**UNIVERSIDADE ESTADUAL DE CAMPINAS**
**INSTITUTO DE COMPUTAÇÃO**

MC853 - Projeto em Sistemas de Programação

# Um Estudo Sobre Redes Neurais Siamesas Para o Reconhecimento de Impressões Digitais

Luana Felipe de Barros RA: 201705

14/07/2022

Resumo:

O reconhecimento de impressão digital é considerado o sistema de identificação biométrica mais confiável e preciso. A maioria dos sistemas tradicionais de identificação de impressões digitais usa tecnologia de processamento de imagem digital em vez de aprendizado profundo. Neste estudo avaliamos seis arquiteturas de redes neurais siamesas, baseadas em redes convolucionais, para a tarefa de reconhecimento de impressões digitais. Testamos três variações de redes siamesas criadas , além de utilizar modelos pré-treinados como AlexNet, VGG, ResNet-50 como backbone. O melhor modelo nomeado RNS2 atingiu 87,1% no conjunto de teste, sendo seu diferencial a utilização da Local Response Normalization, que se mostrou aliada na resolução de reconhecimento de impressões digitais.

# 1. Introdução

O reconhecimento de impressões digitais é atualmente um dos métodos mais confiáveis, aceitáveis e de baixo custo para biometria. É amplamente utilizado para autenticação de indivíduos, devido a propriedades como unicidade, visto que a probabilidade de duas pessoas possuírem a mesma impressão digital é extremamente baixa, estabilidade - em relação a impressão digital ser uma característica que pouco muda com a idade do indivíduo - e alta acurácia na identificação.

Figura 1 - Classificação de minúcias em uma impressão digital [1]


Com a possibilidade do uso da tecnologia para criação de sistemas de biometria, a automatização do processo de reconhecimento de impressões digitais tem se tornado cada vez mais comum. Um exemplo disso é a utilização de sensores em dispositivos móveis como principal método de biometria e segurança para autenticar um usuário.

Atualmente, a área de inteligência artificial tem sido usada na resolução de problemas envolvendo biometria, não somente de impressão digital, como também facial, utilizando a íris, voz etc. Em particular, a área de Deep Learning (DL, em português Aprendizado Profundo) têm se destacado, alcançando ótimos resultados principalmente na área de análise de imagens, devido a sua alta capacidade de extração de boas características para problemas de classificação e reconhecimento. Os algoritmos de reconhecimento de impressão digital usam informações detalhadas da imagem da impressão digital para extrair características e, em seguida, adotam uma regra de cálculo de similaridade específica para medir a semelhança da imagem da impressão digital para realizar o processo de reconhecimento de impressão digital [2]. Essas características podem ser pontos chaves (minúcias) e outros padrões visuais que o algoritmo irá julgar relevante para o reconhecimento. A crista ou epiderme, indicada como linhas pretas, e os vales (ou derme) referem-se aos espaços brancos entre as cristas, também são características utilizadas no processo. Ainda, existem atributos chamados de minúcias, que são pontos de mudanças na estrutura da crista [3]. A figura 1 mostra alguns tipos de minúcias normalmente extraídos de uma impressão digital.

Deep Learning tem sido aplicado para resolução de diversos problemas na área de biometria de impressão digital, como classificação, reconhecimento, segmentação, melhoria na captura da impressão digital, extração de minúcias, etc [3].

As redes neurais siamesas (RNS), introduzidas por Bromley and LeCun [10], têm se mostrado competitivas em tarefas que dependem de métricas de similaridade. Têm sido utilizadas em diversos domínios, como reconhecimento facial [11], detecção de fraudes na área financeira, reconhecimento de voz, reconhecimento de assinaturas, entre outros.

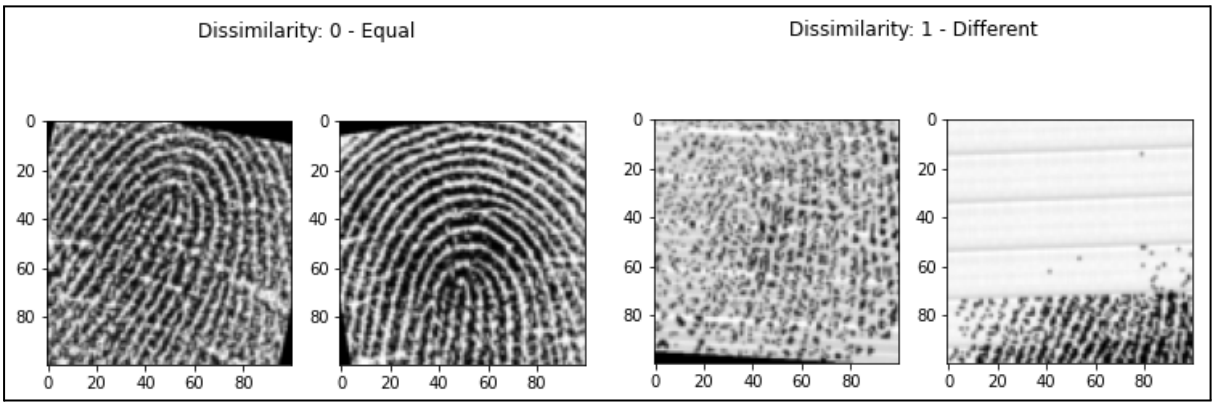
No contexto de impressões digitais, existem alguns trabalhos na literatura envolvendo RNS e Redes Neurais Convolucionais (CNNs). Por exemplo, L. Zhu et al. [4] treinaram três RNS: uma arquitetura definida por eles, *transfer learning* utilizando a VGG[9] como backbone e *transfer learning* utilizando a AlexNet[8] como backbone. Além de obter ótimos resultados, mostraram que a estrutura da CNN é a chave para afetar a acurácia do reconhecimento de impressões digitais.

Outro exemplo é a classificação de padrões de impressões digitais, como mostrado por Wu, F., Zhu, J. & Guo, X.[12]. Eles utilizaram uma arquitetura CNN, sem ser siamesa, com normalização *Local Response Normalization (LRN),* em que as impressões digitais foram classificadas em seis categorias, melhorando a acurácia de reconhecimento de impressões digitais a fim de facilitar a pesquisa nas características da personalidade humana.

Ainda, Prateek Nahar et. al [13] utilizaram a técnica de *transfer learning* utilizando a arquitetura ResNet-50[14] como backbone para a tarefa de reconhecimento de impressão digital com notável acurácia.

Neste trabalho, foi realizada uma série de experimentos - muitos deles baseados na literatura citada acima - utilizando RNS baseadas em CNNs para comparar o impacto da arquitetura da rede na precisão do reconhecimento de impressão digital.

Figura 2 - Duas Comparações ilustrando o processo de reconhecimento de impressão digital com a métrica de dissimilaridade.




## 2. Metodologia

A tarefa de reconhecimento de impressões digitais, é intrinsecamente uma tarefa de classificação de imagens em termos de dissimilaridade. Isto é, temos a classe genuína (com dissimilaridade igual a 0) e impostora (com dissimilaridade igual a 1). O modelo ideal deve ser capaz de generalizar e ser aplicado para dados não vistos anteriormente. Logo, a otimização dos pesos dos modelos é feita de tal forma a aprender a métrica de dissimilaridade. Para isso, o modelo precisa aprender a extrair características discriminativas, ou seja, que podem ser utilizadas para distinguir entre

as diferentes impressões digitais. Em termos técnicos, esse processo é chamado de Aprendizado Contrastivo e a função de custo utilizada em todas as arquiteturas deste trabalho é a Contrastive Loss [16]. As RNSs normalmente utilizam esta técnica para o processo de otimização dos pesos da rede com o algoritmo de *Backpropagation.*

A arquitetura de RNS consiste em duas redes neurais que compartilham pesos idênticos e que são ligadas por uma ou mais camadas. Uma RNS recebe como entrada um par de leituras com o objetivo de desenvolver similaridade entre pares de uma mesma classe - a classe genuína - e a distanciar pares de dados de classes diferentes - a classe impostora [6,7].

## 2.1. Base de Dados

A base de dados utilizada neste trabalho foi um subconjunto da base FVC-2006 [15]. Esta, consiste em quatro subconjuntos DB1, DB2, DB3 e DB4 em que foram coletadas imagens de impressões digitais com sensores diferentes. Em particular, utilizamos o subconjunto DB1. Esta base tem impressões digitais extraídas de 150 dedos, no qual cada dedo possui 12 amostras, totalizando 1800 imagens. Ela é divida em DB1_A para treinamento e validação e DB1_B para teste. O subconjunto DB1_A possui imagens extraídas de 140 dedos (1680 no total) e a DB1_B possui o restante das imagens - impressões digitais extraídas de 10 dedos (120 no total).

Como as redes siamesas recebem um par de imagens no processo de aprendizagem, para utilizar a base é necessário a criação de arquivos de comparações. Note que para cada dedo, há 12 amostras da mesma impressão digital. Quando estas são comparadas entre si, a comparação é considerada genuína. Ao compará-las com outros dedos, é considerada impostora. O arquivo de comparações da base DB1_A foi disponibilizado pelo professor da disciplina, contendo 18970 comparações, no qual escolhemos dividir como 70% para treino e 30% para validação, de forma que cada conjunto está balanceado. Para o dataset DB1_B, o arquivo de comparações disponibilizado continha 705 comparações, sendo 96% genuínas e 4% impostoras. Como não havia um número suficiente de comparações impostoras para avaliar o modelo, decidimos criar um novo arquivo de comparações para DB1_B. Este, agora contém 4800 comparações, sendo 70% impostoras e 30% genuínas. Um resumo desta organização pode ser visto na Tabela 1.

Tabela 1 - Divisão dos conjuntos e quantidade de comparações das classes Genuína e Impostora

| Conjunto | # Genuína | # Impostora | Total |
|---|---|---|---|
| Treino | 6501 | 6778 | 13279 |
| Validação | 2819 | 2872 | 5691 |
| Teste | 1440 | 3360 | 4800 |

## 2.2. Experimentos

### 2.2.1. Considerações

Neste processo, algumas modificações foram feitas no código original disponibilizado pelo professor. No laço de treinamento do modelo, ao invés de validar o modelo atual no final de cada época, agora o modelo é validado a cada 100 passos. Além disso, antes a acurácia reportada era em relação ao batch de dados de validação. Agora, calculo a média da acurácia de cada batch de validação e também da *loss* de validação para reportar. Ainda, implementei um *early stopping*, de modo a parar o treinamento se o modelo começar a piorar. Essas mudanças impactaram no número de épocas necessárias para o modelo convergir. Ademais, implementamos a acurácia balanceada como principal métrica, ao invés da acurácia comum. Isso porque o conjunto de teste é desbalanceado e o resultado com a acurácia normal seria superestimado.

### 2.2.2. Modelos

Todos os experimentos foram realizados em PyTorch, com a função de custo Contrastive Loss [16] e *constrastive threshold* igual a 1,1, otimizador Adam com *learning rate* igual a 0,001 e tamanho do batch como 32 amostras. Fizemos variações destes parâmetros, mas não obtivemos mudanças significativas.

#### 2.2.2.1. Baseline

Este experimento consiste em treinar uma rede siamesa convolucional, dada pelo professor da disciplina, com as modificações citadas acima em relação ao laço de treinamento, proporção de treino e validação e métricas. Esta rede recebe como entrada uma imagem 100x100 e possui dois blocos convolucionais, seguida por uma camada flatten e três camadas lineares com ativação ReLU. Como saída, para uma imagem, retorna um vetor de 64 dimensões. Cada bloco convolucional consiste em uma camada convolucional seguida de uma camada de batch normalization, então função de ativação ReLU seguida uma camada de MaxPooling.

Detalhes desta arquitetura podem ser vistos na Tabela 2. Há duas variações de experimentos utilizando essa arquitetura, com ou sem Data Augmentation, reportadas na seção 3 como BaselineAug e Baseline respectivamente.

Com a técnica de Early Stopping, o experimento Baseline rodou por 2 épocas, já o BaselineAug por 4 épocas.

Tabela 2 - Arquitetura da RNS Baseline

| Estágios | Shape de Saída |
|---|---|
| 1 - Bloco Convolucional | (None, 16, 100, 100) |
| 2 - Bloco Convolucional | (None, 128, 50, 50) |
| 3 - Flatten | (None, 80000) |
| 4 - Dense | (None, 512) |
| 5 - Dense | (None, 256) |

| | |
|---|---|
| 6 - Dense | (None, 64) |

#### 2.2.2.2. Transfer Learning com AlexNet

Baseado no trabalho de L. Zhu et al. [4], utilizamos a técnica de Transfer Learning com pesos congelados da AlexNet[8] como extratora de características. A AlexNet recebe de entrada uma imagem 256x256 e tem saída igual a 1000 pois foi treinada na base de dados ImageNet. Os detalhes desta do topo da arquitetura podem ser vistos na Tabela 3. Com a técnica de Early Stopping, o experimento AlexNet rodou por 5 épocas, sem Data Augmentation.

Tabela 3 - Arquitetura do Experimento AlexNet e ResNet50

| Estágios | Shape de Saída |
|---|---|
| 1 - Saída do Backbone | (None, 1000) |
| 4 - Dense | (None, 512) |
| 5 - Dense | (None, 256) |
| 6 - Dense | (None, 64) |

#### 2.2.2.3. Transfer Learning com ResNet50

Baseado no trabalho de Prateek Nahar et. al [13], utilizamos a técnica de Transfer Learning com pesos congelados da ResNet-50[14] como extratora de características. A ResNet-50 recebe de entrada uma imagem 224x224 e tem saída igual a 1000 pois também foi treinada na base de dados ImageNet. Os detalhes desta do topo da arquitetura podem ser vistos na Tabela 3. Com a técnica de Early Stopping, este experimento rodou por 5 épocas, sem Data Augmentation.

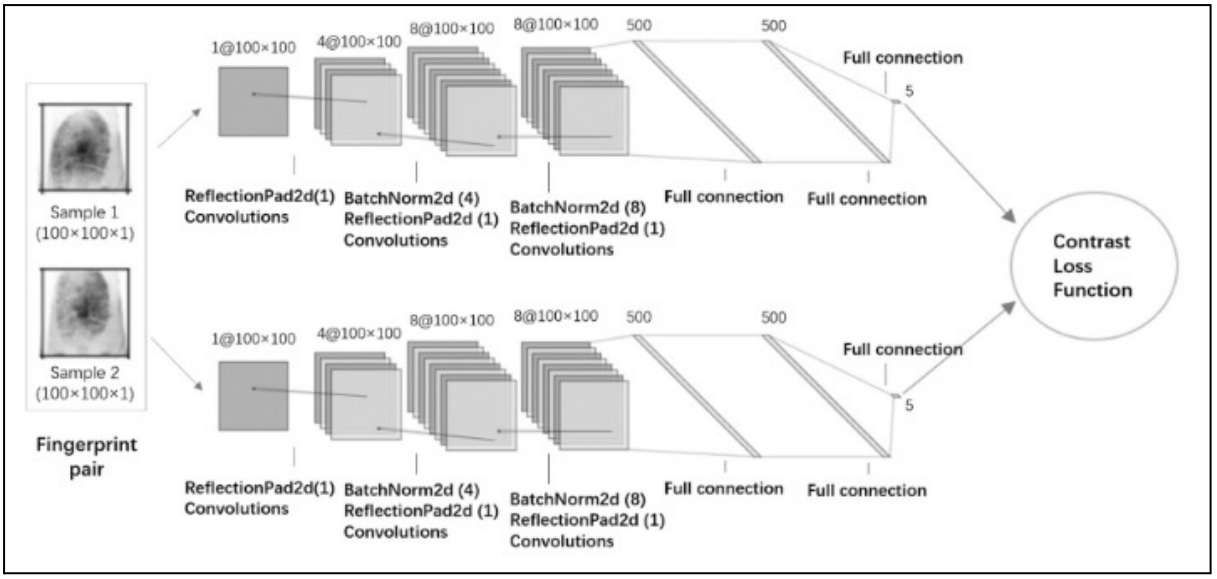
#### 2.2.2.4. Transfer Learning com VGG16

Baseado no trabalho de L. Zhu et al. [4], utilizamos a técnica de Transfer Learning com pesos congelados da VGG16[9] como extratora de características. A VGG16 recebe de entrada uma imagem 224x224 e tem saída igual a 4096. Com a técnica de Early Stopping, o experimento rodou por apenas 1 época visto que o resultado começou a piorar e sem Data Augmentation. Os detalhes desta do topo da arquitetura podem ser vistos na Tabela 4.

Tabela 4 - Arquitetura do Experimento VGG16

| Estágios | Shape de Saída |
|---|---|
| 1 - Saída Backbone | (None, 4096) |
| 4 - Dense | (None, 512) |
| 5 - Dense | (None, 256) |
| 6 - Dense | (None, 64) |

Figura 3 - Arquitetura da rede criada por L. Zhu et al. [4] (RNS1)




#### 2.2.2.5. Propostas de Mudanças na Arquitetura do Baseline

##### 2.2.2.5.1. Rede Neural Siamesa 1 (RNS1)

Neste experimento, testamos a rede definida por L. Zhu et al. [4], na qual se sobressaiu em desempenho comparada a VGG e AlexNet em seus experimentos. A arquitetura da rede é mostrada na Figura 3. Esta rede recebe como entrada uma imagem 100x100 e o vetor de saída é reduzido para 5 dimensões. Utilizando Early Stopping, o treinamento durou 10 épocas e não utilizamos Data Augmentation.

##### 2.2.2.5.2. Rede Neural Siamesa 2 (RNS2)

Wu, F., Zhu, J. & Guo, X [12], introduziram uma rede chamada FCTP-Net em que é uma rede convolucional na qual contém a normalização *Local Response Normalization* (LRN). A LRN utiliza um conceito de neurobiologia chamado de Inibição Lateral, que significa que um neurônio ativado suprime outro e combinado a função de ativação ReLU tem se mostrado útil [12]. Portanto, neste experimento decidimos utilizar a LRN no lugar do Batch Normalization no bloco convolucional da arquitetura Baseline, chamando esse experimento de RNS2. A arquitetura é análoga à descrita na Tabela 1.

## 3. Resultados

Nesta seção, apresentamos os resultados de cada experimento em termos de acurácia e também uma métrica de distância de cada classe (genuína ou impostora) a fim de verificar a capacidade de separabilidade das classes no espaço de características, utilizando threshold igual a 1,1. Para isso, dado um conjunto de avaliação, calculou-se a média e desvio padrão das distâncias euclidianas - obtidas por cada par de comparação - para cada classe. Os resultados no conjunto de validação e teste estão apresentados na Tabela 5 e 6 respectivamente.

Tabela 5 - Resultados dos Experimentos no conjunto de Validação

| Arquitetura | Validação | | | |
|---|---|---|---|---|
| | Acurácia Balanceada (%) | Loss | Distância - Genuína | Distância - Impostora |
| Baseline | 92,8 | 0,10 | 2,43 ± 0,84 | 0,67 ± 0,40 |
| BaselineAug | 81,4 | 0,18 | 2,12 ± 0,84 | 0,92 ± 0,46 |
| AlexNet | 81,4 | 0,19 | 1,64 ± 0,57 | 0,75 ± 0,44 |
| Resnet-50 | 93,2 | 0,09 | 1,98 ± 0,55 | 0,61 ± 0,33 |
| VGG16 | 74,4 | 0,26 | 1,84 ± 0,94 | 0,90 ± 0,53 |
| RNS1 | 93,5 | 0,08 | 2,29 ± 0,79 | 0,47 ± 0,33 |
| **RNS2** | **94,3** | **0,08** | **2,21 ± 0,75** | **0,53 ± 0,28** |

Tabela 6 - Resultados dos Experimentos no conjunto de Teste
(DB1_B com novo arquivo de comparações)

| Arquitetura | Teste | | | |
|---|---|---|---|---|
| | Acurácia Balanceada (%) | Loss | Distância - Genuína | Distância - Impostora |
| Baseline | 80,9 | 0,27 | 1,34 ± 0,41 | 0,52 ± 0,27 |
| BaselineAug | 80,1 | 0,17 | 1,80 ± 0,61 | 0,87 ± 0,46 |
| AlexNet | 50,5 | 0,65 | 0,67 ± 0,17 | 0,44 ± 0,21 |
| Resnet-50 | 60,3 | 0,50 | 0,91 ± 0,37 | 0,59 ± 0,34 |
| VGG16 | 74,2 | 0,21 | 1,65 ± 0,71 | 0,82 ± 0,40 |
| RNS1 | 74,5 | 0,38 | 1,17 ± 0,59 | 0,44 ± 0,24 |
| **RNS2** | **87,1** | **0,19** | **1,60 ± 0,56** | **0,63 ± 0,34** |

## 4. Discussão

Com os resultados, é possível perceber que a técnica de Data Augmentation não teve melhora significativa na arquitetura do Baseline. Como muitos outros experimentos foram baseados nele, não utilizamos mais essa técnica. Acreditamos que isso tenha ocorrido devido a quantidade de dados utilizados no treinamento, que não era tão grande e essa técnica pode ter atrapalhado o processo de aprendizagem.

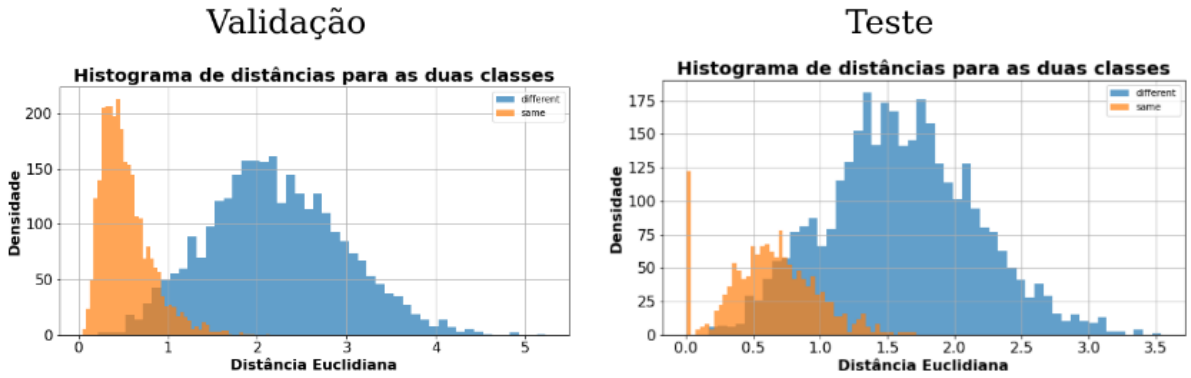
Em todos os experimentos, a acurácia no conjunto de teste foi menor, principalmente para ResNet-50 e RNS1 que tiveram uma diminuição significativa. Acreditamos que possivelmente isso deve-se a algum problema de distribuição dos

conjuntos de treinamento e teste. No teste, temos mais comparações impostoras, sendo mais fiel a um problema real. Seria importante o conjunto de treino apresentar mais amostras impostoras para ter uma melhor generalização e separabilidade melhor no espaço de características.

As métricas de distância euclidiana são importantes para demonstrar se há sobreposição das distâncias entre as classes. Este foi o caso da AlexNet, em que obteve uma baixa acurácia, devido a uma pobre separação entre as classes causando assim uma confusão para o modelo. Além disso, a AlexNet teve muita dificuldade de convergência após atingir 80% de acurácia na validação.

O melhor modelo para reconhecimento de impressão digital nesta base de dados foi o RNS2, mostrando que a camada de normalização *Local Response Normalization* teve impacto significativo no desempenho da rede, comparado a normalização comum. Na Figura 4, temos os histogramas de distâncias de cada classe como forma ilustrativa da separabilidade do modelo nos conjuntos de validação e teste. É possível obter que o modelo conseguiu separar melhor com threshold 1.1 para o conjunto de validação. No conjunto de teste, o ideal seria utilizar um threshold menor, para diminuir a sobreposição das distâncias. Isso confirma que a distribuição das amostras dos conjuntos era diferente. Em segundo lugar, temos a rede Baseline provida pelo professor da disciplina, que se mostrou muito promissora e importante para atingir bons resultados.

Figura 4 - Histogramas de Distâncias para as classes genuína (same) e impostora (different) para validação e teste no experimento RNS2




## 5. Conclusão

Neste trabalho testamos seis arquiteturas sobre redes neurais siamesas baseadas em redes convolucionais, variando desde a arquitetura em si aumentando ou tirando camadas, mudança de hiperparâmetros como *learning rate*, normalização, mudança no conjunto de teste, implementação de acurácia balanceada e mudança na lógica de treinamento para considerar salvar o melhor modelo baseado no resultado do conjunto de validação por completo. Além disso, houve uma pesquisa na literatura sobre redes neurais que solucionam problemas de dissimilaridade e também sobre aprendizado contrastivo.

Por fim, reiteramos que a arquitetura da rede RNS2 que utiliza *Local Response Normalization* foi mais adequada para resolução deste problema, sendo uma boa partida de estudo para próximos trabalhos.

Além disso, acreditamos que algumas estratégias podem ser exploradas a fim de melhorar ainda mais os resultados. Como por exemplo, combinar o peso da distribuição das classes com a Contrastive Loss; usar conjuntos de treinamentos mais diversos e voltados para a distribuição do mundo real; utilizar outra função de custo como a triplet loss; implementar a técnica de validação cruzada para melhorar o processo de treinamento e validação, de modo a garantir um modelo menos enviesado. Além disso, poderia modificar o treinamento para reconhecer impressões digitais extraídas de imagens parciais. Isso é importante porque muitos sensores hoje utilizam partes da impressão para o reconhecimento, como dispositivos móveis para autenticação de usuários. Por fim, seria interessante testar a rede FCTP-Net, que foi a inspiração para utilizar a normalização LRN.

O código dos experimentos, checkpoints dos modelos e a base de dados com os arquivos de comparações utilizados podem ser encontrados em [16].

## Referências


[1] Imagem retirada de: https://www.joaodefreitas.com.br/impressao-digital.htm

[2] Zeng, F., Hu, S. & Xiao, K. Research on partial fingerprint recognition algorithm based on deep learning. *Neural Comput & Applic* **31,** 4789–4798 (2019). https://doi.org/10.1007/s00521-018-3609-8

[3] Haruna Chiroma (2021). Deep Learning Algorithms based Fingerprint Authentication: Systematic Literature Review. Journal of Artificial Intelligence and Systems, 3, 157–197. https://doi.org/10.33969/AIS.2021.31010.

[4] L. Zhu, P. Xu and C. Zhong, "Siamese Network Based on CNN for Fingerprint Recognition," 2021 IEEE International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology (CEI), 2021, pp. 303-306, doi: 10.1109/CEI52496.2021.9574487.

[5] R. Hadsell, S. Chopra and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping", *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pp. 1735-1742, 2006.

[6] Andrey Gustavo de Souza, Aplicação de Redes Neurais Siamesas na Autenticação de Condutores. Em: Anais do 14º Simpósio Brasileiro de Automação Inteligente; Ouro Preto.Minas Gerais.Brasil. Campinas : Galoá; 2019. Disponível em: https://proceedings.science/sbai-2019/papers/aplicacao-de-redes-neurais-siamesas-na-autenticacao-de-condutores.

[7] Harandi, M., Roy Kumar, S., and Nock, R. (2017). Siamese Networks: A Thing or Two to Know. Technical report.

[8] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems 25 (2012): 1097-1105.

[9] Zhang, Xiangyu, et al. "Accelerating very deep convolutional networks for classification and detection." IEEE transactions on pattern analysis and machine intelligence 38.10 (2015): 1943-1955.

[10] Bromley and LeCun.“Signature Verification using a “Siamese” Time Delay Neural Network”.

[11] Siamese Network in PyTorch with application to face similarity - Master Data Science. Disponível em: https://datahacker.rs/019-siamese-network-in-pytorch-with-application-to-face-similarity/

[12] Wu, F., Zhu, J. & Guo, X. Fingerprint pattern identification and classification approach based on convolutional neural networks. *Neural Comput & Applic* **32,** 5725–5734 (2020). https://doi.org/10.1007/s00521-019-04499-w
[13] Prateek Nahar, Dr. Sanjay Tanwani, Dr. Narendra S. Chaudhari, "FINGERPRINT CLASSIFICATION USING DEEP NEURAL NETWORK MODEL RESNET50", IJRAR - International Journal of Research and Analytical Reviews (IJRAR), E-ISSN 2348-1269, P- ISSN 2349-5138, Volume.5, Issue 4, Page No pp.1521-1537, December 2018, Available at : https://www.ijrar.org/papers/IJRAR1944186.pdf
[14] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).
[15] R. Cappelli, M. Ferrara, A. Franco and D. Maltoni, "Fingerprint verification competition 2006", Biometric Technology Today, vol.15, no.7-8, pp.7-9, August 2007.
[16] R. Hadsell, S. Chopra and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping," 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), 2006, pp. 1735-1742, doi: 10.1109/CVPR.2006.100.
[16] https://github.com/httplups/fingerprint_recognition/