



APLICAÇÃO DE FUNÇÕES DE PERDA EM MODELOS DE VERIFICAÇÃO DE ASSINATURAS

Trabalho de Conclusão de Curso

Engenharia da Computação

Diego Mendes da Silva

Orientador: Byron Leite Dantas Bezerra

Coorientador: Celso Antônio Marcionilo Lopes Junior



**Universidade de Pernambuco
Escola Politécnica de Pernambuco
Graduação em Engenharia de Computação**

DIEGO MENDES DA SILVA

**APLICAÇÃO DE FUNÇÕES DE PERDA
EM MODELOS DE VERIFICAÇÃO DE
ASSINATURAS**

Monografia apresentada como requisito parcial para obtenção do diploma de Bacharel em Engenharia de Computação pela Escola Politécnica de Pernambuco – Universidade de Pernambuco.

Orientador: Byron Leite Dantas Bezerra
Coorientador: Celso Antônio Marcionilo Lopes Junior

**Recife
outubro de 2022**

Silva, Diego Mendes da
Aplicação de Funções de Perda em Modelos de Verificação de
Assinaturas / Diego Mendes da Silva. - Recife - PE, 2022.
VIII, 31 f. : il. ; 29 cm.

Trabalho de Conclusão de Curso (Graduação em Engenharia de
Computação) Universidade de Pernambuco, Escola Politécnica de
Pernambuco, Recife, 2022

Orientador: Prof. Dr. Byron Leite Dantas Bezerra.

Inclui Referências.

1. Verificação de Assinaturas. 2. Redes Neurais Convolucionais.
3. Funções de Perda. I. Título. II. Bezerra, Byron Leite Dantas. III.
Universidade de Pernambuco.

MONOGRAFIA DE FINAL DE CURSO

Avaliação Final (para o presidente da banca)*

No dia 21/10/2022, às 11h00min, reuniu-se para deliberar sobre a defesa da monografia de conclusão de curso do(a) discente **DIEGO MENDES DA SILVA**, orientado(a) pelo(a) professor(a) **BYRON LEITE DANTAS BEZERRA**, sob título Aplicação Da Função Triplet-Loss Em Modelos De Verificação De Assinaturas, a banca composta pelos professores:

CARMELO JOSE ALBANEZ BASTOS FILHO (PRESIDENTE)

BYRON LEITE DANTAS BEZERRA (ORIENTADOR)

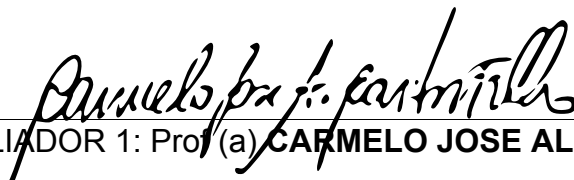
Após a apresentação da monografia e discussão entre os membros da Banca, a mesma foi considerada:

Aprovada Aprovada com Restrições* Reprovada

e foi-lhe atribuída nota: 10,0 (Dez)

*(Obrigatório o preenchimento do campo abaixo com comentários para o autor)

O(A) discente terá 7 dias para entrega da versão final da monografia a contar da data deste documento.


AVALIADOR 1: Prof (a) **CARMELO JOSE ALBANEZ BASTOS FILHO**


AVALIADOR 2: Prof (a) **BYRON LEITE DANTAS BEZERRA**

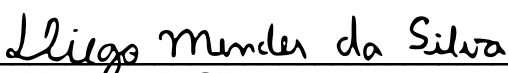
AVALIADOR 3: Prof (a)

* Este documento deverá ser encadernado juntamente com a monografia em versão final.

Autorização de publicação de PFC

Eu, **Diego Mendes da Silva** autor(a) do projeto de final de curso intitulado: **Aplicação Da Função Triplet-Loss Em Modelos De Verificação De Assinaturas**; autorizo a publicação de seu conteúdo na internet nos portais da Escola Politécnica de Pernambuco e Universidade de Pernambuco.

O conteúdo do projeto de final de curso é de responsabilidade do autor.



Diego Mendes da Silva



Orientador(a): **Byron Leite Dantas Bezerra**



Coorientador(a): **Celso Antonio Marcionilo Lopes Junior**



Prof. de TCC: **Daniel Augusto Ribeiro Chaves**

Data: 21/10/2022

Resumo

São inúmeras as áreas em que é preciso verificar ou identificar um usuário, e a assinatura manuscrita continua sendo uma biometria indispensável no mundo atual. A implementação de modelos de Redes Neurais Convolucionais(CNN) para a verificação de assinaturas faz parte do estado da arte, e mesmo com vários modelos publicados com ótimo desempenho, existem aspectos não muito analisados, como as funções de perda. Estas funções possuem um papel importante no treino de modelos CNN, calculando seu erro e permitindo a aprendizagem. Este estudo analisa três funções de perda(Cross-Entropy, Contrastive Loss e Triplet Loss), aplicadas em dois modelos CNN diferentes, em busca de entender melhor as características, vantagens e desvantagens de cada uma. Foi utilizada a base de dados GPDS Synthetic Signature Database, e os modelos foram implementados seguindo as especificações e hiperparâmetros de seus artigos originais. A função Cross-Entropy alcançou 18,62% de EER no primeiro modelo e 17,04% no segundo, se mostrando uma função consistente. A Contrastive Loss obteve 24,97% no primeiro modelo, e após vários testes com hiperparâmetros, alcançou 12,82% no segundo modelo e mostrou que pode obter bons resultados com ajustes-finos nos parâmetros. A função Triplet Loss alcançou 21,19% no primeiro modelo e 7,92% no segundo, a depender dos hiperparâmetros, sendo a melhor entre as funções testadas. Foi concluído que usar diferentes funções de perda pode ser benéfico, e o ideal é realizar testes no início da elaboração dos modelos a fim de decidir qual a melhor função de perda mais adequada ao projeto.

Palavras-Chave: Verificação de Assinaturas, Funções de Perda, Redes Neurais Convolucionais.

Abstract

There are countless areas in which it is necessary to verify or identify a user, and the handwritten signature remains an indispensable biometric in today's world. The implementation of Convolutional Neural Networks (CNN) models for signature verification is part of the state of the art, and even with several published models with excellent performance, there are aspects that are not much analyzed, such as loss functions. These functions play an important role in training CNN models, calculating their error and allowing learning. This study analyzes three loss functions (Cross-Entropy, Contrastive Loss and Triplet Loss), applied in two different CNN models, in order to better understand the characteristics, advantages and disadvantages of each one. The GPDS Synthetic Signature Database was used, and the models were implemented following the specifications and hyperparameters of their original articles. The Cross-Entropy function reached 18.62% of EER in the first model and 17.04% in the second, showing a consistent function. Contrastive Loss obtained 24.97% in the first model, and after several tests with hyperparameters, it reached 12.82% in the second model and showed that it can obtain good results with fine-tuning of the parameters. The Triplet Loss function reached 21.19% in the first model and 7.92% in the second, depending on the hyperparameters, being the best among the tested functions. It was concluded that using different loss functions can be beneficial, and the ideal is to carry out tests at the beginning of the elaboration of the models in order to decide which the best loss function is most suitable for the project.

Keywords: Signature Verification, Loss Functions, Convolutional Neural Networks.

Lista de ilustrações

Figura 1. Exemplos de duas assinaturas genuínas e uma forjada.	14
Figura 2. Exemplo de uma análise grafotécnica.	15
Figura 3. Exemplo de uma arquitetura CNN.	16
Figura 4. Comparação do <i>loss</i> e <i>acc</i> no modelo <i>SigNet-F</i> .	30
Figura 5. Comparação do <i>loss</i> e <i>acc</i> no modelo Siamesa.	32
Figura 6. Análise do <i>batch size</i> no modelo Siamesa, com a função <i>Cross-Entropy</i> .	33
Figura 7. Análise do <i>batch size</i> no modelo Siamesa, com a função <i>Contrastive Loss</i> .	34
Figura 8. Análise do <i>batch size</i> no modelo Siamesa, com a função <i>Triplet Loss</i> .	34

Lista de Tabelas

Tabela 1. Camadas do modelo CNN <i>SigNet-F</i> .	23
Tabela 2. Camadas do modelo CNN Siamesa.	25
Tabela 3. Hiperparâmetros diferentes entre as arquiteturas analisadas.	28
Tabela 4. Resultados dos experimentos com o <i>SigNet-F</i> .	31
Tabela 5. Resultados dos experimentos com a Siamesa.	32
Tabela 6. Análise da influência do <i>batch size</i> na taxa EER.	35

Lista de abreviaturas e siglas

acc. - accuracy (acurácia)

cnn - convolutional neural network (rede neural convolucional)

eer - equal error rate (taxa de erro igual)

gpds - grupo de processado digital de señales (grupo de processamento digital de sinais)

norm. - normalization (normalização)

sgd - stochastic gradient descent (gradiente descendente estocástico)

svm - support vector machine (máquina de vetores de suporte)

Sumário

1	Introdução	10
1.1	Motivação	10
1.2	Justificativa	11
1.3	Objetivos	11
1.3.1	Objetivo Geral	11
1.3.2	Objetivos Específicos	11
1.4	Organização do TCC	12
2	Fundamentação Teórica	13
2.1	Verificação de Assinaturas	13
2.1.1	Análise Grafotécnica	13
2.2	Redes Neurais Convolucionais	15
2.3	Funções de Perda	17
2.3.1	Cross-Entropy	17
2.3.2	Contrastive Loss	18
2.3.3	Triplet Loss	19
2.4	Trabalhos Relacionados	20
3	Procedimentos e Métodos	22
3.1	Arquiteturas Analisadas	22
3.2	Funções de Perda Implementadas	25
3.3	Descrição da Base de Dados	26
3.4	Pré-Processamento dos Dados	27
3.5	Treinamento, Validação e Teste	28
4	Resultados e Discussão	30
4.1	SigNet-F	30
4.2	Siamesa	31
4.3	Discussão dos Resultados	33
5	Conclusão e Trabalhos Futuros	36
	Referências	38

Capítulo 1

Introdução

1.1 Motivação

Desde quando a sociedade passou a necessitar de comprovação da identidade das pessoas, a assinatura manuscrita foi utilizada para verificar a autenticidade. Ao contrário de senhas geradas a partir de caracteres facilmente replicáveis, a assinatura manuscrita se provou uma biometria confiável, única a cada indivíduo e de fácil utilização[1].

Atualmente são inúmeras as áreas e situações em que é necessário verificar a genuinidade da identidade de uma pessoa na sociedade[2], e mesmo com os avanços tecnológicos que possibilitaram uma melhor extração das especificidades de cada indivíduo(chamados dados biométricos, como as digitais ou a íris), a assinatura manuscrita continua indispensável e com grande adoção[3][4].

Para a verificação da identidade de um indivíduo utilizando dados biométricos, são necessários métodos com elevada acurácia. Analisando o cenário de técnicas para verificação de assinaturas manuscritas, apesar dos bons resultados encontrados na literatura [1][4][5][6], ainda existe espaço para melhorias, inclusive nas arquiteturas baseadas na extração de características para generalização dos modelos em diferentes situações(como um grande número de usuários)[7][8].

A utilização de modelos de Redes Neurais Convolucionais(CNN) é amplamente aceita na comunidade científica[4], alcançando ótimos resultados em taxas de erro. Porém, existem aspectos na arquitetura dos modelos que podem ser explorados e melhorados, como as funções de perda[8].

1.2 Justificativa

Grande parte das implementações de modelos CNN para verificação de assinaturas utilizam a função de perda *Cross-Entropy*[8] e que alcançam bons resultados. Porém algumas redes CNN implementam a função de perda *Triplet Loss*[9], principalmente em arquiteturas de redes siamesas[6] e obtém ótimos resultados. Portanto o conhecimento sobre o desempenho das funções de perda em diferentes modelos CNN deve ser cuidadosamente considerado para a seleção mais adequada em diferentes cenários.

Além do desempenho na verificação, é importante considerar os custos computacionais no treinamento dos modelos, e as dificuldades nas suas implementações. Uma diferença ínfima nas taxas de acerto e erro entre as funções de perda, pode significar que é possível decidir em favor do uso da função mais simples e/ou menos computacionalmente custosa.

1.3 Objetivos

1.3.1 Objetivo Geral

O principal objetivo é analisar o uso de diferentes funções de perda a partir de algumas arquiteturas de modelos de verificação de assinaturas, para entender se existem situações em que seu uso trará benefícios em relação às outras funções analisadas.

1.3.2 Objetivos Específicos

- Explorar as funções de perda no domínio das CNNs;
- Implementar a função *Cross-Entropy*, *Triplet Loss* e *Contrastive Loss*;
- Selecionar e implementar arquiteturas CNN do estado da arte no domínio da verificação de assinaturas;
- Utilizar as funções de perda implementadas nos modelos de CNN disponíveis;
- Comparar as vantagens e desvantagens da utilização de cada uma das funções de perda implementadas;

1.4 Organização do TCC

Este trabalho se divide em 4 capítulos, a partir do próximo: no capítulo 2 é feita uma breve explicação dos principais conceitos fundamentais para o entendimento do tema, tanto da verificação de assinaturas quanto das Redes Neurais Convolucionais; No capítulo 3, os métodos e procedimentos aplicados no trabalho serão explicados, assim como informações sobre a base de dados, modelos CNN, funções escolhidas e pré-processamento utilizados; No capítulo 4, será descrito os resultados obtidos, além de uma análise das informações encontradas; Por último, no capítulo 5 estão as conclusões deste estudo a partir dos resultados.

Capítulo 2

Fundamentação Teórica

2.1 Verificação de Assinaturas

Cada indivíduo compõe sua assinatura a partir de um conjunto único de características[1]. Para a confirmação da identidade da pessoa através de sua assinatura, profissionais especializados(chamados de peritos grafotécnicos) utilizam diversos critérios que definem as especificidades que uma pessoa possui naturalmente ao escrever[10]. Com o reconhecimento dos critérios existentes na assinatura de uma pessoa, é possível compará-la com outras assinaturas e identificar um usuário previamente cadastrado[1][3] ou ainda verificar a autenticidade de documentos assinados[7].

Um dos grandes desafios para a verificação de assinaturas é o método em que as mesmas são coletadas[10]. Existem essencialmente dois modos de verificação: *on-line* e *off-line*. No *on-line*, é utilizado um sistema de assinatura eletrônica capaz de armazenar informações sobre velocidade e ordem enquanto a assinatura é realizada, além da própria assinatura finalizada[2][4][5].

No modo *off-line*, é armazenada apenas a assinatura após a escrita, dificultando a extração das características do usuário, pois toda a verificação se baseia em pixels que representam o formato da assinatura[5][7]. Entretanto, com o modo *off-line* é possível se aproveitar de documentos com assinaturas já realizadas anteriormente, mesmo sem o uso de equipamentos apropriados(como mesas digitalizadoras)[2].

2.1.1 Análise Grafotécnica

A análise grafotécnica(também conhecida como exame grafotécnico) é realizada a partir da constatação das características utilizadas pelo autor na escrita da assinatura[10]. É comum utilizar assinaturas já escritas(*off-line*), portanto o número de informações disponíveis está limitado aos dados existentes nas imagens[7].

Os peritos que analisam as assinaturas (também conhecidas na grafotécnica como peças questionadas as que estão sendo verificadas, e peças padrão de confronto as genuínas utilizadas para comparação) buscam as características gráficas mais relevantes a partir de critérios conhecidos, e as compara com as características já armazenadas do autor verdadeiro. A partir do resultado desta comparação, é possível decidir se a assinatura testada é genuína ou forjada[1].

Na Figura 1 está um exemplo da complexidade no ramo de verificação de assinaturas. Duas assinaturas são genuínas, e uma é forjada (terceira imagem). Apesar das três serem diferentes, os peritos grafotécnicos são capazes de identificar os critérios na escrita do usuário, e verificar se as assinaturas são genuínas.



Figura 1. Exemplos de duas assinaturas genuínas e uma forjada.

Os critérios utilizados pelos peritos buscam evidenciar as técnicas utilizadas pelo autor durante a assinatura[10]. Alguns exemplos de critérios são:

- Ataque, que caracteriza o início do movimento gráfico;
- Remate, que identifica o final do movimento gráfico;
- Momentos gráficos, que são as análises de como o autor realiza o movimento durante o toque do instrumento escritor (caneta) com o suporte (papel), além dos espaçamentos entre os movimentos.

É importante salientar que a extração das características não é simples, pois alterações pontuais do autor (como o humor ou nervosismo) impactam na escrita, assim como os meios utilizados (papel e caneta, ou mesa digitalizadora)[4]. Na Figura 2 está um exemplo de uma análise grafotécnica, a partir de duas assinaturas: uma peça questionada e uma peça padrão de confronto.

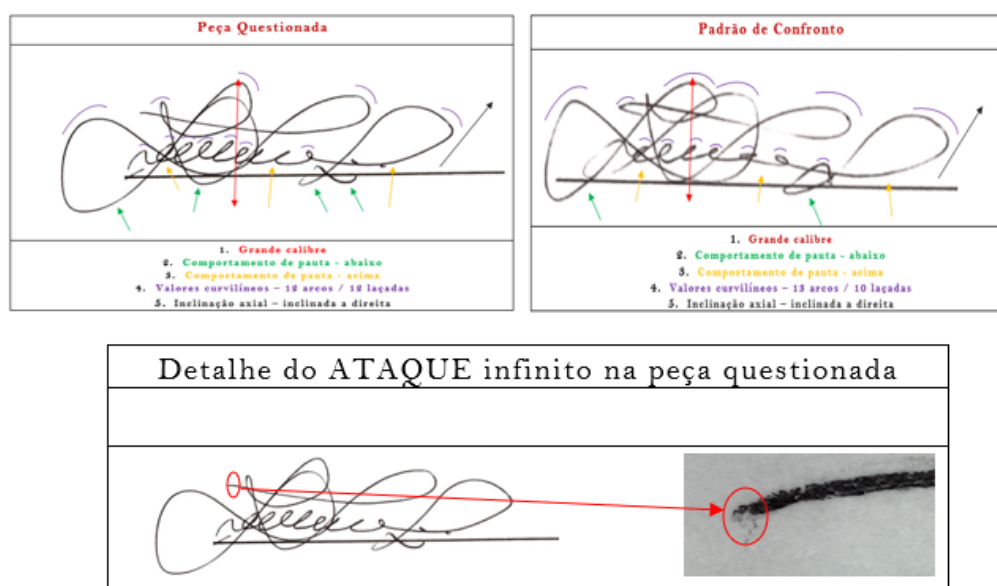


Figura 2. Exemplo de uma análise grafotécnica.

2.2 Redes Neurais Convolucionais

As Redes Neurais Convolucionais(CNN) são arquiteturas de aprendizagem profunda(*Deep Learning*), muito utilizadas para extração de características em problemas complexos como classificação e detecção de objetos, além de reconhecimento de imagens[11].

Apesar dos modelos CNNs serem diferentes entre si, os conceitos básicos são os mesmos[12]:

- É fornecida uma entrada, comumente no formato de vetores multidimensionais;
- A rede realiza uma série de operações e transformações a partir da entrada, possibilitando a extração de características automaticamente.
- Com as características(em inglês, *features*), é utilizado um modelo de classificação que pode tomar uma decisão a partir das características extraídas. Por exemplo, inferir sobre qual classe de objetos a entrada fornecida à CNN pertence.

As operações realizadas a partir da entrada são separadas em camadas, cada uma em busca da extração de certas características[13]. Desconsiderando a primeira e última(entrada e saída), as camadas são chamadas de “camadas

ocultas”, e o conjunto de múltiplas delas em sequência é denominado *deep learning*[12]. Na Figura 3 é ilustrada uma arquitetura CNN(a etapa de extração de características), assim como uma etapa de classificação. O conjunto das duas etapas compõem as operações básicas para o treinamento dos modelos.

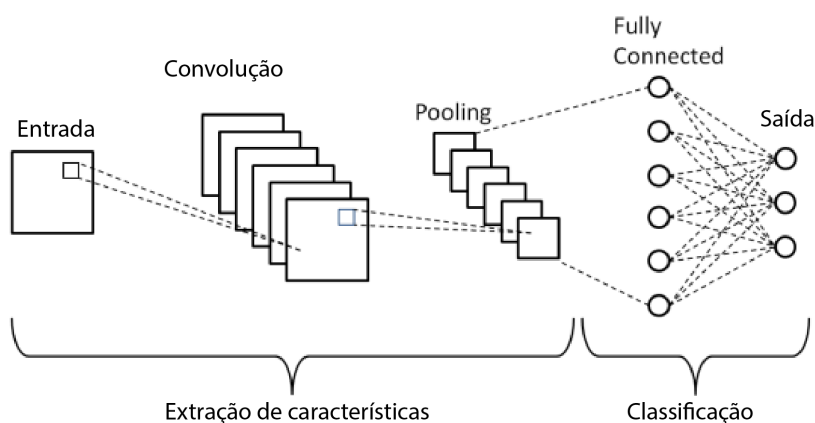


Figura 3. Exemplo de uma arquitetura CNN.

Entre as operações realizadas durante o processamento de uma CNN, a mais notória(e que dá nome ao tipo de rede) é a convolução. Considerando uma sequência de camadas em que a saída de uma camada se torna a entrada para a próxima, a convolução aplica filtros específicos na entrada recebida, separando características da imagem original. Com isso, uma camada que aplica a convolução pode capturar os contornos, formas ou texturas de uma imagem, entre outras *features*[11][12].

Além da convolução, existem outros conceitos matemáticos muito importantes na elaboração de CNNs, tais como:

- *Pooling*, que busca reduzir a complexidade da rede nas camadas subsequentes. Essa redução na complexidade se dá a partir da escolha de *pixels* que mais representam uma seção da imagem(o *pooling* faz a “junção” dos *pixels*, e daí vem a origem do termo). Uma técnica muito utilizada é conhecida como “*max-pooling*”[12], que a partir de uma imagem, observa-se seções de tamanho definido(como 5x5), e gera uma nova saída com o *pixel* de maior valor em cada seção[13];
- Camadas *Fully-Connected*, que recebe as características extraídas pela CNN e as classifica entre classes definidas anteriormente. No contexto de verificação de assinaturas, tais classes poderiam ser o usuário da assinatura

a partir de uma lista de usuários cadastrados, ou ainda uma classificação entre genuína ou forjada[14]. Esta etapa pode ser substituída por outros métodos de classificação, como SVM(*Support Vector Machine*).

A essência de uma Rede Neural Convolucional é sua arquitetura, descrevendo a ordem das camadas de convolução, *pooling*, *Fully-Connected*, além dos detalhes específicos de cada camada (como número de *features* e tamanho do filtro). Cada arquitetura possui vantagens e desvantagens, dependendo de seu propósito, das imagens e/ou dados de entrada ou escolha dos hiperparâmetros. O uso de CNNs para verificação de assinaturas possibilita altos índices de acurácia[5].

2.3 Funções de Perda

Durante o treinamento de uma Rede Neural Convolucional, existe um aspecto da arquitetura que não é muito estudado: as funções de perda[8]. Tais funções possuem um papel importante na qualidade do treinamento da rede, afetando diretamente sua velocidade e acurácia[8][15], pois é o modo que o modelo CNN reconhece onde pode ser aperfeiçoado, e decide quais as mudanças a fazer em suas camadas.

As funções de perda são utilizadas para calcular o erro cometido pelo modelo CNN, em relação ao resultado desejado. A escolha da função depende do objetivo do modelo, além da entrada da rede[8].

2.3.1 Cross-Entropy

A função mais utilizada nos modelos com melhores desempenhos é a *Cross-Entropy*, também conhecida como *log loss* quando se trata de modelos de *deep learning*[8]. Esta função recebe a probabilidade de uma classificação, e calcula a probabilidade esperada (encontrada a partir da utilização da função *Softmax*). O valor da perda (*loss*, em inglês) é a soma entre as probabilidades entre cada uma das classes que o modelo busca classificar.

A equação geral para o *Cross-Entropy* está a seguir:

$$L_{CE} = - \sum_{i=1}^n t_i \log(p_i), \text{ onde:} \quad (1)$$

- p_i é a probabilidade da classe i , encontrada pelo modelo CNN;
- t_i é a probabilidade esperada para a classe i , calculada a partir da função *Softmax*.

A função *Softmax* calcula a probabilidade em que os dados possuem de pertencer a cada uma das classes analisadas na CNN:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{k=1}^K e^{z_k}}, \text{ onde } z_i \text{ é o elemento analisado.} \quad (2)$$

2.3.2 Contrastive Loss

Enquanto a função *Cross-Entropy* calcula um valor que pode ser usado diretamente na classificação em modelos CNN, a *Contrastive Loss* é implementada a partir do conceito de similaridade.

Na *Contrastive Loss*, é recebida características extraídas pela CNN de duas imagens (ao invés de *features*, são chamadas de *embeddings* por possuir mais características úteis da imagem, mesmo que para isso seja necessário possuir um tamanho maior)[8][9][16]. Caso as duas imagens pertençam à mesma classe (por exemplo, sejam assinaturas genuínas), seus *embeddings* terão certa semelhança, enquanto que caso as duas imagens sejam de classes diferentes (caso uma das assinaturas seja forjada), os *embeddings* serão dissimilares[19].

Com os *embeddings* das duas imagens, a *Contrastive Loss* calcula a distância entre as mesmas (utilizando a distância euclidiana, por exemplo). A perda (*loss*) é reduzida com menores distâncias, assim como é elevada com maiores distâncias[19].

A equação geral para a *Contrastive Loss* está a seguir:

$$L(W, (Y, a, b)^i) = (1 - Y)L_s(D) + YL_d(D), \text{ onde:} \quad (3)$$

- Y representa a similaridade entre as duas imagens. Caso sejam similares, então $Y = 0$. Caso contrário, $Y = 1$.

- L_s é a função que será usada caso as imagens sejam similares. Um exemplo da função é a distância euclidiana entre as duas entradas.
- L_d é a função aplicada quando as imagens forem dissimilares. Um exemplo dessa função é o inverso da distância euclidiana. Assim, o erro é maior caso as entradas sejam semelhantes.
- D é a distância calculada entre as duas imagens.

É válido notar a relação entre Y , L_s e L_d . No caso das imagens serem dissimilares, o primeiro termo da função se torna igual a 0, tornando a perda calculada a partir de L_d . Do mesmo modo, com imagens similares, o segundo termo da função se torna 0, fazendo com que a perda seja calculada a partir de L_d .

2.3.3 Triplet Loss

A *Triplet Loss* possui semelhanças à *Contrastive Loss*, pois calcula a perda a partir da similaridade entre suas entradas. É uma função de perda usada em situações onde as entradas possuem poucas diferenças entre si[9], e possui grandes diferenças em comparação com a *Cross-Entropy*.

Ao invés de receber as probabilidades a partir de uma função *Softmax*, a *Triplet Loss* recebe *embeddings* de três imagens. A escolha de quais imagens são utilizadas se baseia na relação entre elas. A partir de uma base de dados com assinaturas de vários usuários, é possível encontrar *Triples* através das seguintes regras:

- A primeira imagem é chamada de *anchor*(âncora), é uma assinatura aleatória de um usuário;
- A segunda imagem é chamada de *positive*(positiva), e é uma assinatura da mesma classe que o *anchor*, mas do mesmo usuário;
- A terceira imagem é chamada de *negative*(negativa), e precisa ser um exemplo negativo(classe diferente do *anchor*).

No problema de verificação de autenticidade de assinaturas, As *triples* podem ser formadas por duas assinaturas genuínas(âncora e positiva), e uma assinatura forjada(negativa), todas do mesmo usuário. Com os três *embeddings*, o cálculo da perda é feito a partir da similaridade entre eles: quanto mais similar o âncora for em

relação Δ positiva, menor será o erro. Do mesmo modo, o erro é maior caso o âncora seja mais similar ao negativo[9][16].

A equação para o *Triplet Loss* está a seguir:

$$L_{TL} = \sum_{i=1}^N \max\{d(a_i, p_i) - d(a_i, n_i) + \alpha, 0\}, \text{ onde:} \quad (4)$$

- $d(a, b)$ é o cálculo da similaridade entre os *embeddings* de duas imagens. É comum o uso da distância euclidiana;
- α é chamada de “margem”, é um valor adicionado na diferença entre as similaridades, indicando o quão diferente a imagem negativa precisa ser para as suas diferenças serem consideradas como erro. Assim, mesmo que a imagem negativa possua certa similaridade com o âncora, o erro não será considerado caso não ultrapasse a margem;
- $\max\{err, 0\}$ limita o erro para apenas valores positivos. Um valor de erro negativo significa que a imagem positiva possui mais similaridades com o âncora do que a imagem negativa. Nesse caso, o erro deve ser zero.

2.4 Trabalhos Relacionados

Na literatura existem trabalhos que analisam diferentes funções de perda, mas no contexto de verificação de assinaturas não foram encontrados testes com diferentes arquiteturas CNN e diferentes funções de perda, e que incluam funções baseadas em similaridade (como a *Contrastive Loss* e *Triplet Loss*).

No artigo “*On Loss Functions for Deep Neural Networks in Classification*”[8], os autores realizam uma análise detalhada com diferentes funções de perda, incluindo a *Cross-Entropy*. Porém, o foco é a função L_2 . É concluído que a função *Cross-Entropy*, apesar de ser a mais utilizada, pode ser substituída por outras funções dependendo do objetivo central. Neste artigo não foi discutido o uso de funções de perda que calculam similaridades.

Em “*Loss Functions for Image Restoration with Neural Networks*”[15], é feita uma análise de várias funções de perda para aplicações na área de restauração de imagens. Este estudo mostra a importância de verificar as possíveis alternativas, ao invés de escolher os parâmetros simplesmente de acordo com a literatura. A função

de perda L_2 , muito utilizada no ramo de restauração de imagens, não se mostrou melhor que outras funções estudadas.

Em “*Offline Signature Verification with Convolutional Neural Networks*”[2] é publicado um modelo CNN para verificação de assinaturas utilizando a *Contrastive Loss*. O modelo se baseia na rede VGG16, e foi considerado para análise neste estudo, porém necessitaria utilizar o conceito de *transfer learning* (treinar o modelo a partir de um modelo pré-treinado), e o modelo usado pelo artigo não está disponível gratuitamente na internet.

No artigo intitulado “*Learning Features for Offline Handwritten Signature Verification using Deep Convolutional Neural Networks*”[7] são publicados dois modelos CNN para verificação de assinaturas. Chamados de *SigNet* e *SigNet-F*, utilizam a função *Cross-Entropy* para alcançar resultados do estado da arte.

Em “*FaceNet: A Unified Embedding for Face Recognition and Clustering*”[20] utiliza uma rede CNN em conjunto com a função *Triplet Loss*, e alcança bons resultados em reconhecimento facial. Aqui, não houve estudos sobre aplicações na área de verificação de assinaturas.

Na publicação “*Recent Advances in Convolutional Neural Networks*”[15] os autores realizam uma revisão e explicação das técnicas utilizadas em modelos CNN. É verificado que a função *Contrastive Loss* é bastante utilizada em redes siamesas (que são redes onde existem múltiplas saídas, como vários modelos CNN diferentes). Já em “*SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification*”[6], uma rede siamesa é implementada utilizando a função *Contrastive Loss*, alcançando bons resultados quando comparado com outros trabalhos na literatura. Além disso, este trabalho analisa, entre outras bases, a base de dados GPDS, amplamente utilizada atualmente.

Outro artigo interessante é “*Face Recognition for Embedded System Based on Optimized Triplet Loss Neural Network*”[15], que aplica os conceitos de reconhecimento facial em sistemas embarcados, utilizando um modelo CNN baseado no *Inception v1* em conjunto com a função *Triplet Loss*

Capítulo 3

Procedimentos e Métodos

O estudo foi realizado a partir da implementação de algumas funções de perda, e sua aplicação em modelos CNN que são estado da arte. Cada função analisada foi implementada para este estudo, e seu desempenho foi comparado à uma versão disponível publicamente em bibliotecas (como *Pytorch*). A função *Cross-Entropy* foi utilizada a partir da versão disponível pelo *Pytorch*, por questões de otimização.

Os modelos CNN foram implementados seguindo as especificações de seus respectivos artigos, e possuem bons desempenhos. Porém, os resultados apresentados em seus artigos não serão considerados, pois a programação, base de dados e características foram alteradas para uma melhor análise das funções de perda. Com isso, não são esperados resultados iguais ou superiores aos encontrados originalmente pelos autores das arquiteturas CNN.

3.1 Arquiteturas Analisadas

As arquiteturas utilizadas são modelos CNN já publicados e testados. O código e testes disponibilizados pelos autores não foram considerados, apenas as especificações da arquitetura, como: número de camadas, filtros, normalização e hiperparâmetros. Os modelos implementados estão a seguir:

- *SigNet-F* [7][18];
- Siamesa[6] (no artigo original, esta arquitetura é chamada apenas de “*SigNet*”. Como esta rede utiliza o conceito de Redes Siamesas, será utilizado o nome “Siamesa”);

O modelo *SigNet-F*, que possui um dos melhores resultados na área de verificação de assinaturas, obtendo taxas EER de 1.72% em determinadas condições, foi publicado juntamente com outro modelo (chamado “*SigNet*”). A diferença entre os dois é o tipo de entrada recebida. Enquanto que o modelo *SigNet*

é treinado a partir de assinaturas genuínas de vários usuários, o modelo *SigNet-F* recebe também assinaturas forjadas, diversificando o treinamento. A arquitetura dos dois modelos são iguais, e existe esta separação pelo fato de que em situações reais, a disponibilidade de assinaturas forjadas por profissionais é bastante escassa[7]. Este trabalho utilizará o *SigNet-F*, por conta do alto número de assinaturas forjadas na base de dados.

Os autores do modelo propõem a classificação em duas partes: a primeira parte é um treinamento *Writer-Independent*, bastante similar às CNN comuns. A saída resultante dessa etapa é um modelo capaz de extrair características que diferenciam as assinaturas entre genuínas e forjadas, mesmo que outras assinaturas dos autores não tenham sido utilizadas no treinamento(o uso independe do autor).

Em seguida, é implementada uma parte *Writer-Dependent*, onde é feita uma classificação utilizando uma rede SVM(*Support Vector Machine*), recebendo como entrada as *features* extraídas de assinaturas, obtidas a partir do modelo treinado na primeira etapa. A saída é a classificação entre genuína e forjada.

Como é uma técnica diferente das demais CNNs e gera uma complexidade muito maior para treinar, foi decidido que esta segunda etapa será utilizada exatamente como discutido no artigo original. Além disso, não será possível comparar diretamente os resultados, pois utilizam uma base de dados diferente(e não mais acessível).

Tabela 1. Camadas do modelo CNN *SigNet-F*.

Camada	Tamanho	Parâmetros
Entrada	1x150x220	
Convolução 1	96x11x11	stride = 4, pad=0
Pooling	96x3x3	stride = 2
Convolução 2	256x5x5	stride = 1, pad=2
Pooling	256x3x3	stride = 2
Convolução 3	384x3x3	stride = 1, pad=1
Convolução 4	384x3x3	stride = 1, pad=1
Convolução 5	256x3x3	stride = 1, pad=1

Pooling	256x3x3	stride = 2
Fully Connected 1	2048	
Fully Connected 2	2048	

Na tabela 1 estão as camadas que implementam o modelo *SigNet-F*. É válido notar que após cada camada convolucional, é aplicado o *batch normalization*, e foi omitido da tabela apenas para facilitar a leitura.

O segundo modelo analisado também é chamado de *SigNet* em seu artigo, mas para fins de comparação, será chamado de Siamesa neste trabalho. O nome se dá ao uso do conceito de redes siamesas, onde são usadas duas redes CNNs com os mesmos parâmetros e pesos, gerando duas saídas (*features*) ao invés de uma.

Com as características, é realizada uma comparação pela similaridade. No caso da verificação de assinaturas, ao gerar *features* de duas imagens genuínas do mesmo usuário, elas serão similares, enquanto que duas imagens de classes diferentes (usuários diferentes, ou genuína e forjada) produzirão *features dissimilares*.

O modelo CNN Siamesa equivale à primeira etapa do treinamento do *SigNet-F*. As arquiteturas são semelhantes, diferenciando apenas em alguns parâmetros, além do número de neurônios na camada *Fully-Connected* (Totalmente conectada, a última camada da rede), onde a rede Siamesa possui 128 neurônios, contra 2048 na primeira etapa do *SigNet-F*.

Outra diferença é a função de perda: O *SigNet-F* calcula o erro a partir da função *Cross-Entropy*, enquanto que a Siamesa utiliza a *Contrastive Loss*, o que sinaliza uma diferença em como os autores pensaram para verificar as assinaturas, criando oportunidades de comparação. Os resultados da Siamesa não são melhores que o *SigNet-F*, mas entre as bases de dados testadas está a *GPDS Synthetic Signature Database*[17], também utilizada neste trabalho.

Na tabela 2 estão as camadas configuradas no modelo CNN Siamesa. É possível notar que é bem similar com a *SigNet-F*,

Tabela 2. Camadas do modelo CNN Siamesa.

Camada	Tamanho	Parâmetros
Entrada	1x150x220	
Convolução 1	96x11x11	stride = 1
Local Response Norm.		$\alpha = 10^{-4}$, $\beta = 0.75$ $k = 2$, $n = 5$
Pooling	96x3x3	stride = 2
Convolução 2	256x5x5	stride = 1, pad=2
Local Response Norm.		$\alpha = 10^{-4}$, $\beta = 0.75$ $k = 2$, $n = 5$
Pooling + Dropout	256x3x3	stride = 2, $p = 0.3$
Convolução 3	384x3x3	stride = 1, pad=1
Convolução 4	256x3x3	stride = 1, pad=1
Pooling + Dropout	256x3x3	stride = 2, $p = 0.3$
Fully Connected 1	1024	$p = 0.5$
Fully Connected 2	128	

O processo para treinar os modelos CNN é o mesmo para todas as arquiteturas, assim como o uso dos hiperparâmetros. Todos os modelos foram treinados várias vezes com todas as funções de perda analisadas, para inibir adversidades durante o processo de treinamento.

3.2 Funções de Perda Implementadas

As funções de perda escolhidas para o estudo são conhecidas por alcançar resultados do estado da arte, em conjunto com modelos CNN. A implementação se baseia em seus artigos originais, sem alterações na fórmula. As funções de perda escolhidas estão a seguir:

- *Cross-Entropy*[2];
- *Contrastive Loss*[6][19];
- *Triplet Loss*[16];

A escolha do uso da *Cross-Entropy* se deve ao fato de ser amplamente utilizada nos modelos CNN atualmente, atingindo altas taxas de acurácia no treinamento.

As funções *Contrastive Loss* e *Triplet Loss* possuem conceitos semelhantes: verificar a similaridade entre as entradas, e a partir do resultado, tornar possível a verificação das assinaturas. A *Contrastive Loss* é muito utilizada no ramo de verificação de assinaturas em conjunto com redes siamesas[6][19], enquanto que a *Triplet Loss* faz parte do estado da arte no ramo de reconhecimento facial[20].

Neste estudo, o uso de cada função de perda é o mesmo, diferenciando-se apenas pelo número de imagens recebidas. Foi gerado uma base de dados específica para cada função de perda, diminuindo o custo computacional no momento do treinamento.

3.3 Descrição da Base de Dados

A base de dados utilizada para treinamento dos modelos é a *GPDS Synthetic Signature Database*[17]. No total, são 54 assinaturas de 4.000 indivíduos sintéticos(gerados a partir das características de assinaturas reais), sendo 24 assinaturas genuínas e 30 forjadas, para cada usuário.

Por limitações técnicas, não foram usados todos os 4.000 usuários. O estudo utilizou o mesmo grupo de usuários em todos os testes, sendo ao todo 1300 usuários, divididos em três grupos:

- Treino e validação, são usados todas as assinaturas de 1000 usuários, que são separados antes do início do treinamento ao gerar os *batches*. A separação leva em consideração as assinaturas individuais, e não usuários, por isso as assinaturas do mesmo usuário podem ser encontradas em ordem aleatória nos dois grupos de dados. A divisão das assinaturas é 75% para o grupo de treino, e 25% para o grupo de validação.
- Teste, com 300 usuários, são usados exclusivamente para encontrar os resultados do estudo.

3.4 Pré-Processamento dos Dados

Os modelos CNN escolhidos recebem como entrada imagens de tamanho 150x220, que é menor que a resolução disponível pela base de dados *GPDS*[17], portanto é necessário um pré-processamento. Seguindo as informações disponíveis nos artigos onde foram publicados os modelos CNN[2][7][16], foi realizado as seguintes operações em todas as imagens:

- Redimensionamento das imagens para a resolução 150x220;
- Recorte da imagem, gerando mais variações;
- Remoção do fundo, utilizando o método de limiarização de *Otsu*;
- Normalização, convertendo os valores das imagens de [0-255] para [0-1];

A ordem das imagens são randomizadas, aumentando a generalização nos treinamentos. Além disso, cada função de perda possui especificidades (como número de imagens necessárias), então foi necessário gerar diferentes bases de imagens, preparadas para cada função.

Para a função *Cross-Entropy*, a base de dados é organizada em tuplas contendo três dados: a imagem, o código do usuário, e um *Boolean* que diz se a imagem é genuína ou forjada.

Para a função *Contrastive Loss*, a base de dados se organiza com duplas de imagens do mesmo usuário, sendo a primeira genuína, e a segunda escolhida aleatoriamente. Além das duas imagens, um terceiro dado indica se a segunda imagem é genuína ou forjada.

Para a função *Triplet Loss*, é gerado apenas tuplas com três imagens, sendo duas genuínas (representando o âncora e positivo), e uma imagem forjada (representando o negativo).

Considerando que as funções *Contrastive Loss* e *Triplet Loss* utilizam mais imagens em cada processamento, o número de itens disponíveis na base de dados será diferente:

- *Cross-Entropy*: 54.000 itens (todas as assinaturas de cada usuário são utilizadas);

- *Contrastive Loss*: 24.000 itens(24 duplas são usadas para cada usuário, sendo 12 duplas genuínas e 12 duplas com assinaturas forjadas). Seis imagens forjadas de cada usuário são descartadas.
- *Triplet Loss*: 12.000 itens(12 grupos com três imagens, sendo duas genuínas e uma forjada). Dezoito imagens forjadas de cada usuário não são utilizadas.

A organização e separação das imagens da base de dados foi feita utilizando métodos *off-line*, para uma menor utilização de recursos computacionais. Porém, o processamento das imagens foi realizado durante o treinamento.

3.5 Treinamento, Validação e Teste

O treinamento dos modelos CNN são idênticos aos seus artigos originais, diferenciando apenas a função de perda. Os hiperparâmetros de cada modelo são os mesmos que os publicados. Alguns dos hiperparâmetros são idênticos em seus artigos originais:

- Número de épocas = 60;
- *Momentum* = 0.9;
- *Optimizer* = SGD(*Stochastic Gradient Descent*, ou Gradiente Descendente Estocástico);
- Resolução da Imagem = 150x220.

Alguns dos hiperparâmetros são diferentes entre as arquiteturas, então foram respeitados os mesmos valores propostos originalmente. Na tabela 3 estão os hiperparâmetros aplicados em cada uma das arquiteturas.

Tabela 3. Hiperparâmetros diferentes entre as arquiteturas analisadas.

	<i>Learning Rate</i>	<i>Batch size</i>	<i>Weight Decay</i>
<i>SigNet-F</i>	0,001 (x0,1 a cada 20 épocas)	32	0,0001
Siamesa	0,0001 (x0,1 a cada 20 épocas)	128	0,0005

As comparações entre os modelos foram realizadas a partir de uma métrica conhecida: EER(*Equal Error Rate*, ou taxa de erro igual). Além disso, foi observado

o andamento do treinamento, a fim de encontrar problemas como *overfitting/underfitting*.

A arquitetura CNN proposta com o *Signet-F*[18] realiza uma segunda etapa de classificação com o conceito de *Writer-Dependent*, utilizando uma biblioteca *Python*(*sklearn*). Este treinamento é único para este modelo, por isso foi implementado seguindo as mesmas especificações do artigo original, assim como o cálculo das métricas analisadas.

Para a análise a partir da arquitetura Siamesa[6], foi utilizado um código em *Python* que calcula as métricas automaticamente[21].

Capítulo 4

Resultados e Discussão

Foram analisados dois modelos CNN(*SigNet-F* e *Siamesa*), e três funções de perda(*Cross-Entropy*, *Contrastive Loss* e *Triplet Loss*).

4.1 SigNet-F

Durante o treinamento dos modelos *SigNet-F*, foi possível perceber uma tendência nos valores de perda(*loss*) e acurácia(*acc*). O *Cross-Entropy* conseguiu os melhores valores, principalmente por ser mais adequado à etapa *Writer-Dependent*. A Figura 4 compara os valores encontrados durante a validação no treinamento.

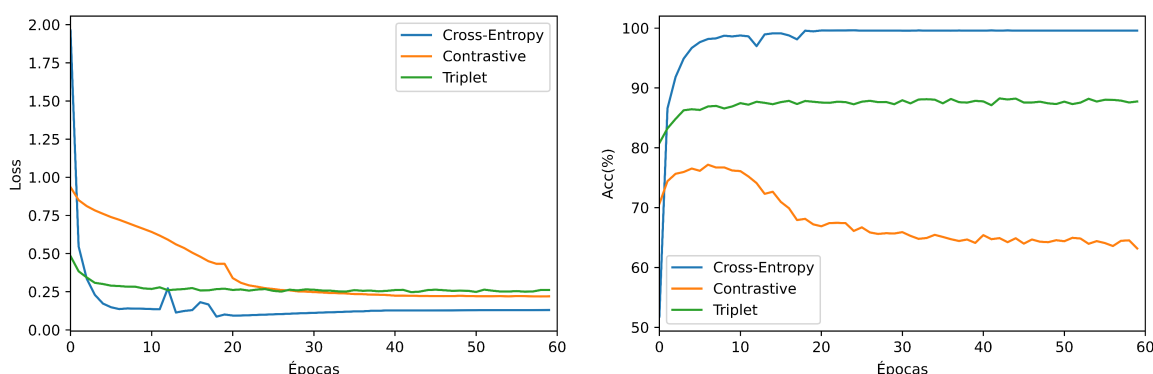


Figura 4. Comparação do *loss* e *acc* no modelo *SigNet-F*.

Analisando os gráficos da função *Contrastive Loss*, é possível notar que os valores destoam das outras funções. Isso se deve à própria equação, já que é preciso escolher um limiar para calcular a acurácia. No *Triplet Loss*, a acurácia é encontrada contabilizando o número de resultados iguais a zero(que significa que não houve erro). No *Contrastive Loss*, esse limiar entre erro e acerto não é zero, então é necessário decidir manualmente(testando vários valores diferentes para encontrar o ótimo), mas não se trata do foco do estudo analisar a acurácia na validação.

Nos testes, os resultados obtidos pelo modelo *SigNet-F* foram similares. Na tabela 4, é possível notar que a melhor taxa EER foi obtida ao utilizar a função

Cross-Entropy, com 18,62%. Ao aplicar a *Triplet Loss* o resultado não foi muito distante(21,19%), enquanto que a função *Contrastive Loss* gerou a pior métrica(mas ainda alcançou valores similares aos outros dois).

Tabela 4. Resultados dos experimentos com o *SigNet-F*.

<i>SigNet-F</i>	EER
<i>Cross-Entropy</i>	18,62%
<i>Contrastive Loss</i>	24,97%
<i>Triplet Loss</i>	21,19%

Para calcular a taxa EER, o modelo *SigNet-F* utiliza as *features* extraídas num segundo modelo mais simples, que realiza a classificação entre genuína/forjada.

A função *Triplet Loss* alcançou resultados satisfatórios, considerando que o modelo *SigNet-F* não foi arquitetado para trabalhar com esta função. Mesmo assim, o bom resultado mostra que esta função auxilia o modelo a extrair melhor as características das assinaturas.

Outro ponto a ressaltar é a organização da base de dados. Como explicado no item 3.4, o número de *Triplets* utilizado é bem menor, pois as imagens precisam atender os requisitos(duas assinaturas genuínas e uma forjada). Portanto, ao utilizar a função *Triplet Loss*, foram alcançados resultados semelhantes em comparação ao uso do *Cross-Entropy*, mesmo com menos dados disponíveis.

A função *Contrastive Loss* não alcançou os mesmos resultados que as outras funções. Uma das razões é a quantidade menor de dados disponíveis em relação à *Cross-Entropy*. Além disso, a implementação da função utilizada no estudo é bastante sensível ao formato em que estão os dados(as *embeddings* extraídas pelo modelo), necessitando normalizar a entrada para que seja possível realizar o treinamento, e testar os parâmetros corretos manualmente para a função.

4.2 Siamesa

No treinamento com o modelo Siamesa, a Figura 5 mostra que em relação à perda, as funções por similaridade se saíram melhor que a *Cross-Entropy*. Também

é possível observar uma estabilização nos valores a partir da época 30, tanto no *loss* quanto na acurácia(*acc*). Novamente o gráfico da *Contrastive Loss* chama a atenção pela diferença, mas afeta no treinamento do modelo.

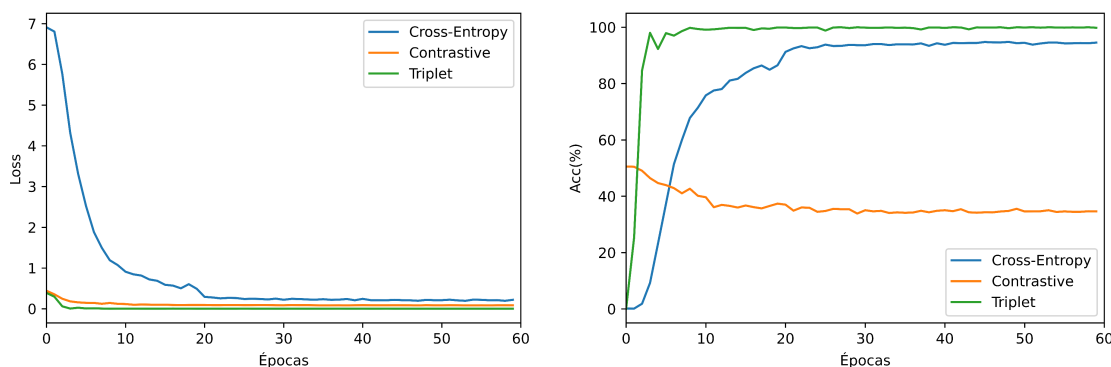


Figura 5. Comparação do *loss* e *acc* no modelo Siamesa.

Analisando os resultados obtidos com o modelo CNN Siamesa, é possível notar que a função *Triplet Loss* alcançou a melhor taxa EER(9,51%). Isso se deve ao fato de que o modelo foi originalmente implementado com a *Contrastive Loss*, que assim como a *Triplet Loss*, treina o modelo a partir das similaridades entre as assinaturas.

Tabela 5. Resultados dos experimentos com a Siamesa.

Siamesa	EER
<i>Cross-Entropy</i>	17,04%
<i>Contrastive Loss</i>	24,27%
<i>Triplet Loss</i>	9,51%

A função *Contrastive Loss* não obteve o melhor resultado, apesar de ter sido a função usada no artigo do modelo Siamesa. Como já foi elucidado anteriormente, esta função é mais sensível às *embeddings* de entrada, e o modelo Siamesa possui na última camada *Fully Connected* apenas um tamanho de 128 neurônios(a *SigNet-F* possui 2048 neurônios em sua última camada). Portanto, as funções de perda não possuem muitos dados para calcular o erro correto.

Com a função *Cross-Entropy*, os resultados também foram inferiores à *Triplet Loss*, principalmente pelo fato de que a função não calcula o erro a partir das similaridades, mas sim pelas probabilidades.

4.3 Discussão dos Resultados

Durante o desenvolvimento do experimento, foram realizados testes com diferentes hiperparâmetros. Foi constatado que a mudança no valor do *batch size* poderia estar afetando o desempenho dos modelos: enquanto que no treinamento as métricas continuavam idênticas, na validação e teste os resultados pioravam pois a capacidade do modelo de classificar novas assinaturas diminuía.

Com o modelo *SigNet-F*, o *batch size* padrão não levantou questionamentos, pois era um valor baixo. Já com o modelo Siamesa, o *batch size* apresentado no artigo era bem alto em comparação com outros modelos considerados[2][4][9]. Como o código desenvolvido para este estudo possibilita testar os hiperparâmetros facilmente, a partir de novos testes com diferentes *batch size*. Foram escolhidos os valores 32 e 64, por serem conhecidos na literatura.

Durante o treinamento dos diferentes modelos, foi possível perceber algumas diferenças. Com a função *Cross-Entropy*, ao diminuir o valor do *batch size* o modelo não obteve resultados melhores, e diminuiu sua capacidade de aprendizagem(somente após a época 20, onde ocorreu uma diminuição em 10 vezes da taxa de aprendizagem, que o modelo voltou a apresentar melhora em seus valores).

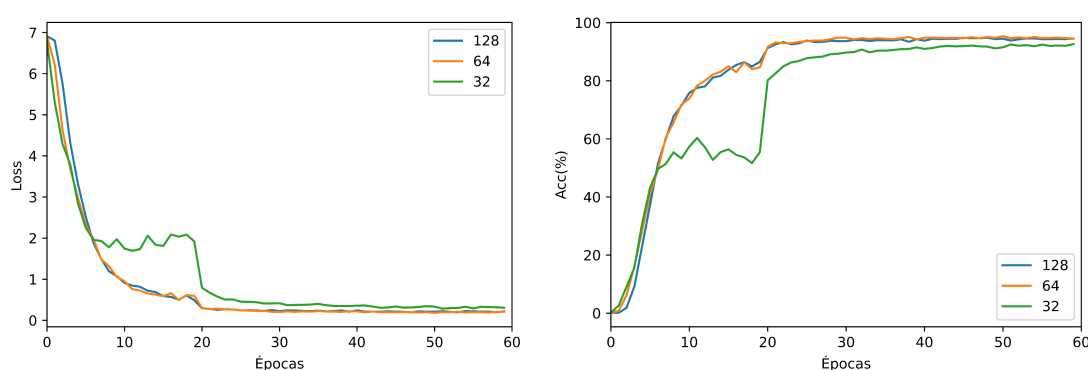


Figura 6. Análise do *batch size* no modelo Siamesa, com a função *Cross-Entropy*.

Analisando os resultados obtidos com a função *Contrastive Loss*, não houve uma diferença clara entre os diferentes testes. Porém, ao utilizar *batch size* = 32, o treinamento se tornou menos consistente, assim como na função analisada anteriormente.

Apesar da diferença entre os gráficos, os três testes se mantiveram similares, o que implica que a função não é afetada negativamente com o aumento ou diminuição do tamanho do *batch*. Além disso, pelos mesmos motivos citados anteriormente, os gráficos com a acurácia não estão dentro do padrão.

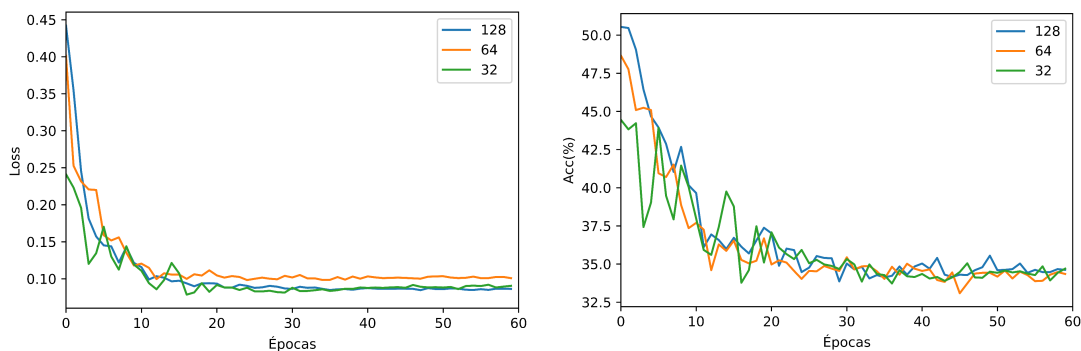


Figura 7. Análise do *batch size* no modelo Siamesa, com a função *Contrastive Loss*.

A última função a ser testada é a *Triplet Loss*. Novamente, a variabilidade é maior nos treinamentos com valor de *batch size* menores, e assim como a *Contrastive Loss*, não houve mudança significativa na *loss* ou acurácia.

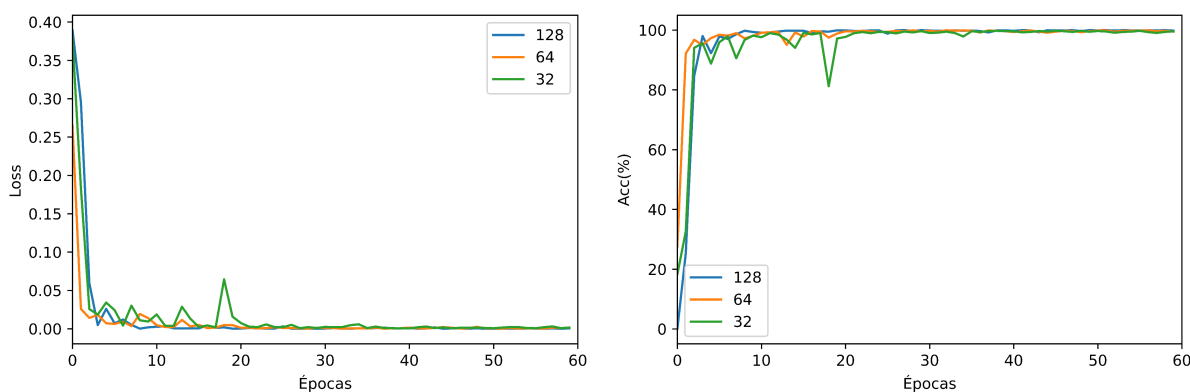


Figura 8. Análise do *batch size* no modelo Siamesa, com a função *Triplet Loss*.

Os dados das Figuras 7 e 8 mostram similaridades entre o comportamento do treino das funções que comparam diretamente as assinaturas (*Contrastive Loss* e *Triplet Loss*). Já com a *Cross-Entropy*, levando em consideração apenas o treino, houve uma leve diminuição na acurácia.

A tabela 6 apresenta as taxas EER no conjunto de teste, com modelos treinados por cada uma das funções de perda, nos três valores de *batch size* escolhidos: 32, 64 e 128, treinados no modelo CNN Siamesa.

Tabela 6. Análise da influência do *batch size* na taxa EER.

Batch size	128(padrão)	64	32
Cross-Entropy	17,04%	17,22%	18,09%
Contrastive Loss	24,27%	21,78%	12,82%
Triplet Loss	9,51%	7,92%	8,37%

Para calcular a taxa EER, é feita uma busca pelo limite entre correto e incorreto nas funções de erro. Essa etapa não é realizada na validação, e caso o fosse feito, melhoraria os valores dos gráficos de acurácia obtidos pela função *Contrastive Loss*.

É possível notar que o *batch size* não afeta significativamente o cálculo da função *Cross-Entropy*. Porém, a melhor generalização do modelo Siamesa significou um pequeno avanço nas taxas EER da função *Triplet Loss*. Também é possível perceber uma melhoria com a função *Contrastive Loss*, chegando próximo aos valores alcançados pela *Triplet Loss*. Isso indica que com mais ajustes em seus parâmetros, possui potencial de alcançar resultados mais promissores.

Observando os valores encontrados nos dois modelos analisados, o modelo pensado para utilizar uma função de perda comum alcançou melhores taxas ao combinar com a *Cross-Entropy*. Já o modelo Siamesa, que originalmente utilizava uma função de perda baseada na similaridade entre assinaturas, conseguiu melhores resultados ao aplicar justamente tais funções.

Durante o desenvolvimento da análise, foram realizados testes com outros hiperparâmetros (como a taxa de aprendizagem) e diferentes números de assinaturas disponíveis. Os resultados mostraram que não houve alterações significativas em relação ao uso dos valores propostos nos artigos originais das arquiteturas.

Capítulo 5

Conclusão e Trabalhos Futuros

Os testes realizados demonstram que o uso de funções de perda que calculam similaridade pode ser vantajoso, por obter bons resultados utilizando menos assinaturas para treinamento, além de melhor generalizar os modelos CNN. Porém, essas funções necessitam de ajustes-finos em sua programação, que podem influenciar positivamente ou negativamente o desempenho do modelo.

Modelos CNN desenvolvidos para certo tipo de função de perda, tendem a se saírem melhor em combinação com estas funções. Porém, o uso de funções mais especializadas para o problema (como uma variação do *Triplet Loss*) pode ser muito vantajoso, e vale a pena realizar testes iniciais para garantir que foi escolhida a melhor função.

A função de perda *Cross-Entropy* se mostrou a mais consistente, pois ao utilizá-la nos diferentes modelos e hiperparâmetros, os resultados se mantiveram próximos. Isso comprova o motivo para esta função ser o padrão na literatura: funciona bem em boa parte das situações e problemas.

As funções *Contrastive Loss* e *Triplet Loss* são mais sensíveis a quais assinaturas estão sendo usadas. Caso haja um bom método de escolha das próprias assinaturas, é possível utilizar uma base de dados menor e ainda assim alcançar bons resultados.

A maior dificuldade encontrada durante os estudos foi o recurso computacional limitado (a máquina utilizada possui uma GPU Nvidia 940mx, considerada de baixo desempenho). Com isso, algumas técnicas que podem ser usadas para melhorar o desempenho das funções de perda não puderam ser aplicadas.

A base de dados foi preparada e organizada separadamente antes do experimento. No caso das funções *Contrastive Loss* e *Triplet Loss*, que precisam receber duas ou três assinaturas de classes específicas, foi necessário criar duplas

e *triplets* que atendem aos requisitos necessários e armazenar num arquivo para ser carregado posteriormente. Com isso, a aleatoriedade da base de dados foi diminuída por se tratar sempre das mesmas duplas/*triplets*.

Existe uma técnica que faz o uso do *on-line mining*, onde as duplas e *triplets* são calculadas quando necessário, possibilitando um bom treinamento com ainda menos assinaturas. Porém, é computacionalmente custosa por ter que calcular a similaridade entre todas as assinaturas várias vezes durante o treinamento.

Como trabalhos futuros, mediante a disponibilidade de máquinas mais robustas para o processamento, seria interessante realizar testes com outras arquiteturas CNN que são estado da arte, principalmente as que aplicam as funções de perda que otimizam a rede com base na similaridade entre as imagens.

As funções de perda testadas possuem muitas variantes, e apesar de terem sido brevemente analisadas durante os procedimentos deste trabalho, não foram devidamente documentadas por não se tratar do foco da pesquisa. Por isso, é interessante realizar testes com diferentes versões.

É possível também abranger outros tipos de verificação de assinaturas. Este estudo considerou os casos em que o sistema precisa classificar se a assinatura é genuína ou forjada por profissionais. Entretanto, é interessante o teste para a identificação do usuário(classificar a identidade do usuário a partir da assinatura, ao invés de verificar sua genuidade).

Referências

- [1] IMPEDOVO, Donato; PIRLO, Giuseppe. **Automatic Signature Verification: The State of the Art**. IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: PART C: APPLICATIONS AND REVIEWS, [s. l.], v. 38, n. 5, 2008.
- [2] ALVAREZ, Gabe; SHEFFER, Blue; BRYANT, Morgan. **Offline Signature Verification with Convolutional Neural Networks**. 6th International Conference on Signal Processing and Communication (ICSC), [s. l.], 2020.
- [3] PLAMONDON, Réjean; SRIHARI, Sargur. **On-Line and Off-Line Handwriting Recognition: A comprehensive Survey**. IEEE Transactions on Pattern Analysis and Machine Intelligence, [s. l.], v. 22, n. 1, 2000.
- [4] DIAZ, Moises; FERRER, Miguel; IMPEDOVO, Donato; MALIK, Muhammad; PIRLO, Giuseppe; PLAMONDON, Réjean. **A Perspective Analysis of Handwritten Signature Technology**. ACM Computing Surveys, [s. l.], v. 51, n. 6, 2019.
- [5] Luiz G. Hafemann, Robert Sabourin, Luiz S. Oliveira, **Offline Handwritten Signature Verification - Literature Review**. Seventh International Conference on Image Processing Theory, Tools and Applications (2018).
- [6] DEY, Sounak; DUTTA, Anjan; TOLEDO, J. Ignacio; GHOSH, Suman K.; LLADÓS, Josep; PAL, Umapada. **SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification**. ArXiv, [s. l.], 30 set. 2017.
- [7] Luiz G. Hafemann, Robert Sabourin, Luiz S. Oliveira, **Learning Features for Offline Handwritten Signature Verification using Deep Convolutional Neural Networks**, Pattern Recognition (2017), doi: 10.1016/j.patcog.2017.05.012.
- [8] JANOCZA, Katarzyna; CZARNECKI, Wojciech. **On Loss Functions for Deep Neural Networks in Classification**. Faculty of Mathematics and Computer Science, Jagiellonian University, Krakow, Poland, [s. l.], 2017.
- [9] ZHAO, Yubo; YANG, Cheng; WANG, Yushi; CAI, Jing; XUE, Yanbing. **Face Recognition for Embedded System Based on Optimized Triplet Loss Network**. 3rd International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE), [s. l.], 2020.
- [10] DE MELO, Ana Patrícia Carvalho; BEZERRA, Byron Leite Dantas; JÚNIOR, Celso Antônio marcionilo Lopes; LIMA, Fernanda Gabrielle Andrade; LUCENA, Luciana Vaz de Oliveira; STODOLNI, Murilo Campanhol; MENESES, Denise Costa; ADVÍNCULA, Karina Paes. **Análise dos principais critérios utilizados em assinaturas e rubricas na perícia grafotécnica**. Revista Cefac, [s. l.], 2021
- [11] GU, Jiuxiang; WANG, Zhenhua; KUEN, Jason; MA, Lianyang; SHAHROUDY, Amir; SHUAI, Bing; LIU, Ting; WANG, Xingxing; WANG, Li; WANG, Gang; CAI, Jianfei; CHEN, Tsuhan. **Recent Advances in Convolutional Neural Networks**. ArXiv, [s. l.], 2017.

-
- [12] O'SHEA, Keiron; NASH, Ryan. **An Introduction to Convolutional Neural Networks**. ArXiv, [s. l.], 2015.
- [13] ALBAWI, Saad; MOHAMMED, Tareq Abed; AL-AZAWI, Saad. **Understanding of a Convolutional Neural Network**. ICET: The International Conference on Engineering & Technology, [s. l.], 2017.
- [14] HAFEMANN, Luiz G.; SABOURIN, Robert; OLIVEIRA, Luiz S. **Analyzing features learned for Offline Signature Verification using Deep CNNs**. 23rd International Conference on Pattern Recognition, [s. l.], 2016.
- [15] ZHAO, Hang; GALLO, Orazio; FROSIO, Iuri; KAUTZ, Jan. **Loss Functions for Image Restoration with Neural Networks**. IEEE Transactions on Computational Imaging, [s. l.], 2018.
- [16] SCHULTZ, Matthew; JOACHIMS, Thorsten. **Learning a Distance Metric from Relative Comparisons**. Advances in Neural Information Processing Systems 16, [s. l.], 2003.
- [17] FERRER, Miguel; DIAZ, Moises; MORALES, Aythami. **Static Signature Synthesis: A Neuromotor Inspired Approach for Biometrics**. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, [s. l.], 2014.
- [18] HAFEMANN, Luiz G.; SABOURIN, Robert; OLIVEIRA, Luiz S. **Writer-independent Feature Learning for Offline Signature Verification using Deep Convolutional Neural Networks**. International Joint Conference on Neural Networks (IJCNN), [s. l.], 2016.
- [19] CHOPRA, Sumit; HADSELL, Raia; LECUN, Yann. **Learning a Similarity Metric Discriminatively, with Application to Face Verification**. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), [s. l.], 2005.
- [20] SCHROFF, Florian; KALENICHENKO, Dmitry; PHILBIN, James. **FaceNet: A Unified Embedding for Face Recognition and Clustering**. ArXiv, [s. l.], 2015.
- [21] PINTO, Joao Ribeiro; CARDOSO, Jaime S.; CORREIA, Miguel V. **Secure Triplet Loss for End-to-End Deep Biometrics**. 8th International Workshop on Biometrics and Forensics (IWBF), [s. l.], 2020.