Erick Ramos dos Santos

Constellation Loss em Modelos de Reconhecimento Facial

Brasil

Vitória, 2022

Erick Ramos dos Santos

Constellation Loss em Modelos de Reconhecimento Facial

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica.

Universidade Federal do Espírito Santo

Programa de Pós-Graduação em Engenharia Elétrica

Orientador: Prof. Dr. Patrick Marques Ciarelli

Brasil

Vitória, 2022

Ficha catalográfica disponibilizada pelo Sistema Integrado de Bibliotecas - SIBI/UFES e elaborada pelo autor

---

Santos, Erick Ramos dos, 1992-
S237c Constellation Loss em modelos de reconhecimento facial / Erick Ramos dos Santos. - 2022.
61 f. : il.

Orientador: Patrick Marques Ciarelli.
Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Identificação biométrica. 2. Processamento de imagens. 3. Aprendizagem discriminativa. 4. Sistemas de reconhecimento de padrões. 5. Redes neurais (Computação). 6. Inteligência artificial. I. Ciarelli, Patrick Marques. II. Universidade Federal do Espírito Santo. Centro Tecnológico. III. Título.

CDU: 621.3

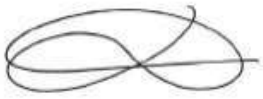
Erick Ramos dos Santos

Constellation Loss em Modelos de Reconhecimento Facial

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica.

Trabalho aprovado. Brasil, 04 de abril de 2022:

Prof. Dr. Patrick Marques Ciarelli
Orientador



Prof. Dr. Luiz Alberto Pinto
Instituto Federal do Espírito Santo

Prof. Dr. Filipe Wall Mutz
Instituto Federal do Espírito Santo

Brasil
Vitória, 2022

Agradecimentos

Primeiramente agradeço à Deus, pois só Ele para me dar forças e sabedoria para superar por todos os momentos de luta que passei durante todo o percurso. Agradeço à minha esposa, por ter sido a pessoa que mais sentiu comigo o meu cansaço e desafios dessa caminhada, sendo meu porto seguro sempre que precisei, mesmo que ao mesmo tempo esteja vivenciando o momento mais difícil de sua vida. Agradeço aos meus pais pelo exemplo que sempre me deram dentro de casa, sendo meus maiores exemplos de vida e me proporcionando um amor incondicional, sempre me motivando a continuar e não desanimar. Agradeço aos meus irmãos por todas as palhaçadas que deixavam meus dias mais leves e fáceis de passar. Agradeço também aos meus sogros, visto que mesmo neste deserto que estão enfrentando, sempre se preocupavam comigo e oravam por mim. Agradeço à todos os professores que tive durante toda minha vida, pois não sei o que seria do meu futuro sem estes homens e mulheres que dedicam suas vidas ao ensino. Agradeço em especial ao meu orientador, Patrick Ciarelli, por ter tido paciência comigo ao me ver fazer malabarismo para conciliar todos pontos da minha vida, me guiando e me ensinando sempre que precisei. Agradeço também ao professor Jorge Aching, que me ajudou diretamente no trabalho, tendo bastante sabedoria ao tirar cada uma das minhas dúvidas. Por fim, agradeço ao Programa de Pós-Graduação em Engenharia Elétrica da UFES, por ter me dado esta chance única em minha vida.

"Mas graças a Deus, que sempre nos conduz vitoriosamente em Cristo
e por nosso intermédio exala em todo lugar
a fragrância do seu conhecimento."
2 Coríntios 2:14

Resumo

Modelos extratores de características para sistemas de reconhecimento facial se tornaram objetos de profundo estudo ao longo dos últimos anos. Desejando encontrar uma melhor discriminação dos objetos e, consequentemente, uma melhor separabilidade das classes, funções de perdas, como a *Triplet Loss*, foram projetadas para serem empregadas em conjunto com as Redes Neurais Siamesas. Entretanto, redes com essas funções sofrem com uma lenta convergência por considerar apenas duas classes (positiva e negativa) a cada iteração de aprendizagem, e assim não são apropriadas quando existe uma grande quantidade de classes. Recentemente, uma função de perda chamada *Constellation Loss* foi proposta no sentido de minimizar este problema.

Neste trabalho é proposto um modelo para reconhecimento facial usando uma Rede Neural Convolucional (*Convolutional Neural Network* - CNN) como *backbone* e o *Constellation Loss* como função de perda. Para validar o modelo, foram utilizadas duas bases de dados públicas e feitas comparações com diferentes funções de perda e arquiteturas CNN. Também é proposto neste trabalho o uso de uma abordagem para construção dos *batches*, o qual permite o treinamento da rede com uso reduzido de memória.

Os resultados obtidos indicam que *Constellation Loss* é uma técnica promissora quando comparada às demais funções de perda avaliadas, alcançando valores médios de AUC (*Area Under The Curve*) iguais a 99,9% no conjunto de dados *Olivetti Faces* e 98,7% no desafiador conjunto de dados *Labeled Faces in the Wild* (LFW). A efetividade do método pôde ser certificada, viabilizando sua aplicação para sistemas de reconhecimento facial.

Palavras-chave: Constellation Loss, Reconhecimento Facial, Métrica de similaridade, Redes Neurais Siamesas, Aprendizagem profunda de métricas.

Abstract

Feature extracting models for facial recognition systems have become objects of in-depth study over the past few years. In order to find a better discrimination of objects and, consequently, a better separability of classes, loss functions, such as Triplet Loss, were designed to be used in conjunction with Siamese Neural Networks. However, networks with these functions suffer from a slow convergence by considering only two classes (positive and negative) at each learning iteration, thus, they are not appropriate when there is a large number of classes in the dataset. Recently, a loss function called Constellation Loss was proposed in order to minimize these problems.

In this work, a model for facial recognition using a Convolutional Neural Network (CNN) as a backbone and Constellation Loss as a loss function is proposed. To validate the model, two public databases were used and comparisons were made with different loss functions and CNNs architectures. It is also proposed in this work the use of an approach for the construction of batches, which allows network training with a reduced memory usage.

The results obtained indicate that Constellation Loss is a promising technique when compared to the other loss functions evaluated, reaching average values of AUC (Area Under The Curve) equal to 99.9% in the Olivetti Faces dataset and 98.7% in the challenging Labeled Faces in the Wild (LFW) dataset. The effectiveness of the method could be certified, enabling its application to facial recognition systems.

Keywords: Constellation Loss, Facial Recognition, Similarity Metrics, Siamese Neural Networks, Deep metric learning.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

BPNN	*Back-Propagation Neural Networks*
CNN	*Convolutional Neural Network*
DCT	*Discrete Cosine Transform*
FC	*Fully Connected*
GPU	*Graphics Processing Units*
HOG	*Histograms of Oriented Gradients*
ICA	*Independent Component Analysis*
IoT	*Internet of Things*
LDA	*Linear Discriminant Analysis*
LFW	*Labeled Faces in the Wild*
NN	*Nearest Neighbor*
PCA	*Principal Component Analysis*
RNS	*Redes Neurais Siamesas*
SGD	*Stochastic Gradient Descent*
SIFT	*Invariant Feature Transform*
SOM	*Self Organizing Map*
SURF	*Speeded Up Robust Features*
t-SNE	*t-distributed Stochastic Neighbor Embedding*

Sumário

1 Introdução

O reconhecimento facial é uma das maneiras mais utilizadas pelos seres humanos para identificação de indivíduos. De acordo com uma pesquisa realizada pelo LastPass (LASTPASS, 2018), no Brasil, 78% das pessoas afirmam confiar mais na impressão digital ou no reconhecimento facial do que nas senhas em texto como método de autenticação. Além de rápido e preciso, os sistemas de reconhecimento facial otimizam o processo de identificação e o faz de forma extremamente segura. Seja para acesso a dados e informações, seja para a tentativa de acesso físico, o reconhecimento facial atua para inibir tentativas de fraudes ou para diminuir a exposição de um local aos riscos.

Segundo a Pesquisa Global sobre Fraudes e Crimes Econômicos feita pela PwC (PricewaterhouseCoopers) em 2020 (PWC, 2020), considerando-se tecnologias e técnicas disruptivas no Brasil, aproximadamente 40% das empresas estão usando inteligência artificial (IA) em comparação com apenas 25% no mundo. Mais de 60% dos que usam IA no mundo perceberam seu valor como ferramenta de combate à fraude.

Os dados ainda mostram que empresas de todos os portes foram vítimas de em média 6 casos de fraudes no mundo nos últimos 24 meses. Dentre os principais autores de fraudes apontados pela pesquisa, no Brasil, 44% são agentes internos, distribuídos por equipe operacional, gerência média e alta administração. Com a implementação da tecnologia para identificação de identidade por meio do reconhecimento facial e analítico de vídeo, o número de fraudes pode diminuir drasticamente nas empresas.

Com isso, estudos na área são feitos há anos buscando alcançar um sistema completamente automatizado de reconhecimento facial utilizando visão computacional. Sistemas de reconhecimento facial se iniciam pela detecção da face, seguida pela extração das características e, por fim, a etapa de classificação, que visa verificar se uma face específica corresponde a alguma disponível em um banco de imagens de faces. Logo, ao desenvolver um projeto na área, as escolhas do detector, do extrator de características e do classificador precisam ser feitas de forma bem estruturadas, pois afetarão diretamente nos resultados do projeto.

Alguns trabalhos, como (GOLDSTEIN et al., 1971), (COTTRELL; FLEMING, 1990), (TURK; PENTLAND, 1991a) e (TURK; PENTLAND, 1991b), obtiveram bons resultados na área antes da virada do século, destacando-se as Redes Neurais Siamesas (RNS), introduzidas em Bromley et al. (1993). A RNS consiste em uma arquitetura que faz uso de redes neurais idênticas, na qual o modelo recebe entradas distintas, visando calcular uma métrica de distância entre os *embeddings* de cada entrada, ou seja, seus vetores de características. Entretanto, foram as Redes Neurais Convolucionais (*Convolutional Neural*

Network - CNN) que revolucionaram os modelos de reconhecimento facial. Ao contrário dos algoritmos propostos até então, que utilizavam características definidas previamente, as CNN trouxeram uma nova abordagem: uma rede neural que extrai as suas próprias características, e ela mesma define quais são as mais relevantes. De forma geral, técnicas de *Deep Learning* determinam de forma autônoma as regiões da face com maior poder de discriminação dos indivíduos.

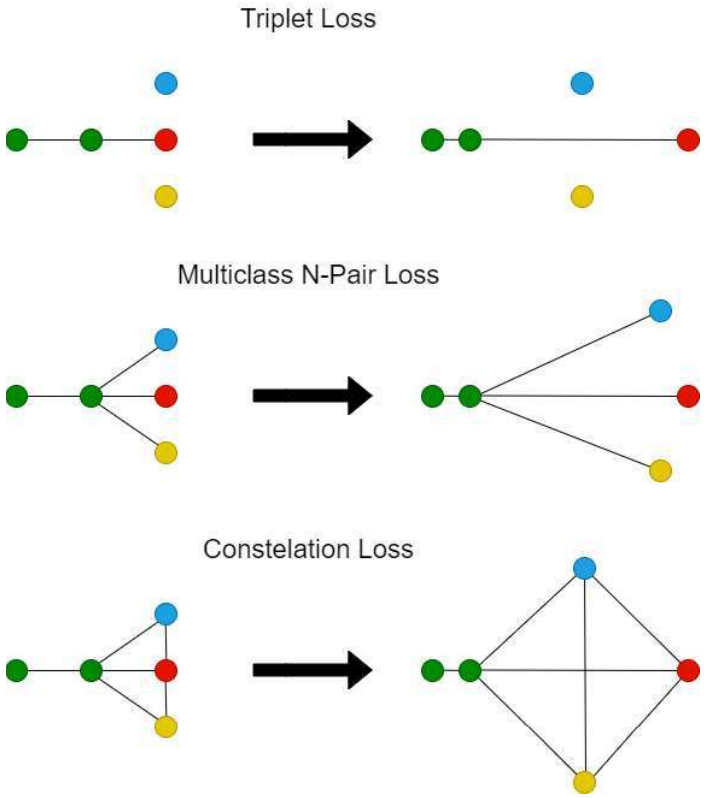
Com a popularização das CNN na classificação de imagens, o estudo em torno da aprendizagem profunda de métricas se tornou comum. A aprendizagem contrastiva em conjunto com as CNN's se tornou uma abordagem padrão para modelos de reconhecimento de padrões em imagens, mesmo em bancos de dados com poucos exemplos por classe, visto sua capacidade de generalização e efetividade na prevenção de *overfitting* em rótulos ruidosos das redes profundas (XUE et al., 2022). Apresentado por Chopra et al. (2005), o *Constrastive Loss* tornou-se uma função de perda natural em modelos que utilizam o conceito de RNS no treinamento, calculando o valor de perda em torno de pares de entradas de uma mesma classe ou de classes diferentes.

O trabalho proposto por Schroff et al. (2015) foi inovador para a área, trazendo um novo conceito aos algoritmos de extração de características. Nele, os pesquisadores configuraram um modelo utilizando CNN, comparando as arquiteturas Inception de Szegedy et al. (2014) e de Zeiler e Fergus (2013), para encontrar 128 características de uma face. Estas características de cada face são direcionadas a uma função de perda chamada *Triplet Loss*. O conceito da função é uma extensão do *Contrastive Loss*, porém, em vez de duas, são utilizadas três imagens para determinação das distâncias euclidianas entre as mesmas. A primeira imagem é chamada de âncora e as duas imagens subsequentes são chamadas de positiva e negativa, ou seja, a imagem positiva trata-se da mesma pessoa presente na imagem âncora e a negativa, não. O objetivo do uso da função é minimizar a distância da âncora para a imagem positiva, enquanto maximiza a distância para a imagem negativa, tornando a rede capaz de aumentar as distâncias de uma imagem para as diferentes classes presentes no conjunto de dados de treinamento, enquanto minimiza a distância intraclasse. Entretanto, a convergência deste tipo de metodologia é lenta em casos de conjuntos de dados com uma grande quantidade de classes, visto que a cada atualização a função verifica apenas duas classes.

Diante de tal cenário, Sohn (2016) propôs a função de perda *Multiclass-N-Pair Loss*. O método consiste em uma generalização do *Triplet Loss*, onde a comparação da âncora é feita com todas as classes presentes no conjunto de dados, ou seja, a cada iteração, ao minimizar a distância do exemplo positivo, a distância de $N - 1$ classes são maximizadas. Porém, mesmo alcançando uma convergência mais rápida, a perda de N pares persistiu com o problema de maximizar a distância das classes apenas em relação à classe da âncora.

Na busca de aproveitar o melhor dessas funções de perda, Medela e Picón (2019)

Figura 1 – Diagrama Visual das Funções de Perda. Cada cor representa uma classe



Fonte: (MEDELA; PICÓN, 2019)

propuseram a função *Constellation Loss*. Agora, a cada iteração de aprendizagem, as "distâncias negativas" são encontradas não apenas em relação à âncora, mas entre as próprias imagens das classes negativas. A Figura 1 ilustra, de forma simplificada, o procedimento das três funções de perdas mencionadas.

Sendo criado originalmente para o desenvolvimento de modelos capazes de reconhecer tumores colorretais por meio de imagens digitalizadas, até onde se tem conhecimento, o *Constellation Loss* foi usado pela primeira vez para reconhecimento de faces neste trabalho.

1.1 Método Proposto

O presente trabalho propõe um modelo extrator de características para reconhecimento facial usando uma CNN como *backbone* e o *Constellation Loss* como função de perda. Para validar o modelo, serão utilizadas duas bases de dados públicas: *Olivetti Faces* (OF) e *Labeled Faces in the Wild* (LFW), sendo feitas comparações com diferentes funções de perda e arquiteturas CNN. Cabe indicar que, para superar problemas de escalabilidade do modelo, é apresentada uma metodologia para a construção dos *batches* de treinamento

proposta por Medela e Picón (2019), que mostra ser uma abordagem apropriada para a arquitetura neural usada. O código desenvolvido durante este trabalho está disponível na plataforma GitHub [1].

1.2 Objetivos

O objetivo geral deste trabalho consiste em desenvolver um extrator de características para modelos de reconhecimento facial que alcance uma eficaz separabilidade das classes com uma rápida convergência. Para tal fim, pretende-se utilizar uma RNS que usa como *backbone* CNN pré-treinadas, as quais geram os *embeddings* relacionados a cada face. Na etapa de treinamento foi usada a função *Constellation Loss*, e uma metodologia particular para a conformação dos *batches* de treinamento.

O objetivo geral pode ser detalhado nos seguintes objetivos específicos:

- Desenvolver algoritmo para extrair características em imagens de faces;
- Desenvolver metodologia de construção dos *batches* de treinamento de forma a eliminar a necessidade de compartilhamento de pesos para arquitetura RNS;
- Realizar experimentos e comparar o desempenho de diferentes funções de perda;
- Realizar experimentos e comparar o desempenho de diferentes arquiteturas CNN.

1.3 Estrutura da dissertação

Este trabalho está organizado da seguinte forma: O Capítulo 2 apresenta um embasamento teórico sobre o reconhecimento facial, mostrando os desafios encontrados na área e o atual estado da arte. No Capítulo 3 são apresentadas as tecnologias já existentes utilizadas como base para este trabalho. O Capítulo 4 explica a metodologia desenvolvida para o modelo extrator de características proposto, e é feita a descrição das bases de dados utilizadas. No Capítulo 5 são expostas as métricas de avaliação e procedimentos experimentais executados, são exibidos os resultados obtidos e é feita a discussão deles. Por fim, o Capítulo 6 consiste em uma conclusão e a possibilidade de trabalhos futuros.

[1] https://github.com/ErickRDS/constellation_loss_face_recognition

2 Referencial Teórico

2.1 Reconhecimento Facial

A biometria é a ciência que mede e analisa os dados biológicos com o intuito de utilizá-los em mecanismos de identificação. Como cada ser humano possui características únicas, a tecnologia biométrica é vista cada vez mais como um recurso útil na área de segurança. Nesse contexto, a visão computacional merece destaque, sendo utilizada para tarefas de verificação biométrica como o reconhecimento de impressões digitais, digitalização da íris e reconhecimento facial, podendo este ser categorizado como verificação da face e identificação da face. A identificação confirma quem é a pessoa; enquanto a verificação confirma que trata-se realmente da pessoa identificada. Em qualquer um dos cenários, um conjunto de dados conhecidos é inicialmente registrado no sistema (banco de dados) e, durante o teste, um novo dado (imagem de entrada) é apresentado.

A face é considerada a parte mais crítica do corpo humano. Ela desempenha um papel crucial na interação com as pessoas na sociedade, transmitindo a identidade das pessoas, podendo, portanto, ser usada como chave para soluções de segurança em muitas organizações. O sistema de reconhecimento facial está cada vez mais em alta em todo o mundo como uma tecnologia de segurança extraordinariamente segura e confiável. Está ganhando importância e atenção significativas de milhares de organizações corporativas e governamentais devido ao seu alto nível de segurança e confiabilidade (SALAMA et al., 2020). Existem pelo menos duas razões para essa tendência: a primeira é a ampla aplicação comercial, principalmente na área de segurança, e a segunda é a disponibilidade de tecnologias viáveis após 30 anos de pesquisa (ZHAO et al., 2003).

Na prática forense e de segurança moderna, um software de reconhecimento automático da face é frequentemente usado para aumentar a velocidade e precisão de processos de identificação importantes. Uma aplicação cada vez mais comum da tecnologia de reconhecimento facial é conhecida como identificação um-para-muitos – em que algoritmos de correspondência de padrões são usados para comparar uma única imagem às imagens armazenadas em grandes bancos de dados (GROTHER; NGAN, 2014). Essa função pode ser usada para proteger contra fraudes de identidade ao emitir documentos de identidade nacional, como passaportes, vistos de imigração e carteiras de motorista, melhorando a detecção de solicitações duplicadas pelo mesmo indivíduo. A recente proliferação de provas baseadas em imagens de TV e dispositivos móveis implica que as imagens faciais são muitas vezes uma importante fonte de provas em investigações criminais. Em aplicações forenses, um software de reconhecimento facial permite que os policiais usem essa evidência de imagem para pesquisar em grandes bancos de dados de infratores conhecidos. Tecnologia

semelhante também é usada para aprimorar a experiência do usuário em plataformas populares de mídia social (WHITE et al., 2015).

O Facebook, por exemplo, cita a importância do reconhecimento facial em sua rede social:

> "Olhando para o futuro, ainda vemos a tecnologia de reconhecimento facial como uma ferramenta poderosa, por exemplo, para pessoas que precisam verificar sua identidade ou evitar fraudes e falsificação de identidade. Acreditamos que o reconhecimento facial pode ajudar produtos como esses com privacidade, transparência e controle, para que você decida se e como seu rosto é usado. Continuaremos trabalhando nessas tecnologias e contratando especialistas externos." (FACEBOOK, 2021).

A NVIDIA, empresa pioneira em construção de GPU (*Graphics Processing Units*, unidades de processamento gráfico) que auxiliam diretamente no poder computacional necessário para aplicar a metodologia *Deep Learning*, cita o seguinte:

> "O módulo de software de reconhecimento monitora continuamente as zonas-alvo para fornecer a contagem, sexo, idade e identificação única de indivíduos ao longo do tempo. O Modelo de Reconhecimento Facial rastreia indivíduos únicos e fornece correspondências faciais para indivíduos específicos. Isso permite melhorar a segurança geral, extrair maior valor da infraestrutura de segurança tradicional, ajudar os varejistas a reconhecer clientes importantes em tempo real ou quantificar a frequência de visitantes." (NVIDIA, 2020).

Entretanto, mesmo com diversas pesquisas na área, e sendo um cenário tecnológico que constantemente vem sendo aprimorado com novas descobertas, o reconhecimento de faces não é uma tarefa fácil, principalmente na presença de variações do ambiente onde a face deverá ser identificada. A pose facial e a iluminação, por exemplo, podem ser classificadas como situações de variações não controladas do ambiente.

O portal Asmag[1], um guia internacional de segurança e IoT (*Internet of Things*), detalha o quanto essas dificuldades afetam a qualidade do modelo:

> "Para que a tecnologia de reconhecimento facial funcione, existem certas condições que devem ser atendidas. Essas condições desempenham um papel crítico no sucesso do funcionamento do reconhecimento facial, principalmente em um sistema de reconhecimento facial um-para-muitos, em que o sistema compara faces a um banco de dados de fotos em vez de apenas uma foto. Infelizmente, muitas vezes as condições necessárias para o desempenho ideal são o maior desafio da tecnologia." (ASMAG, 2019).

Ao abordar modelos de reconhecimento de padrões, torna-se necessário o entendimento entre reconhecimento *offline* e *online*. No âmbito da identificação facial, o

[1] www.asmag.com

reconhecimento *offline* consiste de imagens obtidas em um ambiente controlado, com iluminação suficiente, onde o usuário interage com a câmera e assume uma posição ideal. Entretanto, o reconhecimento *online* traduz-se em situações de reconhecimento com uma câmera de vídeo em tempo real, onde na maioria das vezes as imagens captadas não estão em situações ideais, dificultando ainda mais a identificação do indivíduo. O portal Asmag pontua sobre as considerações a serem feitas ao sistema a ser instalado para captação em tempo real:

> "O posicionamento da câmera e a qualidade da captura de imagem são componentes críticos para o sucesso dos sistemas de reconhecimento facial. Se as imagens se originarem de câmeras de vigilância por vídeo de qualidade inferior e também estiverem fora do eixo, a imagem estática provavelmente não será ideal para reconhecimento facial. Se as câmeras forem definidas a uma distância para capturar grandes multidões, isso não é propício para o reconhecimento facial porque os rostos dos sujeitos estariam muito distantes durante a captura da imagem." (ASMAG, 2019).

Além disso, a utilização de óculos, chapéus, entre outros, colaboram negativamente para a eficácia do algoritmo de classificação. Dessa forma, para minimizar as influências externas no processo de reconhecimento facial, a utilização de técnicas como o pré-processamento da imagem desempenha um papel fundamental na correção e normalização das imagens.

Outra dificuldade muito importante a ser superada, essencialmente em aplicações um-para-muitos, é desenvolver um modelo capaz de lidar com conjuntos de imagens desbalanceados, ou seja, conjuntos em que cada classe possui um número de exemplos variado, o que acaba sendo os casos de maior ocorrência para bases públicas de faces. Assim, ao longo da última década, o foco de muitos estudos na área foi o desenvolvimento de metodologias capazes de classificar corretamente um indivíduo em meio a grandes conjuntos de imagens, mesmo este indivíduo tendo apenas uma imagem de sua face disponível. Para isso, o estado da arte atualmente são as Redes Neurais Siamesas unidas ao conceito das Redes Neurais Convolucionais Profundas (do inglês, *Deep Convolution Neural Networks*) (MEI; DENG, 2018), onde as diferentes funções de perda para determinação das distâncias entre cada classe presente no conjunto de dados se tornou um visado objeto de estudo.

2.2 Trabalhos Relacionados

Com o objetivo de avançar e buscar conhecimento no campo de reconhecimento facial, é preciso primeiro conhecer o que já foi desenvolvido por outros pesquisadores. Esta seção tem como objetivo conhecer a forma como esse assunto já foi analisado em estudos anteriores, apresentando as principais técnicas desenvolvidas na área.

O reconhecimento facial é uma tarefa extremamente desafiadora e, por esse motivo, a literatura é bastante diversificada quanto às abordagens relacionadas ao problema. Frequentemente, uma única aplicação utiliza múltiplos métodos motivados por diferentes princípios, isso faz com que esses sistemas sejam difíceis de serem classificados baseando-se apenas nas técnicas de extração de características e identificação implementadas (ZHAO et al., 2003). Apesar disso, de uma maneira geral é possível definir as técnicas de reconhecimento em três categorias:

- Métodos Holísticos: Este método é baseado na representação da imagem, sendo a imagem inteira levada em consideração ao invés de uma área específica ou pontos específicos da face. Esta metodologia se baseia no fato de que os dados da face geralmente possuem poucos destaques que são diretamente representados por *pixels* da imagem. Logo, o modelo precisaria de toda a imagem para conseguir diferenciar corretamente os vários indivíduos presentes em um bancos de imagens. Os exemplos para métodos holísticos são o PCA (*Principal Component Analysis*), o LDA (*Linear Discriminant Analysis*), o ICA (*Independent Component Analysis*), entre outros.

- Métodos de Correspondências Locais: Neste método são localizadas várias características faciais, como boca, nariz, olhos, etc., e, em seguida, calcula-se as relações geométricas entre aqueles pontos faciais, reduzindo assim a entrada da imagem facial a um vector de características geométricas (JAFRI; ARABNIA, 2009). A CNN é a técnica mais conhecida que utiliza esta metodologia.

- Métodos Híbridos: Assim como o sistema perceptivo humano, há métodos que usam todas as informações disponíveis para a avaliação, ou seja, tanto as características marcantes da face (locais) como todo o conjunto (globais). Assim, eles tentam extrair o melhor de cada abordagem (ZHAO et al., 2003). Correspondência de Características Geométricas é um exemplo conhecido de técnica com metodologia híbrida.

Grande parte dos primeiros métodos de reconhecimento facial faziam uso do PCA. O PCA é uma técnica estatística simples e útil usada para extração de características e redução de dimensão. A técnica consiste de um procedimento matemático em que as imagens originais são representadas em um espaço reduzido de componentes principais (PC), tal que a primeira dimensão ortogonal deste novo espaço carrega a maior quantidade de variância das imagens. Assim, o objetivo é representar as imagens em menores dimensões sem perder informações importantes. A Tabela 1 mostra os principais métodos que, ao utilizarem PCA, alcançaram resultados significativos.

Assim como o PCA, a Análise de Componente Independente (ICA) também é usada para representar os dados em menores dimensões. Ele pode ser usado para aplicativos de reconhecimento de faces para encontrar onde as informações importantes estão presentes nos

Tabela 1 – Principais métodos que utilizam PCA

Método	Técnica	Vantagens	Desafios
Kshirsagar et al. (2011)	O reconhecimento é feito pelo *Eigenfaces*	Determinação eficiente do espaço de menor dimensão	Diferentes variações de pose
Kadam (2014)	A redução das dimensões dos dados de entrada se faz mais rápida com a combinação do PCA com DCT (*Discrete Cosine Transform*)	O tempo é menor com o DCT e melhor desempenho no reconhecimento é alcançado do que um simples PCA em um algoritmo híbrido	Taxas de reconhecimento reduzidas para diferentes condições faciais
Bakhshi et al. (2016)	Para classificar imagens de face com PCA, utiliza-se um método que aproveita os recursos SIFT (*Invariant Feature Transform*) e SURF (*Speeded Up Robust Features*)	A rotação e a mudança nas expressões terão melhores resultados de correspondência quando o método PCA for aplicado	SIFT e SURF são um pouco mais complexos e demorados
Hazim (2016)	Utiliza-se PCA com DCT. Para o reconhecimento, faz-se uso de *Back-Propagation Neural Networks* (BPNN)	A combinação de BPNN com PCA facilita o reconhecimento facial no sistema	São necessários mais dados de treinamento para cada amostra

Fonte: Tradução do autor de tabela retirada de (RAZZAQ et al., 2021)

pixels de uma imagem. O PCA não é adequado para redução de dimensão de dependências de ordem superior, mas o ICA é adequado principalmente para dependências de segunda ordem e de ordem superior. A Tabela 2 representa os métodos relacionados ao ICA.

Tabela 2 – Principais métodos que utilizam ICA

Método	Técnica	Vantagens	Desafios
Sharma e Dubey (2014)	Utiliza-se PCA e ICA, sendo o reconhecimento feito por redes neurais	O sistema de reconhecimento facial é robusto e confiável, tendo características das faces invariáveis	Dependente de um ambiente bem iluminado
Bhat e Pujari (2015)	Combinação do PCA e ICA para redução de dimensionalidade	A acurácia é maior com a combinação do PCA e ICA do que o ICA sozinho	Desafiador lidar com variações de emoções
Movellan e Sejnowski (2002)	Através dos neurônios sigmoidais, o princípio da transferência ótima de informações é utilizado na nova versão do ICA	Alta confiabilidade em mudanças de expressões para reconhecimento das faces	Aplicável apenas para imagens com altas dimensões

Fonte: Tradução do autor de tabela retirada de (RAZZAQ et al., 2021)

Nos anos 2000, pesquisadores buscaram implementar sistemas de reconhecimento de padrões que faziam uso apenas de redes neurais, utilizando diferentes tipos de redes. Isso ocorreu devido à repercussão alcançada pelo trabalho proposto por Lawrence et al. (1997), cujo sistema combinava uma amostragem local da imagem, uma rede neural SOM (do inglês, *Self Organizing Map*) e uma CNN. Este sistema conseguiu alcançar 96,17%

de taxa de acerto em um conjunto com 400 imagens de 40 indivíduos diferentes. Este trabalho abriu espaço para novas pesquisas onde as redes neurais eram utilizadas também na etapa de extração de características.

Tendo em vista a limitação computacional da época, as técnicas de detecção e extração de características para fins de reconhecimento de padrões, que faziam uso das redes neurais, perderam espaço para algoritmos emergentes, capazes de usar de forma eficiente e com bom desempenho os hardwares com baixas capacidades de processamento e armazenamento, sobretudo com conjuntos de testes de maiores dimensões. Entre esses trabalhos podem ser mencionados os trabalhos de Viola e Jones (VIOLA; JONES, 2001) (VIOLA; JONES, 2004) para detecção da região da face, e o algoritmo HOG (*Histograms of Oriented Gradients*) (ALBIOL et al., 2008), que pode ser utilizado para a detecção e extração de características da face do indivíduo. Alguns trabalhos fazem uso da união desses algoritmos com redes neurais (BANDALA et al., 2013) (DA'SAN et al., 2015) (DESHPANDE; RAVISHANKAR, 2016).

Com o desenvolvimento de hardwares com maiores capacidades de processamento e armazenamento, e a maior acessibilidade a estes componentes, técnicas que requeriam maior custo computacional tornaram-se viáveis. Sistemas de *Deep Learning* (LECUN et al., 2015) passaram a ganhar destaque na área de visão computacional por ser um sistema em que melhores resultados podem ser obtidos com conjuntos maiores de imagens. Depois que as redes profundas são treinadas em dados massivos com a supervisão de uma função de perda apropriada, cada uma das imagens de teste é passada pelas redes para obter uma representação das características da face.

Inspirado pelo surpreendente sucesso no desafio ImageNet (RUSSAKOVSKY et al., 2014), arquiteturas CNN foram introduzidas e amplamente usadas como os modelos de linha de base em reconhecimento facial (de forma direta ou ligeiramente modificadas). Exemplos que podem ser citadas são AlexNet (KRIZHEVSKY et al., 2012), VGGNet (SIMONYAN; ZISSERMAN, 2014), GoogLeNet (SZEGEDY et al., 2015a), ResNet (HE et al., 2016a) e SENet (HU et al., 2018). Ao treinar uma CNN com uma grande quantidade de imagens, resultados significativos foram alcançados na etapa de detecção em (LI et al., 2015) e em (FARFADE et al., 2015); e de extração de características. Luo et al. (2012) desenvolveram um modelo utilizando Redes Neurais Convolucionais Profundas capaz de detectar e reconhecer faces e veículos automotivos.

Herdando das redes de classificação de objetos como AlexNet, Deepface (TAIGMAN et al., 2014) e DeepID (SUN et al., 2014), Redes Neurais Discriminativas adotaram a perda *softmax* baseada em *cross-entropy* para aprendizado das características. Depois disso, os pesquisadores perceberam que a perda *softmax* não é suficiente por si só para aprender características discriminativas, e mais pesquisadores começaram a explorar novas funções de perda para aumentar a capacidade de generalização, fazendo com que este tópico de

pesquisa se tornasse o mais visado na área de modelos profundos de reconhecimento facial (MEI; DENG, 2018).

Descobriu-se então que funções de perda baseadas em métricas discriminativas têm melhor capacidade de generalização (CHENG et al., 2018), não apenas para problemas de verificação (LIU et al., 2017) (WEINBERGER et al., 2006), mas também para aprendizado de *few-shot learning* (KOCH et al., 2015) (MEDELA et al., 2019) superando a capacidade de aprendizado das abordagens de classificação tradicionais em situações de pouco número de imagens de treino.

Os modelos que fazem uso das funções de perda discriminativas aprendem a gerar *embeddings* de cada face de entrada, ou seja, um vetor que contém as características mais importantes associadas à imagem. Para isso, normalmente são utilizadas funções de perda baseadas em distância euclidiana, pois restringem conceitualmente os *embeddings* aprendidos a terem distâncias próximas a zero entre elementos da mesma classe e distâncias maiores entre elementos de classes diferentes.

O *Contrastive Loss* (CHOPRA et al., 2005) é um eficaz exemplo destas funções que compara as amostras em pares (positivo e negativo), o qual foi utilizado juntamente às Redes Neurais Siamesas (BROMLEY et al., 1993) e obteve resultados significativos. Isso foi estendido pelo *Triplet Loss* (SCHROFF et al., 2015) ao utilizar comparação de trigêmeos (positivo, negativo e âncora), visto que a cada passo de treinamento, a rede aprendia a distanciar o negativo do âncora e aproximar o positivo do âncora. Enquanto isso, a função *Multiclass-N-Pair Loss* (SOHN, 2016) se concentrou em melhorar as funções de perda anteriores, ao generalizar o *Triplet Loss* comparando a âncora com todas as classes negativas presentes no conjunto de imagens de treinamento. Buscando melhorar ainda mais a separabilidade entre as classes de treinamento, Medela e Picón (2019) criaram o *Constellation Loss*, que visa o distanciamento das classes negativas não apenas em relação a âncora, mas em relação a elas próprias. Os autores alcançaram melhores resultados com o *Constellation Loss* do que as outras funções já consolidadas na área, ao combinar a função com a arquitetura *InceptionV3* (SZEGEDY et al., 2015b) para classificar diferentes tipos de tumores presentes no conjunto de dados criado por Kather et al. (2016).

A Tabela 3 mostra de forma cronológica trabalhos conceituados na área de reconhecimento facial que fizeram uso da metodologia das Redes Neurais Siamesas junto às Redes Neurais Convolucionais, com suas respectivas acurácias na base de dados LFW. A tabela é construída detalhando o nome do método de treinamento, o ano em que o método foi publicado, a função de perda utilizada na etapa de extração de características, o banco de dados junto à quantidade de imagens separadas para treinamento e teste, e a acurácia na etapa de classificação alcançada pelo método ao testar com a base de dados LFW. Ao analisar este conjunto de informações, nota-se o porquê da metodologia das Redes Neurais Siamesas ser considerada o estado da arte nesta área.

Tabela 3 – A acurácia de diferentes métodos avaliados no conjunto de dados LFW

Método	**Ano**	**Função de Perda**	**Arquitetura**	**Treino**	**Acurácia(%)**
DeepFace	2014	softmax	Alexnet	Facebook (4.4M,4K)	97,35±0,25
DeepID2	2014	contrastive loss	Alexnet	CelebFaces+ (0.2M,10K)	99,15±0,13
DeepID3	2015	contrastive loss	VGGNet-10	CelebFaces+ (0.2M,10K)	99,53±0,10
FaceNet	2015	triplet loss	GoogLeNet-24	Google (500M,10M)	99,63±0,09
Baidu	2015	triplet loss	CNN-9	Baidu (1.2M,18K)	99,77
VGGface	2015	triplet loss	VGGNet-16	VGGface (2.6M,2.6K)	98,95
light-CNN	2015	softmax	light CNN	MS-Celeb-1M (8.4M,100K)	98,8
Center Loss	2016	center loss	Lenet+-7	CASIA-WebFace, CACD2000, Celebrity+ (0.7M,17K)	99,28
L-softmax	2016	L-softmax	VGGNet-18	CASIA-WebFace (0.49M,10K)	98,71
Range Loss	2016	range loss	VGGNet-16	MS-Celeb-1M, CASIA-WebFace (5M,100K)	99,52
L2-softmax	2017	L2-softmax	ResNet-101	MS-Celeb-1M (3.7M,58K)	99,78
Normface	2017	contrastive loss	ResNet-28	CASIA-WebFace (0.49M,10K)	99,19
CoCo loss	2017	CoCo loss	-	MS-Celeb-1M (3M,80K)	99,86
vMF loss	2017	vMF loss	ResNet-27	MS-Celeb-1M (4.6M,60K)	99,58
Marginal Loss	2017	marginal loss	ResNet-27	MS-Celeb-1M (4M,80K)	99,48
SphereFace	2017	A-softmax	ResNet-64	CASIA-WebFace (0.49M,10K)	99,42
CCL	2018	center invariant loss	ResNet-27	CASIA-WebFace (0.49M,10K)	99,12
AMS loss	2018	AMS loss	ResNet-20	CASIA-WebFace (0.49M,10K)	99,12
Cosface	2018	cosface	ResNet-64	CASIA-WebFace (0.49M,10K)	99,33
Arcface	2018	arcface	ResNet-100	MS-Celeb-1M (3.8M,85K)	99,83
Ring loss	2018	Ring loss	ResNet-64	MS-Celeb-1M (3.5M,31K)	99,50

Fonte: Tradução do autor de tabela retirada de Mei e Deng (2018)

3 Técnicas

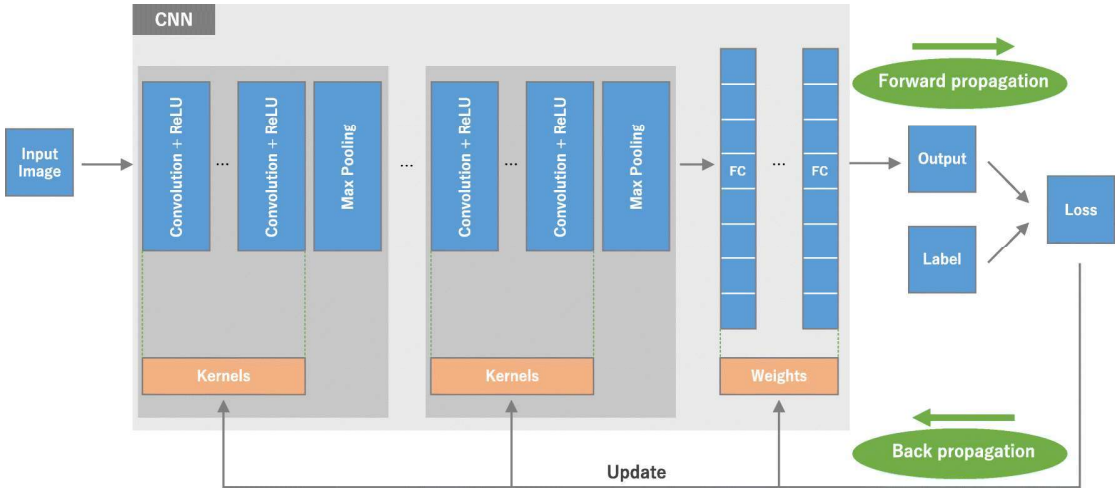
O conceito de *Deep Learning* é o atual estado da arte na área de inteligência artificial, principalmente na área de visão computacional. Unindo os conceitos das Redes Neurais Convolucionais Profundas e as Redes Neurais Siamesas, os atuais modelos de reconhecimento facial alcançam resultados significativos, intensificando, assim, o estudo sobre a função de perda a ser utilizada no modelo. Neste trabalho são utilizadas ambas as técnicas, em conjunto com a inovadora função de perda *Constellation Loss*.

3.1 CNN

Uma Rede Neural Convolucional, ou *Convolutional Neural Network* (CNN), é uma variação das Redes Perceptron de Múltiplas Camadas. Sendo utilizada principalmente na área de reconhecimento de padrões em imagens, este tipo de rede neural foi construída se baseando no córtex visual de animais, tendo em vista que ao receber um estímulo visual na retina, a imagem é reconhecida devido a regiões do cérebro que identificam características específicas do sinal passado (HUBEL; WIESEL, 1959). A arquitetura de uma CNN foi projetada para aproveitar a estrutura 2D do dado de entrada. Isto é realizado com conexões locais e pesos ligados, seguidos de alguma forma de agrupamento que resulta em características invariantes de translação. Um benefício válido de se destacar deste tipo de arquitetura, é o fato de possuir um treinamento mais fácil e rápido, tendo uma quantidade consideravelmente menor de parâmetros do que redes totalmente conectadas com o mesmo número de unidades escondidas.

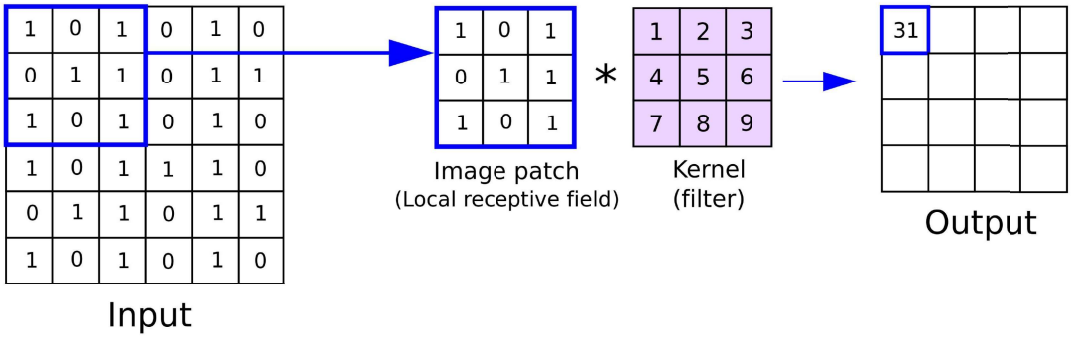
Como pode ser observado na Figura 2, camadas de convolução são aplicadas na imagem de entrada. Estas camadas são compostas de neurônios que estão conectados a um conjunto de *pixels* da imagem, tendo cada conexão um peso diferente. Os neurônios aplicam *kernels* (filtros) sobre cada pixel da imagem considerando uma vizinhança limitada ao redor do pixel (campo receptivo), produzindo uma saída para cada pixel que é passada para a próxima camada. Uma imagem pode ter milhões ou milhares de *pixels*, mas ao processá-la usando o *kernel*, podemos detectar informações locais significativas de dezenas ou centenas de *pixels*. Isso significa que precisamos armazenar menos parâmetros, o que não apenas reduz o requisito de memória do modelo, mas também melhora a eficiência estatística do modelo. A Figura 3 ilustra a aplicação de um *kernel* em um campo receptivo. Esta abordagem é diferente das redes neurais clássicas, em que cada neurônio é conectado a todos os pixels da imagem. O conjunto de saídas de uma mesma camada é chamado de mapa de características. Após as camadas de convolução, é comum utilizar uma camada de agregação (*pooling*) que tem como função a redução das dimensões da entrada (altura

Figura 2 – Uma visão geral da arquitetura de uma CNN e do processo de treinamento. Uma CNN é composta por: camadas de convolução, camadas de *pooling* (por exemplo, *max pooling*) e camadas totalmente conectadas (do inglês *Fully Connected*, FC). O desempenho do modelo é calculado com uma função de perda por meio de propagação direta em um conjunto de dados de treinamento, sendo os pesos atualizados de acordo com o valor da perda por meio de retro-propagação com o algoritmo de otimização de gradiente descendente.



Fonte: (YAMASHITA et al., 2018)

Figura 3 – Aplicação do *kernel* em uma camada



Fonte: (REYNOLDS, 2019)

e largura), tendo, assim, uma velocidade de treinamento maior.

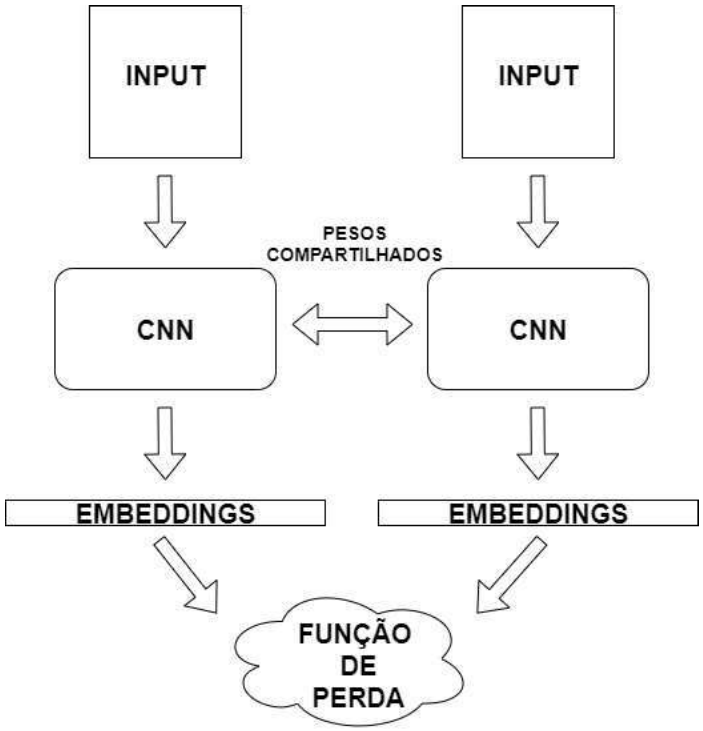
Ao final da camada de extração de características, é típico a arquitetura possuir uma camada de classificação visando rotular a resposta da rede se baseando nas saídas dos *kernels*. Utilizar uma camada totalmente conectada resulta em menor custo computacional para aprender combinações não-lineares das características extraídas da imagem.

As CNN se tornaram mais eficazes com o avanço da tecnologia, tendo grandes resultados na área de visão computacional, principalmente com a disseminação da metodologia de *Deep Learning*, onde CNN com grande número de convoluções se tornaram aptas a serem treinadas e aplicadas. Chamadas de redes neurais convolucionais profundas, estas redes se tornaram capazes de serem utilizadas tanto na detecção de objetos, como no reconhecimento destes.

3.2 Redes Neurais Siamesas

Introduzidas pela primeira vez por Bromley et al. (1993), as Redes Neurais Siamesas (RNS) foram empregadas em um sistema de verificação de assinaturas. Uma RNS é uma classe de arquiteturas de rede neural que contém duas ou mais sub-redes com mesma configuração, mesmos parâmetros e mesmos pesos, como mostra a Figura 4. A atualização de parâmetros é espelhada em ambas as sub-redes, sendo usada para encontrar o grau de similaridade das entradas comparando seus vetores de características (*embeddings*) que são extraídas pela rede.

Figura 4 – Configuração do Modelo RNS



Fonte: Bromley et al. (1993)

Tradicionalmente, uma rede neural aprende a prever várias classes. Isso representa um problema quando é necessário adicionar e/ou remover novas classes à rede. Nesse caso,

a rede neural tem que ser atualizada e treinada novamente com todo o conjunto de dados. A RNS, por outro lado, aprende uma função de similaridade. Assim, pode-se treiná-la verificando se duas imagens pertencem à mesma classe ou não, possibilitando classificar novas classes de dados sem treinar a rede toda novamente.

Assim, uma RNS recebe como entrada um par de entradas com o objetivo de desenvolver similaridade entre pares de uma mesma classe, e distanciar pares de diferentes classes. Com o treinamento, o modelo cria um espaço multi-dimensional igual ao número de neurônios de saída da rede neural base. A representação espacial de cada entrada então é submetida a uma função de similaridade (ou distância), normalmente sendo a distância euclidiana. Uma vez obtida a medida de similaridade, é definido se os dados de entrada pertencem ou não à uma mesma classe dado um limiar definido.

Sobre essa medida de similaridade (ou distância), são aplicadas as funções de perda, as quais são as responsáveis por informar à rede como deve ser feito o aprendizado.

3.3 Funções de Perda

Com o sucesso da metodologia implementada por Schroff et al. (2015), os estudos se intensificaram em torno da aprendizagem de métricas de distância utilizando os *embeddings* gerados pelas RNS. Atualmente, as funções de perda baseadas na distância euclidiana possuem resultados expressivos e, consequentemente, tornaram-se comuns. Nesta seção, são detalhadas as funções de perda consideradas estado da arte nesta área.

3.3.1 *Contrastive Loss*

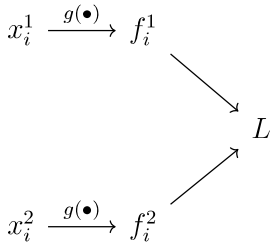
No caso da *Contrastive Loss* a estrutura da RNS vem definida pelo diagrama ilustrado na Figura 5, onde o par de imagens de entrada (x_i^1, x_i^2), são transformados, via uma rede CNN $g(\bullet)$, nos *embeddings* (f_i^1, f_i^2). O *Contrastive Loss* calcula o somatório das perdas individuais dos pares positivos (dados de mesma classe) e pares negativos (dados de classes distintas).

Um par positivo i possui rótulo $y_i = 1$, enquanto um par negativo recebe rótulo $y_i = 0$. A equação desta função é dada por (3.1).

$$L = \frac{1}{2B}\sum_{i=1}^{B}\{(1-y_i)||f_i^1 - f_i^2||_2^2 + (y_i)[max(m - ||f_i^1 - f_i^2||_2^2, 0)]^2\} \tag{3.1}$$

onde m é a margem, normalmente configurada igual a 1, e B é o tamanho do *batch* utilizado para o treinamento da rede.

Figura 5 – Estrutura da RNS ao aplicar *Contrastive Loss*

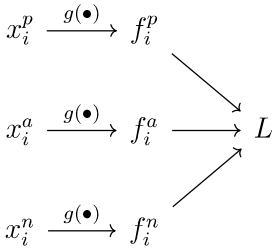


Fonte: (SANTOS et al., 2021)

3.3.2 *Triplet Loss*

Desde sua primeira citação, o *Triplet Loss* tornou-se a função de perda mais popular na área do aprendizado métrico. O método alcançou bons resultados em diversos trabalhos, principalmente ao usar esta perda no treinamento de RNS, podendo ser visto em (SCHROFF et al., 2015). No caso do *Triplet Loss*, a estrutura da RNS é definida pela Figura 6.

Figura 6 – Estrutura da RNS ao aplicar *Triplet Loss*



Fonte: (SANTOS et al., 2021)

Expandindo a metodologia do *Contrastive Loss*, o *Triplet Loss* recebe três imagens de entrada (x_i^a, x_i^p, x_i^n), sendo a âncora x_i^a, uma imagem de uma das classes do conjunto de dados de treinamento, x_i^p uma imagem da mesma classe da âncora chamada de positivo e a imagem x_i^n chamada de negativo, que é um exemplo de uma classe diferente da âncora. O método tem como finalidade maximizar a distância da âncora em relação ao negativo, ao passo que minimiza a distância para o positivo, tornando o modelo extrator de características capaz de gerar uma separação melhor dos *embeddings* de classes distintas.

Deste modo, a *Triplet Loss* recebe um trio de *embeddings* (f_i^a, f_i^p, f_i^n), formados pelos *embeddings* da âncora (f_i^a), da amostra positiva (f_i^p) e da amostra negativa (f_i^n). A

equação desta função é dada em (3.2).

$$L = \frac{1}{B}\sum_{i=1}^{B} max(||f_i^a - f_i^p||_2^2 - ||f_i^a - f_i^n||_2^2 + \alpha, 0) \tag{3.2}$$

onde B é o tamanho do *batch* de treinamento e α é uma constante chamada de margem de distância, utilizada para especificar quando um trio se tornou muito "fácil" e não é desejado mais ajustar os pesos dele.

Com base nessa definição, existem três categorias de *triplets*:

- *Triplets* fáceis: *triplets* que têm uma perda igual a 0, pois $(||f_i^a - f_i^p||_2^2 + \alpha) < ||f_i^a - f_i^n||_2^2$
- *Triplets* difíceis: *triplets* onde o negativo está mais próximo que o positivo, ou seja, $||f_i^a - f_i^p||_2^2 > ||f_i^a - f_i^n||_2^2$
- *Triplets* semi-difíceis: *triplets* onde o negativo não está mais próximo que o positivo, mas ainda possui uma perda positiva, ou seja, $||f_i^a - f_i^p||_2^2 < ||f_i^a - f_i^n||_2^2 < (||f_i^a - f_i^p||_2^2 + \alpha)$

A Figura 7 ilustra as distâncias da âncora para o negativo, conforme as categorias abordadas anteriormente. A figura ilustra as posições da âncora e da amostra positiva.

Figura 7 – Regiões de distanciamento entre a âncora e amostras negativas, em relação à amostra positiva

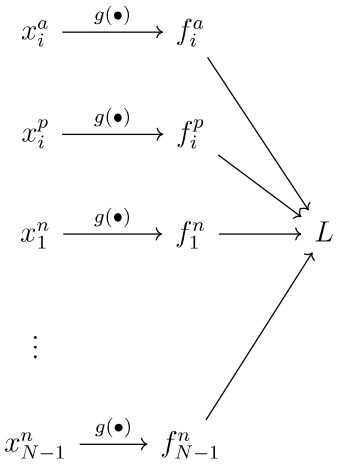
Fonte: (SANTOS et al., 2021)

3.3.3 *Multiclass-N-Pair Loss*

Surgindo como uma extensão do *Triplet Loss* e com o foco de vencer a lenta convergência de cada passo de aprendizagem, Sohn (2016) propôs a função de perda

Multiclass-N-Pair Loss, o qual utiliza o conceito de comparação com mais negativos, mais precisamente, $N-1$ negativos, sendo N o número de classes presentes no conjunto de dados de treinamento. Ou seja, a cada iteração de aprendizagem, o cálculo da perda se baseia nos *embeddings* da âncora (x_i^a), do exemplo positivo (x_i^p) e de um exemplo negativo de cada uma das demais classes (x_j^n, $j = \{1, \cdots, N-1\}$), ou seja, deve-se assegurar que $x_i^p \neq x_j^n$. Nesse contexto, a arquitetura da RNS toma a forma apresentada na Figura 8.

Figura 8 – Estrutura da RNS ao aplicar *Multiclass-N-Pair Loss*



Fonte: (SANTOS et al., 2021)

O *Multiclass-N-Pair Loss* recebe uma N-tupla de *embeddings* aplicando a equação descrita em (3.3):

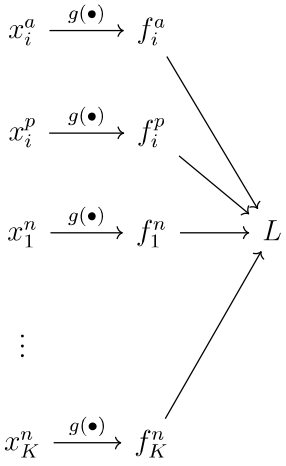
$$L = \frac{1}{B}\sum_{i=1}^{B} \log\left(1 + \sum_{j=1}^{N-1} \exp\left(f_i^{aT} f_j^n - f_i^{aT} f_i^p\right)\right) \tag{3.3}$$

onde B é o tamanho do *batch* de treinamento, que no caso do *Multiclass-N-Pair Loss* é o número de diferentes classes presentes no conjunto de dados de treinamento.

3.3.4 *Constellation Loss*

Unindo a mesma construção de *batches* de treinamento da *Triplet Loss* e uma formulação de perda semelhante à da *Multiclass-N-Pair Loss*, a função de perda *Constellation Loss* busca aproveitar o que tem de melhor nos conceitos dessas funções, como é descrito por Medela e Picón (2019). O diagrama ilustrado na Figura 9 apresenta a arquitetura RNS do *Constellation Loss*, podendo notar a semelhança com a arquitetura usada para *Multiclass-N-Pair Loss*.

Figura 9 – Estrutura da RNS ao aplicar *Constellation Loss*



Fonte: (SANTOS et al., 2021)

A principal diferença do *Constellation Loss* é que enquanto o *Multiclass-N-Pair Loss* subtrai produtos escalares de pares da mesma classe, o *Constellation Loss* faz algo semelhante ao *Triplet Loss*, onde a perda é calculada pela subtração do produto escalar dos *embeddings* de uma âncora e *embeddings* negativos; e o produto escalar dos *embeddings* de uma âncora e *embeddings* positivos. A equação é retratada em (3.4), podendo notar sua similaridade com a Equação (3.3):

$$L = \frac{1}{B}\sum_{i=1}^{B}\log\left(1+\sum_{j=1}^{K}\exp\left(f_i^{aT}f_j^n - f_i^{aT}f_i^p\right)\right) \tag{3.4}$$

onde B é o tamanho do *batch* de treinamento e K é a quantidade de *triplets* que se quer incorporar na função de perda a cada iteração.

Ao passo que a *Triplet Loss* aproxima o exemplo positivo e afasta apenas uma classe negativa, a *Multiclass-N-Pair Loss* afasta todas as classes negativas da âncora. A *Constellation Loss* tem como foco o afastamento dos negativos não apenas em relação à âncora, mas entre os próprios negativos. Visando não utilizar um alto valor no parâmetro K, devido ao significante aumento do custo computacional no treinamento do modelo, os exemplos presentes nos *batches* de treinamento são escolhidos de forma aleatória, ou seja, aumentar o número de exemplos negativos do *batch* não garante que o número de valores negativos distintos irá aumentar. Assim, não existe a necessidade de um valor elevado para K para alcançar uma melhora em relação às funções de perda *Triplet Loss* e *Multiclass-N-Pair Loss*.

Devido à natureza da função considerar a relação entre todas as amostras, não se faz

necessário a aplicação da margem α para encontrar *triplets* difíceis e semi-difíceis. Apenas precisa calcular os produtos escalares como a função *Triplet Loss* faz, e aplicar uma lógica para encontrar os produtos escalares entre *embeddings* da mesma classe (âncora-positiva) e os produtos escalares entre *embeddings* de classes distintas (âncora-negativa).

4 Bases de Dados e Método Proposto

4.1 Bases de Dados e Preparação dos Dados

No intuito de verificar o comportamento dos modelos diante da classificação de faces, foram utilizados dois conjuntos de dados com uma importante característica que os diferem: enquanto o conjunto *Olivetti Faces* possui o mesmo número de faces por pessoa, o conjunto LFW possui números de faces que diferem de pessoa para pessoa. As características de cada base e a preparação dos dados estão descritos abaixo.

4.1.1 Bases de Dados

O conjunto *Olivetti Faces*, utilizado em (DHIFLI; DIALLO, 2016), possui 400 imagens de faces, coletadas entre os anos de 1992 e 1994, sendo 10 exemplos para cada um dos 40 indivíduos do *AT&T Laboratories Cambridge*. Para alguns sujeitos, as imagens foram tiradas em momentos diferentes, variando a iluminação, expressões faciais (olhos abertos / fechados, sorrindo / não sorrindo) e detalhes faciais (óculos / sem óculos). Todas as imagens foram tiradas contra um fundo escuro homogêneo com os indivíduos em uma posição frontal vertical (com tolerância para alguns movimentos laterais), como pode ser observado nos exemplos da Figura 10.

Figura 10 – Imagens presentes no conjunto de dados *Olivetti Faces*

Fonte: Dhifli e Diallo (2016)

O conjunto LFW (*Labeled Faces in the Wild*), criado em (HUANG et al., 2008), contém mais de 13.000 imagens de rostos coletados da web, extraídos com o uso do detector facial *Viola-Jones* (BANDALA et al., 2013). Cada rosto foi etiquetado com o nome da pessoa retratada, levando em consideração o fato de que 1.680 pessoas retratadas possuírem duas ou mais fotos distintas. Sendo utilizado em trabalhos importantes, como em (SCHROFF et al., 2015), tornou-se uma base comum para validações de sistemas de reconhecimento facial. Alguns exemplos de faces existentes no conjunto de dados LFW podem ser vistos na Figura 11.

Figura 11 – Imagens presentes no conjunto de dados LFW

Fonte: Huang et al. (2008)

Como a principal motivação do presente trabalho é propor um modelo extrator de características, foram escolhidos conjuntos de dados com imagens já tratadas, eliminando a necessidade de um detector facial e/ou um pré-tratamento. Assim, tornou-se necessário apenas a normalização dos dados e o redimensionamento para o tamanho desejado, sendo $224 \times 224 \times 3$ o tamanho das imagens usadas no desenvolvimento dos modelos.

4.2 Método Proposto

A arquitetura proposta neste trabalho consiste de uma RNS que usa como *backbone* CNN pré-treinadas, as quais geram os *embeddings* relacionados a cada face. Na etapa de treinamento foi usada a função *Constellation Loss*, e uma metodologia particular para a formação dos *batches* de treinamento, proposta por Medela e Picón (2019). Para via de comparação com funções de perda consideradas estado da arte, o modelo também será treinado com o *Triplet Loss* e o *Multiclass-N-Pais Loss*. Tanta a metodologia de adequação

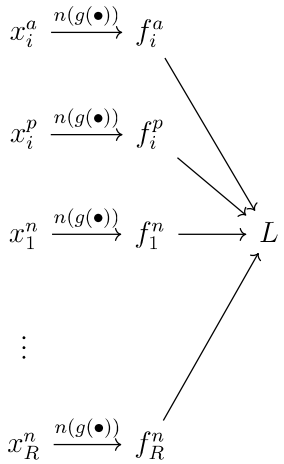
das redes *backbone* CNN como a forma da composição dos *batches* serão explicadas a seguir.

4.2.1 *Backbones*

No presente trabalho foram utilizadas redes CNN conceituadas na área de classificação de imagens, como a *ResNet152V2* (HE et al., 2016b) e a *InceptionV3* (SZEGEDY et al., 2015b). A rede *ResNet152V2* foi escolhida tendo em vista seu desempenho no desafio ImageNet, enquanto que a *InceptionV3* foi a rede utilizada em (MEDELA; PICÓN, 2019), obtendo resultados significativos na classificação de imagens ao combinar com a função de perda *Constellation Loss*.

Ao remover a última camada de cada CNN para que a arquitetura se adapte à classificação do conjunto de dados usado para treino, foi adicionada uma camada de *average pooling*, como sugerem Schroff et al. (2015), seguida de uma camada com 128 unidades de saída, com função de ativação sigmoide. No intuito de manter o processo mais estável e mais apto para ser definida a margem entre âncoras e positivos, quando utilizadas as funções de perda *Triplet Loss* e *Constellation Loss*, os *embeddings* de saída da rede foram normalizados, via norma $L2$. Nesse contexto, o diagrama da arquitetura RNS implementada possui a forma apresentada na Figura 12.

Figura 12 – Estrutura da RNS implementada.



Fonte: (SANTOS et al., 2021)

Neste caso, $g(\bullet)$ representa a CNN pré-treinada com as camadas adicionadas, $n(\bullet)$ representa a operação de normalização $L2$ que é aplicada na saída da rede. $R = 1$ para a *Triplet Loss*, $R = N - 1$ para a *Multiclass-N-Pair Loss* e $R = K$ para a *Constellation Loss*.

4.2.2 Metodologia para Construção dos *Batches*

Para o caso de uma arquitetura RNS, o processo de aprendizado será resumido a seguir. Seja o conjunto de treinamento $\mathcal{T} = \{(x_i^a, x_i^p, \{x_j^n\}_{j=1}^R)\}_{i=1}^T$, formado por T tuplas. Então, usando $\mathcal{T}$, a função de perda L, a quantidade definida de exemplos negativos R e o algoritmo de optimização SGD (*Stocastic Gradient Descent*), os parâmetros da rede $g(\bullet)$ são ajustados a partir de seus valores de pré-treinamento.

Nesse contexto, em cada iteração do SGD, as tuplas do conjunto de treinamento são passados à RNS em *batches*, ou seja, $\mathcal{T}$ é dividido em sub-conjuntos, cada um contendo B tuplas. Se cada *batch* contém uma única tupla ($B = 1$), o processo de treinamento é definido como um aprendizado *online* e se cada *batch* contém mais de uma tupla ($B > 1$), é definido como um aprendizado *offline*.

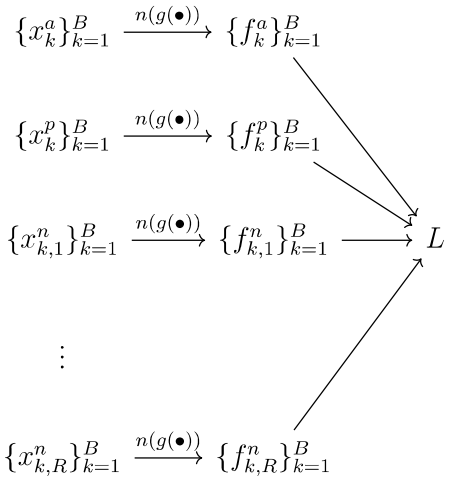
Cada tipo de aprendizado tem seus benefícios e desvantagens. Por exemplo, o aprendizado *online* permite efetuar um treinamento eficiente de um modelo neural, mas fica dependente da natureza estocástica do algoritmo SGD, enquanto que o aprendizado *offline*, ao considerar que cada *batch* contém várias tuplas, fica menos dependente de estocasticidade do SGD, mas requer que os modelos de aprendizado com as funções de perda sejam implementadas de forma vetorizada, para que, em uma única iteração do SGD, se possa processar todo o *batch*, supondo que o processamento computacional será feito em uma GPU (*Graphics Processing Unit*). Então, aqui surge um inconveniente no treinamento de uma RNS, já que tal arquitetura é uma replicação de uma única estrutura vetorizada, representada pela operação $x \xrightarrow{n(g(\bullet))} f$ da RNS mostrada no diagrama da Figura 12. Por exemplo, se for considerado o *batch* $\mathcal{X} = \{(x_k^a, x_k^p, \{x_{k,j}^n\}_{j=1}^R)\}_{k=1}^B$ constituído de B tuplas, ele será aplicado de forma vetorizada à RNS tal como ilustrado na Figura 13.

Então, ao usar um aprendizado *offline* será necessário aplicar os subconjuntos de exemplos $\{x_k^a\}_{k=1}^B$, $\{x_k^p\}_{k=1}^B$ e $\{\{x_{k,j}^n\}_{k=1}^B\}_{j=1}^R$ em cada uma das operações vetorizadas que será apresentada a uma RNS, o que gera um problema de escalabilidade do modelo, considerando que o R é um hiperparâmetro dele.

No caso de usar aprendizado *online*, o *batch* contém uma única tupla, ou seja, $\mathcal{X} = \{(x_1^a, x_1^p, \{x_{1,j}^n\}_{j=1}^R)\}$, implicando que unicamente se tenha que aplicar um único exemplo em cada operação vetorizada da RNS, o que permite a equivalência mostrada na Figura 14, onde $\mathcal{F} = \{(f_1^a, f_1^p, \{f_{1,j}^n\}_{j=1}^R)\}$ é o *batch* dos *embeddings* que serão usados pela função de perda escolhida.

Tomando em conta a equivalência indicada na Figura 14, é possível efetuar um aprendizado *online* com vários *batches*, mas para isso deve-se construir um único *batch* que contenha todos os *batches* a usar. Nesse contexto, aqui é definida a operação *concat*$(\bullet)$, que efetua a concatenação dos *batches* respeitando a ordem que a função de perda espera

Figura 13 – Aplicação de um *batch* com B tuplas, considerando que $B > 1$, na estrutura de RNS implementada.



Fonte: (SANTOS et al., 2021)

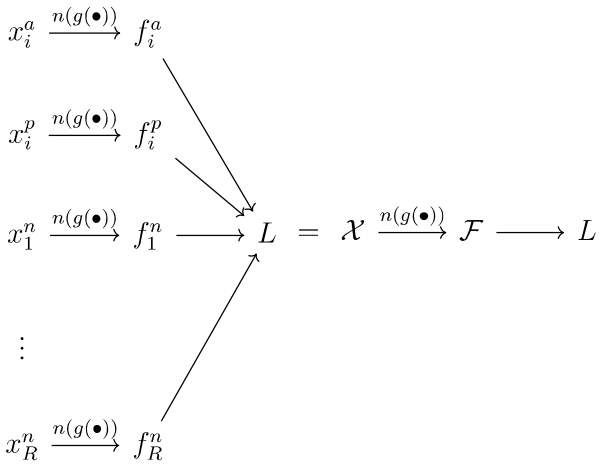
em relação aos *embeddings*. Matematicamente:

$$\begin{aligned} \mathcal{X}_{ct} &= concat(\{(x_k^a, x_k^p, \{x_{k,j}^n\}_{j=1}^R)\}_{k=1}^B) \\ &= \{(x_1^a, x_1^p, \{x_{1,j}^n\}_{j=1}^R, \cdots, x_B^a, x_B^p, \{x_{B,j}^n\}_{j=1}^R)\}, \end{aligned}$$

Então, ao aplicar o $\mathcal{X}_{ct}$ à RNS, devido à relação de equivalência indicada na Figura 14, a tupla de *embeddings* calculada será: $\mathcal{F}_{ct} = \{(f_1^a, f_1^p, \{f_{1,j}^n\}_{j=1}^R, \cdots, f_B^a, f_B^p, \{f_{B,j}^n\}_{j=1}^R)\}$, que ao ser aplicada a L, deverá retornar a perda por *batch*, como indicado nas Equações (3.2), (3.3) e (3.4).

Com essa possibilidade de manipulação dos dados a cada *batch* disponibilizada para a rede, pode-se contornar o problema da escalabilidade das RNS quando operam com funções de perda que trabalham com três ou mais valores de entrada, como os métodos validados neste trabalho. Tal metodologia, proposta por Medela e Picón (2019), foi aplicada no presente trabalho.

Figura 14 – Relação de equivalência entre a estrutura de RNS implementada e uma única operação vetorizada, quando é aplicado um *batch* que contém só uma tupla ($B = 1$).



Fonte: (SANTOS et al., 2021)

5 Experimentos e Resultados

5.1 Métricas

Tendo em vista que o presente trabalho possui foco na extração de características, as métricas para avaliação dos modelos foram escolhidas no intuito de certificar a separabilidade entre as classes de treinamento, uma vez que o desejado é ter uma boa separação interclasse e distribuição intraclasse mais compacta.

Podendo ser utilizada para problemas de classificação binária e multi-classes, a matriz de confusão, ilustrada na Tabela 4, será a base para algumas das métricas utilizadas neste trabalho. O conceito trabalha com quatro combinações diferentes entre valores previstos e reais:

- Verdadeiros Positivos (*True Positives*, TP): Quando o modelo classifica corretamente as amostras da classe positiva.
- Falsos Negativos (*False Negatives*, FN): Quando o modelo classifica incorretamente as amostras da classe positiva como negativas.
- Falsos Positivos (*False Positives*, FP): Quando o modelo classifica incorretamente as amostras da classe negativa como positivas.
- Verdadeiros Negativos (*True Negatives*, TN): Quando o modelo classifica corretamente as amostras da classe negativa.

Tabela 4 – Matriz de confusão 2×2

		Valores Preditos	
		Positivo	Negativo
Valores Reais	Positivo	Verdadeiros Positivos (TP)	Falsos Negativos (FN)
	Negativo	Falsos Positivos (FP)	Verdadeiros Negativos (TN)

O entendimento deste conceito é essencial para métricas conhecidas na área de classificação como Sensibilidade (*Recall*), Precisão, Especificidade, Acurácia e, mais importante, a Curva AUC (*Area Under The Curve*) ROC (*Receiver Operating Characteristics*). Para uma análise visual dos *embeddings* obtidos pelos modelos e a separação entre os

mesmos, será utilizada a técnica t-SNE (t-distributed Stochastic Neighbor Embedding) (MAATEN; HINTON, 2008).

5.1.1 AUC-ROC

O conceito AUC-ROC é frequentemente usado para avaliar o desempenho de algoritmos de classificação binária. Enquanto a curva ROC consiste em uma representação gráfica do desempenho de um modelo de classificação por meio da relação da Taxa de Verdadeiro Positivo (Sensibilidade) e da Taxa de Falso Positivo (1-Especificidade), a AUC é a área abaixo da curva ROC, sendo uma medida de separabilidade entre as classes.

A curva ROC é produzida calculando e plotando a Taxa de Verdadeiros Positivos (do inglês *True Positive Rate*, TPR) em relação à Taxa de Falsos Positivos (do inglês *False Positive Rate*, FPR) para um único classificador em uma variedade de limites. Por exemplo, na regressão logística, o limiar seria a probabilidade prevista de uma observação pertencer ou não à classe positiva. A TPR é representada pela Equação 5.1, enquanto a FPR é representada pela Equação 5.2.

$$TPR = Sensibilidade = \frac{TP}{TP + FN} \tag{5.1}$$

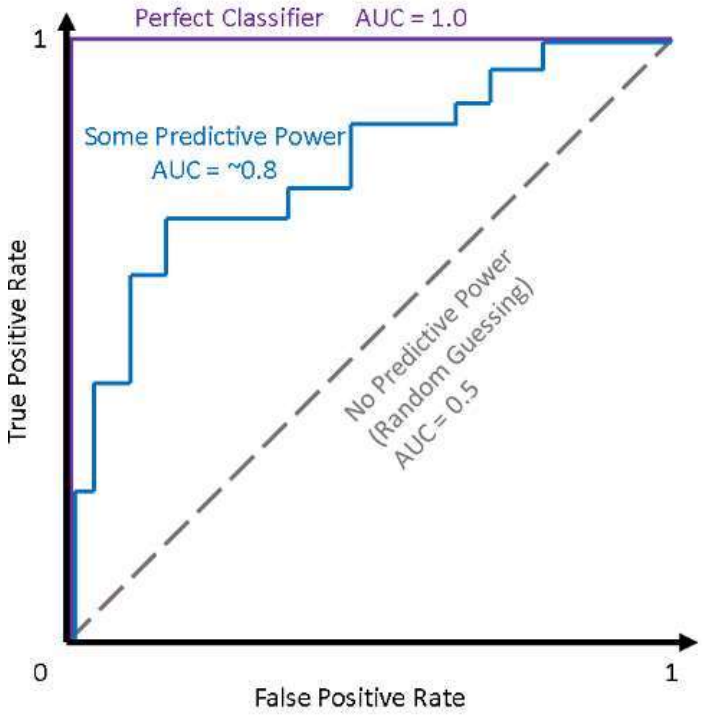
$$FPR = 1 - Especificidade = \frac{FP}{FP + TN} \tag{5.2}$$

A AUC é calculada como uma área embaixo da curva ROC que varia de 0 a 1, tendo sua interpretação como uma probabilidade. Assim, uma AUC próxima de 1 indica que o modelo consegue separar bem as classes, enquanto que próximo de $0,5$ indica que o modelo não tem capacidade discriminativa de distinguir entre as classes positivas e negativas. Uma vantagem de se usar a AUC-ROC neste caso é a funcionalidade de determinação do limiar (*threshold*) que realiza a separação entre as classes. Tendo em vista que: (*i*) um limiar igual a 0% indica que todas as amostras são preditas como positivas, logo, todas as amostras positivas são classificadas corretamente e nenhum verdadeiro negativo e falso negativo é obtido; (*ii*) um limiar igual a 100% indica que todas as amostras são preditas como negativas e não há verdadeiro positivo, nem falso positivo. A Figura 15 ilustra alguns exemplos de curva ROC com seus respectivos valores de AUC, onde o resultado para o limiar de 0% é mostrado no canto superior direito, coordenadas (1,1), e o resultado para o limiar de 100% é indicado no canto inferior esquerdo, coordenadas (0,0).

5.1.2 t-SNE

O método t-SNE é um método de redução de dimensionalidade utilizado principalmente para visualização de dados em gráficos 2D e 3D. Entre algumas das suas utilidades,

Figura 15 – Curvas ROC teóricas com valores de AUC



Fonte: (STEEN, 2020)

o algoritmo t-SNE torna possível visualizar em um plano cartesiano o quão eficaz um modelo foi capaz de separar as classes presentes em um conjunto de dados de teste. A construção do algoritmo pode ser resumida em três etapas:

- Calcular uma distribuição de probabilidade conjunta que representa as semelhanças entre as amostras do conjunto.
- Criar um conjunto de pontos representantes das amostras originais na dimensão de destino e, em seguida, calcular a distribuição de probabilidade conjunta para estes pontos também.
- Usar gradiente descendente para alterar a localização dos pontos no espaço de baixa dimensão para que a distribuição de probabilidade conjunta deles seja o mais semelhante possível à das amostras originais (que estão em alta dimensão).

5.1.3 Acurácia

Uma das métricas mais simples para avaliação de um modelo é a acurácia. É a razão entre o número de previsões corretas e o número total de previsões feitas para um conjunto de dados.

$$Acurácia = \frac{\text{Total de acertos}}{\text{Total de predições}} = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \tag{5.3}$$

A acurácia é útil quando a classe alvo está bem balanceada, mas não é uma boa escolha com classes desbalanceadas. Por exemplo, um conjunto de dados com duas classes de destino contendo 100 amostras, 98 amostras pertencentes à classe A e 2 amostras pertencentes à classe B, o modelo retornaria 98% de precisão se todas as amostras forem classificadas com A. Devido a este fator, a acurácia não se torna uma boa métrica para a base de dados LFW.

5.2 Recursos

O treinamento do modelo e a aquisição das bases de dados foram feitos com a linguagem de programação *Python*, utilizando a API (*Application Programming Interface*) chamada *Keras*, da plataforma *Tensorflow*. Essa API torna mais prática a construção do modelo, assim como a abordagem de criação de lotes inteligentes de treinamento.

Para os experimentos realizados neste trabalho, foi utilizada uma máquina de configuração mediana como uma forma de confirmar a redução no custo computacional da metodologia proposta. As configurações são:

- Windows 10;
- CPU Intel(R) Core(TM) i7-4790k;
- 16 Gb de memória RAM;
- GPU Nvidia GTX 980 com 6 Gb de memória dedicada.

5.3 Custo Computacional

As Tabelas 5 e 6 contêm o tempo de treinamento dos modelos no intuito de medir o esforço computacional ao realizar o treinamento tradiconal de modelos de RNS e ao usar a construção de *batchs* abordada no presente trabalho. A única função de perda comparada nas tabelas de custo computacional é a *Triplet Loss*, pois ao utilizar a metodologia tradicional para as outras funções de perda, onde a rede é replicada para cada entrada, o computador usado para testes não conseguiu executar o treinamento por falta de memória alocada. Para superar a falta de memória, tentou-se realizar a execução do treinamento diretamente na CPU, entretanto o tempo estimado de treinamento para o *Multiclass-N-Pair Loss* e para o *Constellation Loss* superava 10 dias, inviabilizando a comparação para tantas configurações diferentes.

5.4 Hiperparâmetros

Parâmetros que são definidos antes do treino são chamados de hiperparâmetros, sendo estes, os atributos que controlam o treinamento do modelo. Os hiperparâmetros definidos para o treinamento dos modelos estão listados abaixo.

- Learning rate: 0,001
- Momentum: 0,9
- Função de otimização Adam com Learning Rate de 0,0006
- Épocas: 100

5.5 Resultados

Como citado em tópicos anteriores, as bases de dados *Olivetti Faces* e LFW foram utilizadas nos experimentos do modelo proposto. Para cada base é apresentada uma tabela (Tabelas 8 e 10) com os valores de AUC-ROC para o modelo treinado com dois diferentes *backbones* (ResNet152 V2 e Inception V3) treinados no desafio ImageNet e com as funções de perda *Triplet Loss*, *Multiclass-N-Pair Loss* e *Constellation Loss*, com diferentes valores de K. Também é apresentado para cada base o resultado da técnica t-SNE ao ser aplicada no modelo com o melhor resultado de AUC-ROC para cada base, Figuras 17 e 18. Para treinamento das redes, ambas foram carregadas com os .

Tabela 5 – Média do tempo de treinamento no Conjunto de Dados *Olivetti Faces* utilizando o treinamento tradicional em um modelo RNS e a metodologia de construção de *batches* abordada no trabalho. O treinamento ocorreu sem a utilização da memória da GPU

Função de Perda	**ResNet152 V2**	**Inception V3**
Triplet Loss - Treinamento Tradicional	121000 s	30200 s
Triplet Loss - Construção dos *Batches*	20300 s	5020 s

Tabela 6 – Média do tempo de treinamento no Conjunto de Dados LFW utilizando o treinamento tradicional em um modelo RNS e a metodologia de construção de *batches* abordada no trabalho. O treinamento ocorreu sem a utilização da memória da GPU

Função de Perda	**ResNet152 V2**	**Inception V3**
Triplet Loss - Treinamento Tradicional	364000 s	80900 s
Triplet Loss - Construção dos *Batches*	60700 s	13500 s

Para cada base de dados, foi utilizada uma divisão 70% e 30% para treinamento e teste, respectivamente. O número de imagens para cada conjunto pode ser visto na Tabela 7.

Tabela 7 – Número de imagens separados para treinamento e validação nos conjuntos de dados

	Olivetti Faces	**LFW**
Treinamento	280 imagens	3026 imagens
Teste	120 imagens	1298 imagens

5.5.1 Resultados na base *Olivetti Faces*

A Tabela 8 mostra o valor médio de AUC-ROC após 100 épocas de treinamento. Como pode ser observado, o *Constellation Loss* proporcionou melhores resultados mesmo nas duas diferentes arquiteturas de CNN testadas como *backbone* do modelo.

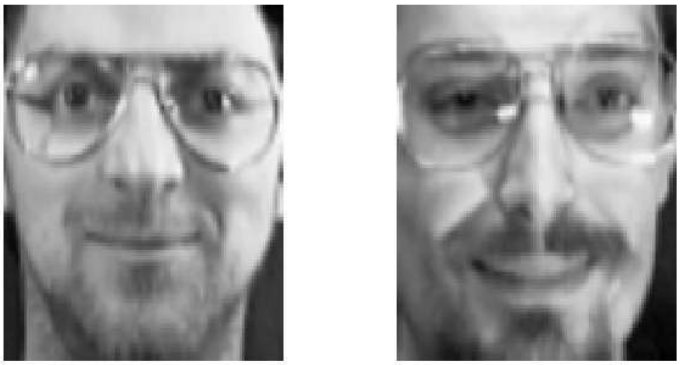
Tabela 8 – Valor de AUC-ROC médio no Conjunto de Dados *Olivetti Faces*

Função de Perda	**ResNet152 V2**	**Inception V3**
Triplet	0,998	0,997
Multiclass N-Pair	0,985	0,987
Constellation $K = 2$	0,994	0,994
Constellation $K = 3$	0,998	0,997
Constellation $K = 4$	**0,999**	**0,999**
Constellation $K = 5$	0,997	0,998
Constellation $K = 6$	0,997	0,996
Constellation $K = 7$	0,998	0,997

Para via de validação do modelo treinado no conjunto *Olivetti*, adicionou-se o classificador *Nearest Neighbor* (NN) ao final da rede extratora de características, o mesmo utilizado em Lu et al. (2021). O modelo obteve $99,17\%$ de acurácia, valor semelhante ao encontrado em Lu et al. (2021), onde o autor alcançou 99% de acurácia após aplicar a técnica de aumento de dados (*data augmentation*) e a mesma divisão de dados utilizada neste trabalho. A Figura 16 mostra o único exemplo na base de testes classificado erroneamente, sendo possível identificar que o par de faces possui um alto nível de semelhança, dificultando sua diferenciação. É válido citar que foi também avaliado o uso da técnica de aumento de dados para a base de dados *Olivetti*, porém a média de acurácia se manteve a mesma. Os valores médios das acurácias alcançadas pelas outras configurações podem ser vistas na Tabela 9, no qual é possível observar que o *Constellation Loss* com $K = 4$ obteve os melhores resultados em ambas as redes.

Devido a uma menor quantidade de classes existentes no conjunto de dados *Olivetti Faces*, o ambiente de aquisição das imagens ser mais controlado e haver uma padronização

Figura 16 – Imagens do conjunto Olivetti rotuladas de maneira equivocada pelo modelo. A face da esquerda é a imagem de entrada do teste e a face da direita pertencente a classe predita pelo modelo



Fonte: O autor

da quantidade de amostras por classe, fica mais nítida a separabilidade que o modelo proporcionou às classes presentes nos dados de teste, como pode ser visualizado pelo mapa 2D do t-SNE apresentado na Figura 17.

5.5.2 Resultados na base LFW

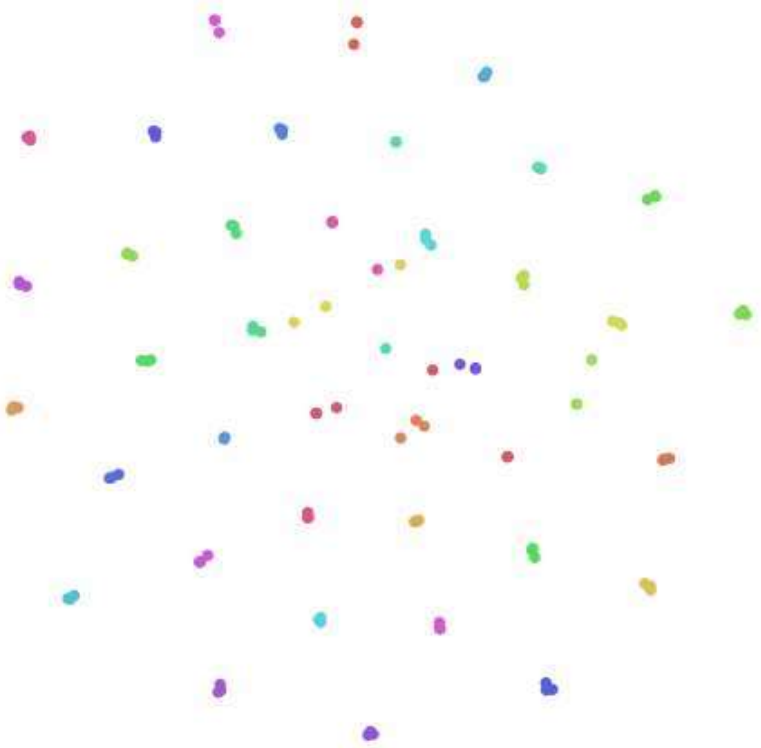
Assim como ocorreu com a base *Olivetti*, o modelo que fez uso do *Constellation Loss* proporcionou melhores resultados com ambas as arquiteturas ResNet152 V2 e Inception V3, conforme visto na Tabela 10, sendo o único diferencial o valor de K.

Alcançando 98,7% de média de AUC-ROC, o modelo obteve um valor próximo ao obtido por Nam et al. (2018), onde o mesmo alcançou o valor da métrica de AUC-ROC de 98,8% no conjunto de dados LFW. Ao adicionar o classificador NN ao modelo, a acurácia obtida foi de 95,21%. Entretanto, uma comparação direta com a Tabela 3 se torna difícil, pois a quantidade de dados para cada trabalho é diferente devido às diferentes

Tabela 9 – Valor médio da acurácia no Conjunto de Dados *Olivetti Faces*

Função de Perda	**ResNet152 V2**	**Inception V3**
Triplet	99%	98,86%
Multiclass N-Pair	97,75%	96,86%
Constellation $K = 2$	98,17%	98,17%
Constellation $K = 3$	98,87%	98,66%
Constellation $K = 4$	**99,17%**	**99,17%**
Constellation $K = 5$	98,33%	98,66%
Constellation $K = 6$	98,33%	98,17%
Constellation $K = 7$	98,87%	98,33%

Figura 17 – Visualização 2D do t-SNE dos Vetores de *Embeddings* dos Dados de Teste do Olivetti Faces. Cada cor representa uma das classes do conjunto de dados



Fonte: O autor

metodologias para treinamento com a base. Outro ponto que dificulta a comparação, é o fato de que os trabalhos da Tabela 3 aplicam a etapa de detecção das faces, devido a falta de padronização nas imagens da base. A acurácia alcançada pelas outras configurações pode ser vista na Tabela 11, podendo ser visto que a melhor acurácia foi obtida para cada rede com o *Constellation Loss*, para os valores de K: 4 e 6.

Tabela 10 – Valor de AUC-ROC médio no Conjunto de Dados LFW

Função de Perda	**ResNet152 V2**	**Inception V3**
Triplet	0,985	0,970
Multiclass N-Pair	0,970	0,955
Constellation $K = 2$	0,974	0,958
Constellation $K = 3$	0,981	0,967
Constellation $K = 4$	0,986	0,958
Constellation $K = 5$	0,983	0,962
Constellation $K = 6$	**0,987**	**0,971**
Constellation $K = 7$	0,982	0,964

A base LFW tem como característica ser uma base desbalanceada, existindo classes com 2 exemplos e classes com mais de 15 exemplos. Além deste fato, a base possui um alto número de classes, dificultando a visualização do resultado pela técnica t-SNE, como pode ser visto na Figura 18. Pode ser notado que muitas classes estão bem distanciadas entre si, mas diferentemente da base *Olivetti*, a técnica t-SNE não forneceu uma visualização ideal da separabilidade expressa nos valores de AUC-ROC.

5.6 Análises Gerais

A primeira análise é feita em relação às diferentes arquiteturas utilizadas como *backbone* juntamente às diferentes funções de perda: quais alcançaram melhor desempenho na extração de características, ou seja, maior separação interclasse e distribuição intraclasse mais compacta.

Como pode ser observado pelas tabelas de valores médio de AUC-ROC, em ambas as bases de dados, a arquitetura ResNet152 V2 obteve melhores resultados do que a Inception V3 em todas as funções de perda. Este é um resultado notável, visto que até onde se tem conhecimento na literatura, o único trabalho que faz uso do *Constellation Loss* na área de visão computacional propõe o Inception V3 como arquitetura de *backbone*.

Outro ponto a ser notado é a afirmação de que o valor de K não segue a regra de quanto maior, melhor o resultado. Este fato pode ser confirmado principalmente pelos resultados da base *Olivetti*, onde $K = 4$ obteve o modelo com melhor desempenho. Um possível questionamento a ser feito é o valor de K estar associado diretamente ao número de classes presentes na base, visto que para a base LFW, o valor ótimo de K foi maior que o valor obtido para a base *Olivetti*. Tendo em vista que o valor de K define quantas classes serão afastadas entre si a cada passo de treinamento, esta afirmação pode ser verdadeira, entretanto, o presente trabalho treinou o modelo em apenas duas diferentes bases de dados, tornando-se inviável validar a afirmação.

Tabela 11 – Valor médio da acurácia no Conjunto de Dados LFW

Função de Perda	**ResNet152 V2**	**Inception V3**
Triplet	94,9%	92%
Multiclass N-Pair	91,77%	89,13%
Constellation $K = 2$	93,21%	90,3%
Constellation $K = 3$	94,33%	91,9%
Constellation $K = 4$	**95,21%**	90,5%
Constellation $K = 5$	94,33%	91,6%
Constellation $K = 6$	**95,21%**	**92,45%**
Constellation $K = 7$	93,77%	91,77%

Figura 18 – Visualização 2D do t-SNE dos Vetores de *Embeddings* dos Dados de Teste do LFW. Cada cor representa uma das classes do conjunto de dados

Fonte: O autor

A segunda análise fica em torno das diferentes funções de perda. Como esperado de uma função de perda definida como estado da arte, o *Triplet Loss* obteve um desempenho significativo, conseguindo ficar a frente do *Multiclass-N-Pair Loss* e do *Constellation Loss* para alguns valores de K. Porém, o destaque fica para o *Constellation Loss* que em ambas as bases de dados, superou as demais funções de perda, mesmo que por uma pequena diferença. Levando-se em consideração que até onde é conhecido não existe trabalho de reconhecimento facial que faça uso do *Constellation Loss*, os resultados alcançados são relevantes. Apesar disto, é importante perceber que a diferença de AUC-ROC entre os métodos é pequena, e talvez o verdadeiro potencial do *Constellation Loss* frente a outros métodos seria melhor percebido em bases de dados mais desafiadoras.

A terceira análise é realizada em torno da separabilidade interclasse e intraclasse, proporcionada pelas imagens da técnica t-SNE.

O modelo com arquitetura ResNet152 V2 junto à função de perda *Constellation Loss* resultou em um extrator de características com desempenho elevado na base de dados

Olivetti. A Figura 17 comprova esta afirmação, podendo ser facilmente visualizada a boa separação interclasse das amostras e as regiões densas da intraclasse. Entretanto, não foi possível tirar conclusões em torno da Figura 18, pois o banco LFW possui um grande número de classes, tornando difícil a visualização em um mapa 2D. Um mapa 3D poderia ser mais informativo, porém ainda sim inconclusivo, pois o espaço destinado a figura poderia ainda assim não ser o suficiente para mostrar a real separabilidade das classes.

A quarta e última análise é feita sobre o esforço computacional alcançado através da metodologia utilizada no presente trabalho para criação dos *batches* de treinamento. Como mencionado no início deste tópico, as funções de perda *Multiclass-N-Pair Loss* e *Constellation Loss* não puderam ser comparadas. O fato do computador de teste não ser capaz de executar o treinamento utilizando a metodologia tradicional utilizando a GPU para todas as funções de perda, prova que o método de criação dos *batches* reduz o custo computacional de forma significativa. Outra confirmação está nas Tabelas 5 e 6, onde o tempo gasto no treinamento do modelo com a metodologia proposta por Medela e Picón (2019) foi menor para as duas bases de dados usadas para avaliação.

6 Conclusão e Trabalhos Futuros

6.1 Conclusão

O presente trabalho teve como objetivo propor um modelo de extração de características para sistemas de reconhecimento facial, de modo a aglomerar espacialmente amostras de uma mesma pessoa e aumentar o distanciamento de aglomerações de amostras de outras pessoas, sendo tal representação denominada como *embedding*. Para isso, inicialmente foram analisados os problemas envolvendo essa tarefa, e as abordagens atuais consideradas estado da arte.

Primeiramente foi limitado o escopo de redes profundas utilizadas, e ficou decidido que o foco seria em redes convolucionais, CNN, atuando como o *backbone* do modelo, sendo ajustado via uma função de perda de natureza contrastiva.

Foi proposta a utilização de duas arquiteturas conhecidas pelos seus desempenhos no desafio ImageNet, a ResNet152 V2 e a Inception V3. Para função de perda, foi proposta o uso da metodologia *Constellation Loss*, a qual busca superar os desafios encontrados por outras técnicas de estado da arte em funções de perda contrastivas. Visto que não foram encontrados trabalhos para reconhecimento facial que faça uso da *Constellation Loss*, o modelo foi testado com *Triplet Loss* e *Multiclass-N-Pair Loss* para via de comparação e confirmação da eficácia do método.

Visando reduzir o custo computacional existente no treinamento de modelos RNS, foi desenvolvida uma metodologia específica, que, ao invés de construir uma estrutura RNS, a qual replica uma CNN para cada entrada considerada, formou-se um único *batch* de treinamento respeitando a organização das entradas esperadas pela função de perda empregada. Este procedimento eliminou a necessidade de replicar a rede para cada imagem de entrada, como é feita nas RNS, e garantiu uma maior facilidade na aplicação de funções de perda que levam em consideração 3 ou mais imagens de entrada, como as funções testadas neste trabalho.

Foram selecionadas duas bases de dados para a realização dos testes: *Olivetti Faces* e LFW. O primeiro devido a seu balanceamento, onde cada classe possui 10 imagens no mesmo ambiente controlado, enquanto que o segundo pelo seu grande uso na literatura e apresentar dificuldade mais elevada que a outra base. Para ambos os conjuntos foi aplicada a normalização dos dados e o redimensionamento para $224 \times 224 \times 3$.

A proposta alcançou elevados resultados ao aplicar a função de perda *Constellation Loss*, juntamente com a CNN *ResNet152V2* como *backbone*. O método proposto alcançou valores significativos de AUC-ROC e classes de *embeddings* bem separadas, principalmente

para a base *Olivetti Faces*. No conjunto de dados *Olivetti Faces*, o modelo atingiu $99,9\%$ de valor médio de AUC-ROC, enquanto nos dados do conjunto LFW, obteve $98,7\%$. Com base nos resultados encontrados, o modelo proposto provou sua eficácia, indicando a viabilidade de aplicação do mesmo em sistemas de reconhecimento facial.

Outra conclusão que pôde ser obtida através dos resultados é a redução do custo computacional da metodologia de construção dos *batches*. Com as especificações da máquina utilizada para execução dos testes, o treinamento convencional de um modelo RNS não conseguiu ser executado para o *Constellation Loss* e o *Multiclass-N-Pair Loss*. Unindo este fato à informação obtida pelos testes de que o tempo gasto no treinamento do modelo ao utilizar o *Triplet Loss* foi menor ao utilizar a metodologia de *batches*, conclui-se que o custo computacional foi reduzido e o objetivo alcançado.

6.2 Trabalhos Futuros

A principal possibilidade de um trabalho futuro fica em torno da metodologia *Constellation Loss*. O trabalho recente de Ghojogh et al. (2020) uniu os conceitos das funções de perda *Contrastive Loss* e *Triplet Loss* com o Discriminante de Fisher, alcançando resultados significativos na separação das amostras de diferentes classes. O interesse inicial do autor do presente trabalho era a aplicação do conceito de forma a ser desenvolvido o *Fisher Discriminant Constellation Loss*. Entretanto, para o desenvolvimento da função, seria necessário mudar toda a abordagem, utilizando apenas o *Tensorflow* para a construção das redes e para a execução do treinamento. Além do tempo que esta mudança demandaria, precisaria ser verificada a possibilidade de utilização do método de criação dos *batches* de treinamento aplicado no presente trabalho, visto que a API *Keras* torna possível a utilização do método de forma intuitiva.

Outro ponto seria o fato de que, como foi dito anteriormente, um modelo de reconhecimento facial conta com etapas obrigatórias: a detecção da face, o pré-tratamento, a extração de características e a classificação. O presente trabalho teve como foco a etapa de extração de características, porém, seria interessante o desenvolvimento de um modelo completo que possa fazer uso do modelo apresentado neste trabalho junto à um detector de faces, aplicando pré-tratamento nas faces detectadas.

Algumas possibilidades ficam em torno de explorar a metodologia em outras bases de dados. Todas as configurações utilizadas para teste no presente trabalho obtiveram uma boa resposta no conjunto *Olivetti*, levantando o questionamento se a utilização em um banco balanceado, porém mais desafiador, não seria uma abordagem que proporcionasse mais conclusões em seus resultados. Uma possível proposta seria a utilização de faces *fakes* para construção de uma base de dados desafiadora, tornando possível testar a robustez do modelo com as mais diferentes variações de faces. Quanto à base *LFW*, mesmo sendo

amplamente abordada na literatura, possui particularidades para execução do treino em cada trabalho da literatura que foi utilizado, isto devido ao alto desbalanceamento, o que torna difícil a comparação dos resultados encontrados na base com outros divulgados. Uma validação *Cross-Dataset* em um futuro trabalho forneceria uma melhor confirmação do poder de generalização do modelo, comprovando que o re-treinamento não se faz necessário.

Uma técnica que tem ganhado respaldo na literatura é o aumento de dados. Para o presente trabalho, a técnica foi testada apenas para o *Olivetti Faces*, visto que o trabalho utilizado para comparação fazia uso da mesma. No teste aplicado, a técnica não proporcionou nenhuma melhoria na acurácia alcançada, porém, em bases como o LFW, uma possibilidade seria aplicar o aumento de dados nas classes com menor quantidade de exemplos, de modo a superar seu conhecido problema de desbalanceamento, podendo alcançar melhores resultados.

Referências

ALBIOL, A. et al. Face recognition using hog–ebgm. Pattern Recognition Letters, v. 29, p. 1537–1543, 07 2008. Citado na página 21.

ASMAG. Facing challenges in face recognition: one-to-one vs. one-to-many. 2019. <https://www.asmag.com/showpost/21158.aspx>. Acessado em 05/02/2022. Citado 2 vezes nas páginas 17 e 18.

BAKHSHI, Y.; KAUR, S.; VERMA, P. An improvement in face recognition for invariant faces. International Journal of Current Engineering and Technology, v. 6, p. 423–426, 03 2016. Citado na página 20.

BANDALA, A. et al. Simultaneous face detection and recognition using viola-jones algorithm and artificial neural networks for identity verification. In: . [S.l.: s.n.], 2013. Citado 2 vezes nas páginas 21 e 34.

BHAT, V. S.; PUJARI, J. D. Human face recognition using combined approaches pca and ica. Digital Image Processing, v. 7, p. 301–307, 2015. Citado na página 20.

BROMLEY, J. et al. Signature verification using a siamese time delay neural network. In: . [S.l.: s.n.], 1993. v. 7, p. 737–744. Citado 3 vezes nas páginas 12, 22 e 26.

CHENG, G. et al. When deep learning meets metric learning: Remote sensing image scene classification via learning discriminative cnns. IEEE Transactions on Geoscience and Remote Sensing, PP, p. 1–11, 01 2018. Citado na página 22.

CHOPRA, S.; HADSELL, R.; LECUN, Y. Learning a similarity metric discriminatively, with application to face verification. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). [S.l.: s.n.], 2005. v. 1, p. 539–546 vol. 1. Citado 2 vezes nas páginas 13 e 22.

COTTRELL, G.; FLEMING, M. Face recognition using unsupervised feature extraction. Proc. Int'l Neural Network Conf., v. 3, p. 322–325, 01 1990. Citado na página 12.

DA'SAN, M.; ALQUDAH, A.; DEBEIR, O. Face detection using viola and jones method and neural networks. In: . [S.l.: s.n.], 2015. p. 40–43. Citado na página 21.

DESHPANDE, N. T.; RAVISHANKAR, D. S. Face detection and recognition using viola-jones algorithm and fusion of lda and ann. In: . [S.l.: s.n.], 2016. Citado na página 21.

DHIFLI, W.; DIALLO, A. Face recognition in the wild. Procedia Computer Science, v. 96, p. 1571–1580, 12 2016. Citado na página 33.

FACEBOOK. An Update On Our Use of Face Recognition. 2021. <https://about.fb.com/news/2021/11/update-on-use-of-face-recognition/>. Acessado em 04/02/2022. Citado na página 17.

FARFADE, S.; SABERIAN, M.; LI, L.-J. Multi-view face detection using deep convolutional neural networks. 02 2015. Citado na página 21.

GHOJOGH, B. et al. Fisher discriminant triplet and contrastive losses for training siamese networks. CoRR, abs/2004.04674, 2020. Disponível em: <https://arxiv.org/abs/2004.04674>. Citado na página 51.

GOLDSTEIN, A.; HARMON, L.; LESK, A. Identification of human faces. Proceedings of the IEEE, v. 59, p. 748 – 760, 06 1971. Citado na página 12.

GROTHER, P.; NGAN, M. Face recognition vendor test (frvt). In: . [S.l.: s.n.], 2014. Citado na página 16.

HAZIM, N. Face recognition using pca-bpnn with dct implemented on face94 and grimace databases. International Journal of Computer Applications, v. 142, p. 8–13, 05 2016. Citado na página 20.

HE, K. et al. Deep residual learning for image recognition. In: . [S.l.: s.n.], 2016. p. 770–778. Citado na página 21.

HE, K. et al. Identity mappings in deep residual networks. CoRR, abs/1603.05027, 2016. Disponível em: <http://arxiv.org/abs/1603.05027>. Citado na página 35.

HU, J.; SHEN, L.; SUN, G. Squeeze and excitation networks. In: . [S.l.: s.n.], 2018. p. 7132–7141. Citado na página 21.

HUANG, G. et al. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. Tech. rep., 10 2008. Citado na página 34.

HUBEL, D. H.; WIESEL, T. N. Receptive fields of single neurons in the cat's striate cortex. Journal of Physiology, v. 148, p. 574–591, 1959. Citado na página 24.

JAFRI, R.; ARABNIA, H. R. A survey of face recognition techniques. J. Inf. Process. Syst., v. 5, p. 41–68, 2009. Citado na página 19.

KADAM, K. D. Face recognition using principal component analysis with dct. In: . [S.l.: s.n.], 2014. Citado na página 20.

KATHER, J. et al. Multi-class texture analysis in colorectal cancer histology. Scientific Reports, v. 6, p. 27988, 06 2016. Citado na página 22.

KOCH, G.; ZEMEL, R.; SALAKHUTDINOV, R. Siamese neural networks for one-shot image recognition. In: . [S.l.: s.n.], 2015. Citado na página 22.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. Imagenet classification with deep convolutional neural networks. Neural Information Processing Systems, v. 25, 01 2012. Citado na página 21.

KSHIRSAGAR, V.; BAVISKAR, M.; GAIKWAD, M. Face recognition using eigenfaces. In: 2011 3rd International Conference on Computer Research and Development. [S.l.: s.n.], 2011. v. 2, p. 302–306. Citado na página 20.

LASTPASS. O que é Reconhecimento Facial? 2018. <https://blog.idwall.co/o-que-e-reconhecimento-facial/>. Acessado em 10/02/2022. Citado na página 12.

LAWRENCE, S. et al. Face recognition: a convolutional neural-network approach. IEEE Transactions on Neural Networks, v. 8, n. 1, p. 98–113, 1997. Citado na página 20.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. Nature, v. 521, p. 436–44, 05 2015. Citado na página 21.

LI, H. et al. A convolutional neural network cascade for face detection. In: . [S.l.: s.n.], 2015. p. 5325–5334. Citado na página 21.

LIU, W. et al. Sphereface: Deep hypersphere embedding for face recognition. In: . [S.l.: s.n.], 2017. Citado na página 22.

LU, P.; SONG, B.; XU, L. Human face recognition based on convolutional neural network and augmented dataset. Systems Science & Control Engineering, Taylor & Francis, v. 9, n. sup2, p. 29–37, 2021. Disponível em: <https://doi.org/10.1080/21642583.2020.1836526>. Citado na página 44.

LUO, P.; WANG, X.; TANG, X. Hierarchical face parsing via deep learning. In: . [S.l.: s.n.], 2012. p. 2480–2487. ISBN 978-1-4673-1226-4. Citado na página 21.

MAATEN, L. van der; HINTON, G. Viualizing data using t-sne. Journal of Machine Learning Research, v. 9, p. 2579–2605, 11 2008. Citado na página 40.

MEDELA, A.; PICÓN, A. Constellation loss: Improving the efficiency of deep metric learning loss functions for optimal embedding. CoRR, abs/1905.10675, 2019. Disponível em: <http://arxiv.org/abs/1905.10675>. Citado 9 vezes nas páginas 13, 14, 15, 22, 30, 34, 35, 37 e 49.

MEDELA, A. et al. Few shot learning in histopathological images:reducing the need of labeled data on biological datasets. In: . [S.l.: s.n.], 2019. p. 1860–1864. Citado na página 22.

MEI, W.; DENG, W. Deep face recognition: A survey. Neurocomputing, v. 429, 04 2018. Citado 3 vezes nas páginas 18, 22 e 23.

MOVELLAN, J.; SEJNOWSKI, T. Face recognition by independent component analysis. IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council, v. 13, p. 1450–64, 02 2002. Citado na página 20.

NAM, G. et al. Psi-cnn: A pyramid-based scale-invariant cnn architecture for face recognition robust to various image resolutions. Applied Sciences, v. 8, p. 1561, 09 2018. Citado na página 45.

NVIDIA. DeepVision - Face Recognition. 2020. <https://catalog.ngc.nvidia.com/orgs/partners/teams/deepvision/containers/vf.face.recognition>. Acessado em 05/02/2022. Citado na página 17.

PWC. Pesquisa Global sobre Fraudes e Crimes Econômicos 2020. 2020. <https://www.pwc.com.br/pt/estudos/servicos/consultoria-negocios/2020/pesquisa-global-sobre-fraudes-e-crimes-economicos-2020.html>. Acessado em 07/02/2022. Citado na página 12.

RAZZAQ, A.; GHAZALI, R.; ABBADI, N. E. Face recognition – extensive survey and recommendations. In: . [S.l.: s.n.], 2021. p. 1–10. Citado na página 20.

REYNOLDS, A. H. Convolutional Neural Networks (CNNs). 2019. <https://anhreynolds.com/blogs/cnn.html>. Acessado em 04/02/2022. Citado na página 25.

RUSSAKOVSKY, O. et al. Imagenet large scale visual recognition challenge. International Journal of Computer Vision, v. 115, 09 2014. Citado na página 21.

SALAMA, D.-D. et al. A deep facial recognition system using computational intelligent algorithms. PLoS ONE, v. 15, p. e0242269, 12 2020. Citado na página 16.

SANTOS, E. R. dos; CIARELLI, P. M.; SAMATELO, J. L. A. Constellation loss em modelos de reconhecimento facial. In: XV Simpósio Brasileiro de Automação Inteligente. [S.l.: s.n.], 2021. p. 1679–1686. ISSN 2175-8905. Citado 7 vezes nas páginas 28, 29, 30, 31, 35, 37 e 38.

SCHROFF, F.; KALENICHENKO, D.; PHILBIN, J. Facenet: A unified embedding for face recognition and clustering. Proc. CVPR, 03 2015. Citado 6 vezes nas páginas 13, 22, 27, 28, 34 e 35.

SHARMA, N.; DUBEY, S. Face recognition analysis using pca, ica and neural network. In: . [S.l.: s.n.], 2014. Citado na página 20.

SIMONYAN, K.; ZISSERMAN, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014. Cite arxiv:1409.1556. Disponível em: <http://arxiv.org/abs/1409.1556>. Citado na página 21.

SOHN, K. Improved deep metric learning with multi-class n-pair loss objective. In: LEE, D. et al. (Ed.). Advances in Neural Information Processing Systems. Curran Associates, Inc., 2016. v. 29. Disponível em: <https://proceedings.neurips.cc/paper/2016/file/6b180037abbebea991d8b1232f8a8ca9-Paper.pdf>. Citado 3 vezes nas páginas 13, 22 e 29.

STEEN, D. Understanding the ROC Curve and AUC. 2020. <https://towardsdatascience.com/understanding-the-roc-curve-and-auc-dd4f9a192ecb>. Acessado em 04/02/2022. Citado na página 41.

SUN, Y.; WANG, X.; TANG, X. Deep learning face representation from predicting 10,000 classes. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, p. 1891–1898, 09 2014. Citado na página 21.

SZEGEDY, C. et al. Going deeper with convolutions. CoRR, abs/1409.4842, 2014. Disponível em: <http://arxiv.org/abs/1409.4842>. Citado na página 13.

SZEGEDY, C. et al. Going deeper with convolutions. In: . [S.l.: s.n.], 2015. p. 1–9. Citado na página 21.

SZEGEDY, C. et al. Rethinking the Inception Architecture for Computer Vision. 2015. Cite arxiv:1512.00567. Disponível em: <http://arxiv.org/abs/1512.00567>. Citado 2 vezes nas páginas 22 e 35.

TAIGMAN, Y. et al. Deepface: Closing the gap to human-level performance in face verification. In: . [S.l.: s.n.], 2014. Citado na página 21.

TURK, M.; PENTLAND, A. Eigenfaces for recognition. Journal of cognitive neuroscience, v. 3, p. 71–86, 01 1991. Citado na página 12.

TURK, M.; PENTLAND, A. Face recognition using eigenfaces. In: . [S.l.: s.n.], 1991. p. 586 – 591. ISBN 0-8186-2148-6. Citado na página 12.

VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001. [S.l.: s.n.], 2001. v. 1, p. I–I. Citado na página 21.

VIOLA, P.; JONES, M. Robust real-time face detection. International Journal of Computer Vision, v. 57, p. 137–154, 05 2004. Citado na página 21.

WEINBERGER, K.; BLITZER, J.; SAUL, L. Distance metric learning for large margin nearest neighbor classification. In: _____. [S.l.: s.n.], 2006. v. 10. Citado na página 22.

WHITE, D. et al. Error rates in users of automatic face recognition software. PLOS ONE, v. 10, p. e0139827, 10 2015. Citado na página 17.

XUE, Y.; WHITECROSS, K.; MIRZASOLEIMAN, B. Investigating why contrastive learning benefits robustness against label noise. CoRR, abs/2201.12498, 2022. Disponível em: <https://arxiv.org/abs/2201.12498>. Citado na página 13.

YAMASHITA, R. et al. Convolutional neural networks: an overview and application in radiology. Insights into Imaging, v. 9, 06 2018. Citado na página 25.

ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional networks. CoRR, abs/1311.2901, 2013. Disponível em: <http://arxiv.org/abs/1311.2901>. Citado na página 13.

ZHAO, W.-Y. et al. Face recognition: A literature survey. ACM Comput. Surv., v. 35, p. 399–458, 12 2003. Citado 2 vezes nas páginas 16 e 19.

A Trabalhos Publicados

Um artigo derivado deste trabalho foi publicado em eventos:

- SANTOS, E. R. dos; CIARELLI, P. M.; SAMATELO, J. L. A. Constellation loss em modelos de reconhecimento facial. In: XV Simpósio Brasileiro de Automação Inteligente. 2021. p. 1679–1686. ISSN 2175-8905.