-H.; LIU, Y.; HUA, K. A. Field effect deep networks for image recognition with incomplete data. **ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)**, ACM, v. 12, n. 4, p. 52, 2016.

KODOVSKY, J.; FRIDRICH, J.; HOLUB, V. On dangers of overtraining steganography to incomplete cover model. In: ACM. **Proceedings of the thirteenth ACM multimedia workshop on Multimedia and security**. [S.l.], 2011. p. 69–76.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. **CoRR**, abs/1502.03167, 2015. Disponível em: <http://arxiv.org/abs/1502.03167>.

BISHOP, C. M. **Pattern recognition and machine learning**. [S.l.]: springer, 2006.

BREIMAN, L. Bagging predictors. **Machine Learning**, v. 24, n. 2, p. 123–140, Aug 1996. ISSN 1573-0565. Disponível em: <https://doi.org/10.1007/BF00058655>.

DENEMARK, T. et al. Selection-channel-aware rich model for steganalysis of digital images. In: IEEE. **Information Forensics and Security (WIFS), 2014 IEEE International Workshop on**. [S.l.], 2014. p. 48–53.

DENEMARK, T.; FRIDRICH, J.; COMESAÑA-ALFARO, P. Improving selection-channel-aware steganalysis features. **Electronic Imaging**, Society for Imaging Science and Technology, v. 2016, n. 8, p. 1–8, 2016.

SEDIGHI, V.; FRIDRICH, J. Histogram layer, moving convolutional neural networks towards feature-based steganalysis. **Electronic Imaging**, Society for Imaging Science and Technology, v. 2017, n. 7, p. 50–55, 2017.

Apêndices

APÊNDICE A – Tabela de AUCs do SRM

	H1	H2	H4	H6
H.0.1	0,7337	0,7075	0,6396	0,6034
H.10.1	0,7470	0,7269	0,6596	0,6192
H.0.2	0,8329	0,8352	0,7935	0,7525
H.10.2	0,8444	0,8483	0,8122	0,7718
H.0.4	0,9181	0,9322	0,9291	0,9147
H.10.4	0,9243	0,8391	0,9384	0,9273
H.0.6	0,9528	0,9660	0,9712	0,9693
H.10.6	0,9561	0,9690	0,9743	0,9736
S.0.1	0,6305	0,5847	0,5519	0,5388
S.10.1	0,6451	0,6014	0,5629	0,5481
S.0.2	0,7469	0,7160	0,6528	0,6237
S.0.4	0,8653	0,8673	0,8349	0,7967
S.10.4	0,8749	0,8801	0,8524	0,8158
S.0.6	0,9259	0,9347	0,9268	0,8995
M.0.1	0,6212	0,5800	0,5476	0,5363
M.0.2	0,7170	0,6893	0,6335	0,6061
M.0.4	0,8230	0,8275	0,7962	0,7599
M.0.6	0,8838	0,8998	0,8913	0,8651

Tabela 29 – AUCs do *Ensemble of Classifiers* treinado com o descritor SRM para o algoritmo HUGO

APÊNDICE B – Tabelas de AUCs da CNN

	H1	H2	H4	H6
H.0.1	0,6119	0,6162	0,6014	0,6029
H.10.1	0,6337	0,6402	0,6234	0,6200
H.0.2	0,7006	0,7173	0,7179	0,7056
H.10.2	0,7273	0,7466	0,7473	0,7275
H.0.4	0,8304	0,8621	0,8788	0,8542
H.10.4	0,8538	0,8837	0,8967	0,8727
H.0.6	0,9085	0,9367	0,9478	0,9366
H.10.6	0,9207	0,9459	0,9549	0,9456
S.0.1	0,5609	0,5589	0,5558	0,5648
S.7.1	0,5758	0,5744	0,5705	0,5808
S.10.1	0,5717	0,5700	0,5667	0,5765
S.12.1	0,5692	0,5672	0,5636	0,5731
S.0.2	0,6331	0,6372	0,6402	0,6560
S.7.2	0,6554	0,6638	0,6706	0,6829
S.10.2	0,6480	0,6553	0,6604	0,6737
S.12.2	0,6449	0,6508	0,6559	0,6703
S.0.4	0,7583	0,7926	0,8268	0,8183
S.7.4	0,7850	0,8243	0,8580	0,8450
S.10.4	0,7751	0,8125	0,8470	0,8353
S.12.4	0,7698	0,8059	0,8402	0,8298
S.0.6	0,8586	0,9043	0,9301	0,9194
S.7.6	0,8820	0,9239	0,9436	0,9354
S.10.6	0,8728	0,9165	0,9388	0,9293
S.12.6	0,8694	0,9133	0,9360	0,9263
M.-.1	0,5702	0,5690	0,5658	0,5756
M.-.2	0,6487	0,6564	0,6601	0,6677
M.-.4	0,7799	0,8128	0,8338	0,8170
M.-.6	0,8759	0,9109	0,9251	0,9148

Tabela 30 – AUCs das CNNs treinadas com HUGO

	S1	**S2**	**S4**	**S6**
H.0.1	0,5640	0,5854	0,6054	0,5793
H.10.1	0,5770	0,6023	0,6228	0,5941
H.0.2	0,6421	0,7034	0,7148	0,6869
H.10.2	0,6629	0,7306	0,7346	0,7061
H.0.4	0,7970	0,8759	0,8729	0,8428
H.10.4	0,8181	0,8900	0,8879	0,8573
H.0.6	0,8875	0,9362	0,9489	0,9336
H.10.6	0,8964	0,9430	0,9543	0,9414
S.0.1	0,5435	0,5593	0,5741	0,5549
S.7.1	0,5534	0,5752	0,5930	0,5703
S.10.1	0,5503	0,5707	0,5877	0,5657
S.12.1	0,5488	0,5681	0,5843	0,5631
S.0.2	0,6062	0,6552	0,6782	0,6552
S.7.2	0,6273	0,6885	0,7055	0,6845
S.10.2	0,6203	0,6771	0,6964	0,6752
S.12.2	0,6174	0,6723	0,6931	0,6710
S.0.4	0,7559	0,8528	0,8521	0,8322
S.7.4	0,7904	0,8779	0,8810	0,8579
S.10.4	0,7778	0,8688	0,8697	0,8477
S.12.4	0,7711	0,8641	0,8643	0,8427
S.0.6	0,8746	0,9290	0,9465	0,9344
S.7.6	0,8904	0,9393	0,9561	0,9489
S.10.6	0,8852	0,9357	0,9526	0,9434
S.12.6	0,8820	0,9338	0,9507	0,9404
M.-.1	0,5482	0,5653	0,5821	0,5617
M.-.2	0,6141	0,6627	0,6830	0,6563
M.-.4	0,7620	0,8504	0,8408	0,8117
M.-.6	0,8686	0,9242	0,9326	0,9102

Tabela 31 – AUCs das CNNs treinadas com S-UNIWARD

	M1	**M2**	**M4**	**M6**
H.0.1	0,5828	0,5867	0,5741	0,5738
H.10.1	0,5980	0,6025	0,5896	0,5872
H.0.2	0,6660	0,6774	0,6725	0,6610
H.10.2	0,6864	0,7010	0,7009	0,6823
H.0.4	0,7927	0,8273	0,8557	0,8129
H.10.4	0,8129	0,8479	0,8767	0,8334
H.0.6	0,8853	0,9183	0,9394	0,9062
H.10.6	0,9000	0,9298	0,9482	0,9192
S.0.1	0,5537	0,5583	0,5493	0,5514
S.7.1	0,5670	0,5723	0,5618	0,5642
S.10.1	0,5633	0,5683	0,5584	0,5607
S.12.1	0,5609	0,5661	0,5562	0,5587
S.0.2	0,6240	0,6327	0,6231	0,6226
S.7.2	0,6461	0,6566	0,6490	0,6459
S.10.2	0,6390	0,6489	0,6403	0,6384
S.12.2	0,6358	0,6451	0,6368	0,6353
S.0.4	0,7521	0,7803	0,8053	0,7711
S.7.4	0,7780	0,8098	0,8429	0,8011
S.10.4	0,7684	0,7986	0,8288	0,7898
S.12.4	0,7633	0,7930	0,8214	0,7838
S.0.6	0,8513	0,8897	0,9236	0,8830
S.7.6	0,8753	0,9116	0,9396	0,9045
S.10.6	0,8662	0,9035	0,9336	0,8968
S.12.6	0,8619	0,8992	0,9306	0,8922
M.-.1	0,5674	0,5711	0,5619	0,5643
M.-.2	0,6465	0,6562	0,6504	0,6468
M.-.4	0,7763	0,8097	0,8378	0,8021
M.-.6	0,8743	0,9094	0,9342	0,9042

Tabela 32 – AUCs das CNNs treinadas com MiPOD