TRABALHO DE GRADUAÇÃO

Re-Identificação de pessoas em diferentes domínios a partir de imagens de CCTV

Tiago de Carvalho Gallo Pereira

Brasília, Julho de 2019

ENGENHARIA
MECATRÔNICA
UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia
Curso de Graduação em Engenharia de Controle e Automação

TRABALHO DE GRADUAÇÃO

Re-Identificação de pessoas em diferentes domínios a partir de imagens de CCTV

Tiago de Carvalho Gallo Pereira

Relatário submetido como requisito parcial de obtenção de grau de Engenheiro de Controle e Automação

Banca Examinadora

Prof. Teófilo Emídio de Campos, CIC/UnB _____
Orientador

Prof. Alexandre Ricardo Soares Romariz, ENE/UnB _____

Prof. Flávio de Barros Vidal, CIC/UnB _____

Brasília, Julho de 2019

FICHA CATALOGRÁFICA

Tiago, de Carvalho Gallo Pereira
Re-Identificação de pessoas em diferentes domínios a partir de imagens de CCTV

[Distrito Federal] 2019.

viii, 64p., 297 mm (FT/UnB, Engenheiro, Controle e Automação, 2019). Trabalho de Graduação – Universidade de Brasília.Faculdade de Tecnologia.

1. Visão Computacional 2.Aprendizado de Máquina
3. Re-Identificação de pessoas

I. Mecatrônica/FT/UnB II. Controle e Automação

REFERÊNCIA BIBLIOGRÁFICA

Pereira, Tiago de Carvalho Gallo, (2019). Re-Identificação de pessoas em diferentes domínios a partir de imagens de CCTV. Trabalho de Graduação em Engenharia de Controle e Automação, Publicação FT.TG-$n°5$, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 64p.

CESSÃO DE DIREITOS

AUTOR: Tiago de Carvalho Gallo Pereira

TÍTULO DO TRABALHO DE GRADUAÇÃO: Re-Identificação de pessoas em diferentes domínios a partir de imagens de CCTV.

GRAU: Engenheiro ANO: 2019

Tiago de Carvalho Gallo Pereira

Campus Universitário Darcy Ribeiro, UnB

70910-900 Brasília – DF – Brasil.

Dedicatória

Dedico a esse trabalho a todas pessoas que sempre me apoiaram e acreditaram em mim.

Tiago de Carvalho Gallo Pereira

Agradecimentos

Agradeço, primeiramente, a Deus por ter me dado essa oportunidade. A minha família e minha namorada por toda a ajuda e o suporte. A todos os funcionários da CyberLabs que me apoiram na confecção deste trabalho, em especial ao Anthony Phillips que sempre me incentivou e trabalhou diretamente comigo na confecção da base de dados. Ao meu orientador, Teófilo Campos, por todos os direcionamentos e ensinamentos. Por fim, um grande agradecimento ao meu pai, Daniel Gallo, por ter me acompanhado de perto em todo o trabalho e me ajudado a melhorar a escrita do relatório para torna-lo legível.

Tiago de Carvalho Gallo Pereira

RESUMO

No universo da visão computacional, a re-identificação de pessoas em diferentes vistas é um desafio ainda não dominado. Cada vista pode apresentar diferentes características variáveis nas imagens das pessoas, seja angulação, iluminação, foco, distorções, roupas. Pode-se denominar esse conjunto de variáveis como um domínio. A re-identificação de pessoas em diferentes domínios é um problema de grande complexidade. Uma solução clássica para re-identificar pessoas em diferentes vistas seria utilizar biometria facial, no entanto há várias restrições para o funcionamento dessa solução. Por exemplo, a câmera precisa estar focada no rosto da pessoa (frontal e na altura do rosto) e há dependência da resolução no rosto da pessoa. O objetivo do presente trabalho é estudar uma solução para o desafio de re-identificação de pessoas em diferentes domínios, que utilize todo o escopo de dados disponível na imagem da pessoa. Para isso, analisou-se as técnicas existentes para re-identificação de pessoas, validarou-se essas técnicas em bases de dados públicas e foi proposta uma nova técnica.

Palavras Chave: Visão Computacional, Aprendizado de Máquina, Inteligência Artificial, Aprendizado Profundo, Re-Identificação, Adaptação de Domínio

ABSTRACT

In computer vision, person re-identification with different camera views still is an open challenge. Changes in camera locations cause severe changes in the appearance of people due to changes in camera angle, light, focus, distortions and people's clothes. We name this group of variables as a domain. The task of re-identifying people across these different domains is a formidable problem. A classical approach to re-identifying people in different domains would be by using face biometrics, however there the approach has some limitations. For instance, the camera needs to focus on the person face and it should preferrably be at the face's height and acquire a frontal view. A large enough resolution is also required in the region of the face. This work's objective is to study a solution for the problem of person re-identification across different domains by using the entire data available in the person's image. To accomplish it, we analyze the existing techniques for person re-identification, validate those techniques on public databases and propose a novel technique.

Keywords: Computer Vision, Machine Learning, Artificial Intelligence, Deep Learning, Re-identification, Domain Adaptation

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Capítulo 1

Introdução

A re-identificação de pessoas é um grande desafio no campo da visão computacional. Para entender melhor a complexidade e a necessidade de se resolver esse desafio, este capítulo apresenta uma motivação do tema na seção 1.1. O escopo do problema é analisado na seção 1.2 e os objetivos propostos são apresentados na seção 1.3. A seção 1.4 resume os resultados obtidos.

1.1 Motivação

Atualmente, as pessoas necessitam cada vez mais de se sentirem seguras e protegidas. Os *hadrwares*, como câmeras de segurança e gravadores de vídeos digital, se tornaram equipamentos com preços acessíveis ao público geral. Com a combinação desses dois fatores, houve um crescimento muito grande na quantidade de CFTV [1] (circuito fechado de televisão) no mundo. Portanto, há uma grande massa de dados de vídeos e imagens sendo geradas o tempo inteiro.

No entanto, o custo para se analisar todos esses dados é muito alto. Por exemplo, para um grande estabelecimento, como um *shopping*, seria necessária uma equipe de funcionários responsáveis por monitorar as imagens e extrair informações úteis, em tempo integral. A Figura 1.1 mostra um grande centro de monitoração e operação na tecnologia atual.

Portanto, há um problema de necessidades vs custo de operação para tornar viável a interpretação dos dados dos CFTV. Atrelado a esse problema, há um custo de oportunidade entre a quantidade de câmeras instaladas e o custo das pessoas para analisá-las. No entanto, os estudos no campo da visão computacional tem sido desenvolvidos com o intuito de facilitar e tornar menos entediante esse trabalho.

Para cada conjunto de informações que pode-se obter com o monitoramento de câmeras de um CFTV, existem técnicas de visão computacional sendo desenvolvidas para resolver esses problemas, com redução na necessidade de intervenção humana. Por exemplo, técnicas de reconhecimento de ações estão sendo desenvolvidas para detectar atividades suspeitas [23], técnicas de reconhecimento

[1] Os circuitos fechados de televisão são sistemas internos de monitoramento e vigilância, esses sistemas são compostos por câmeras de segurança e gravadores de vídeos digital, que são responsáveis por apresentar as imagens das câmeras de segurança em monitores e, se necessário, gravar as imagens dessas câmeras.

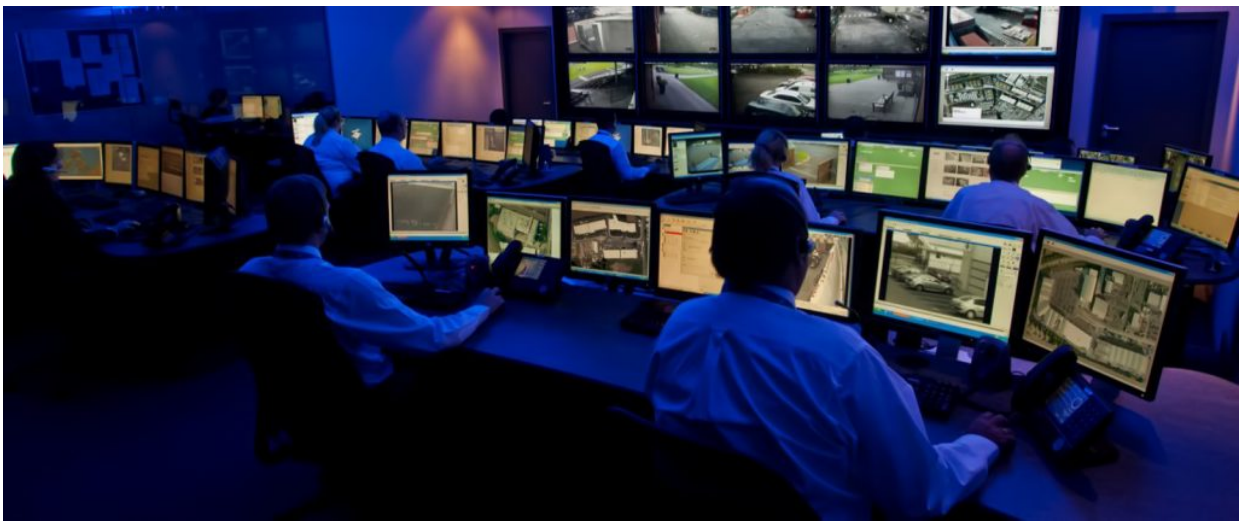

Figura 1.1: Ilustração típica de equipes de monitoramento responsável por CFTV de grandes estabelecimentos. Reproduzida de [1].

facial são desenvolvidas para autenticar pessoas [24], técnicas de re-identificação de pessoas estão sendo desenvolvidas para acompanhar a movimentação de uma pessoa, dentro de um ambiente onde ela é vista por várias câmeras diferentes.

1.2 Definição do problema

A re-identificação é o processo de identificar uma pessoa a partir de várias imagens ou vídeos capturados de diferentes câmeras, quando não há sobreposição entre as vistas de cada câmera, ou de uma mesma câmera em momentos distintos. Ou seja, dado um conjunto de dados (imagens ou vídeos), o problema consiste em associar esses dados a uma pessoa e conseguir identificar essa pessoa em diferentes câmeras e diferentes momentos no tempo.

Explicando o problema de um ponto de vista prático, pode-se pensar no desafio de acompanhar uma pessoa que está passeando por um *shopping center*, sem precisar que alguém fique olhando para as câmeras e procurando essa pessoa. Portanto, o algoritmo utiliza das imagens das câmeras de segurança do *shopping center* e aprende a aparência da pessoa (pode-se dizer que o algoritmo aprende uma assinatura da pessoa). Logo, o desafio é criar esse algoritmo que saiba que a pessoa que entrou pela garagem é a mesma que passou em frente ao cinema e também é a mesma que está sentada na praça de alimentação.

Cada câmera tem várias caraterísticas diferentes, como angulação, iluminação, distorções. Além disso, as pessoas podem apresentar variações quando vistas em câmeras diferentes, como roupas (a pessoa pode colocar/retirar um casaco ou chapéu) ou posição do corpo (a pessoa pode ser vista de costas em uma câmera e de frente ou de lado na outra). O conjunto de variáveis são denominados um domínio, ou seja pode-se tratar cada câmera como um domínio distinto. O desafio passa a ser identificar as pessoas em diferentes domínios, esse fato acrescenta uma grande complexidade ao problema. A Figura 1.2 mostra o quanto a aparência de uma pessoa pode mudar de uma câmera

para outra, dificultando o processo de re-identificação.

Figura 1.2: Exemplo de imagens de uma mesma pessoa em diferentes câmeras. Fonte: Base de dados CUHK03 [2]. ©2014 IEEE.

1.3 Objetivos do projeto

O objetivo do presente trabalho é estudar uma solução para o desafio de re-identificação de pessoas em diferentes domínios, utilizando todo o escopo de dados disponível na imagem da pessoa. Inicialmente, são analisadas algumas técnicas existentes para re-identificação de pessoas. Validadas essas técnicas em bases de dados públicas, e proposta uma nova técnica, ou combinação de técnicas existentes.

Outro objetivo desse trabalho foi o de resolver, para a *CyberLabs* (*startup* brasileira), o problema de cronometrar o tempo de pessoas em uma fila, de forma automatizada. Para isso, é utilizado o algoritmo desenvolvido de re-identificação de pessoas. As imagens da mesma pessoa são agrupadas na vista da entrada e na vista da saída da fila. O tempo entre esses dois grupos de imagens é cronometrado para estimar o tempo que essa pessoa passou na fila.

Objetiva-se, também, neste trabalho, estudar o potencial de alguns métodos de adaptação de domínio. A ideia é possibilitar a utilização, em um domínio alvo, de redes neurais treinadas em um domínio fonte e obter resultados melhores que o aleatório, e melhores que redes neurais que não usaram os métodos de adaptação de domínio.

1.4 Resultados obtidos

Obteve-se um bom conhecimento sobre as técnicas utilizadas e os atuais resultados obtidos pelo estado da arte, no desafio de re-identificação de pessoas, a partir do estudo da literatura. Além do

mais, foram treinadas redes neurais que obtiveram resultados razoáveis em bases de dados públicas que são utilizadas para *benchmark*[2]. Os resultados obtidos ainda estão distantes do estado da arte, no entanto já se demonstram muito melhores do que um palpite aleatório e já tem uso prático.

Criou-se também uma base de dados proprietária (*CyberQueue*) com o objetivo de resolver um problema para a *CyberLabs*. Essa base de dados não pode ser publicada por questões de direitos autorais de imagem, mas foi utilizada para testes de adaptação de domínio e para transferência de conhecimento durante o treinamento das redes neurais. Ademais, treinou-se uma rede especialista para a base de dados *CyberQueue*, essa rede alcançou resultados de treino satisfatórios e aplicabilidade prática com uma boa performance.

Os testes com métodos de adaptação de domínio mostraram todo o potencial desse tipo de técnica. Mesmo não alcançando resultados tão bons como os treinamentos em domínio específico, os métodos de adaptação de domínio mostraram-se melhores que o aleatório e foi possível ver um incremento no resultado a cada novo método implementado. Portanto, acredita-se que esse tipo de técnica está em uma fase inicial de estudos e ainda tem capacidade para evoluir bastante, resolvendo a dificuldade de trabalhar com múltiplos domínios.

[2]O processo de *benchmark* consiste em utilizar sempre as mesmas condições de testes para avaliar algum algoritmo, os resultados são anotados no final do processo com o objetivo de serem utilizados como referência para futuros testes que desejam melhorar os resultados obtidos a priori.

Capítulo 2

Fundamentos

Para abordar os fundamentos mais importantes relacionados ao desafio de re-identificação de pessoas, esse capítulo foi dividido em 8 seções. A seção 2.1 conta o histórico de métodos propostos para resolver o desafio em questão e motiva o uso de redes neurais. A seção 2.2 explica em maiores detalhes como extrair o máximo das redes neurais para o desafio em questão. A seção 2.3 explica como é feita a divisão dos dados para o treinamento de uma rede neural. Na seção 2.4, discorre-se os fundamentos por detrás das técnicas de aumento de dados para extrair o máximo de informações da base de dados utilizada. A seção 2.5 demonstra a importância da escolha da função de custo ao treinar uma rede neural. As seções 2.6 e 2.7 apresentam parâmetros e técnicas importantes serem levadas em consideração ao se treinar uma rede neural. A seção 2.8 apresenta um outro tipo de rede neural que é muito utilizada para auxiliar no treinamento das redes neurais de re-identificação de pessoas.

2.1 Contextualização

Na última década, com o aumento dos CFTVs e de câmeras de segurança, cresceu a necessidade de criar algoritmos capazes de analisar esses dados. Logo, o desafio em questão tem sido cada vez mais estudado, como se pode ver na Figura 2.1 o crescimento do número de publicações desse tema nos últimos anos. Analisando a Figura 2.1, tem-se duas subidas marcantes no número de publicações. A primeira nos meados de 2010 que, provavelmente, está ligada com o aumento dos CFTVs e a segunda nos meados de 2014 que está relacionado com a explosão das redes neurais e os resultados que essas obtiveram.

2.1.1 Histórico de métodos utilizados para o desafio de re-identificação de pessoas

O desafio de re-identificação de pessoas não é recente e muitas estratégias diferentes foram testadas ao longo do tempo. As primeiras estratégias usavam técnicas de rastreamento de objetos (pessoas) em múltiplas câmeras. Depois, foram criadas técnicas que extraiam um vetor de atributos da imagem das pessoas para auxiliar no rastreamento em múltiplas câmeras. Com as melhorias

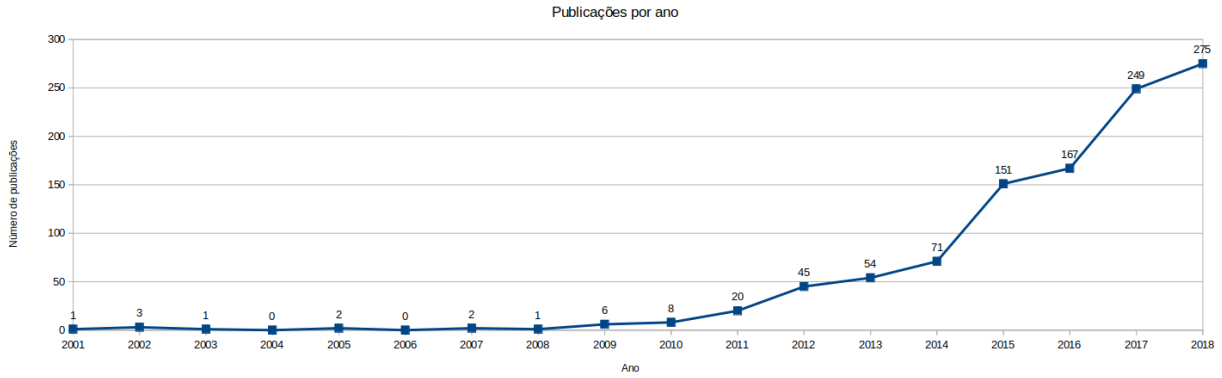

Figura 2.1: Número de publicações por ano com o tema re-identificação de pessoas. Fonte: *Web Of Science*.

nas técnicas de extração de um vetor de atributos da imagem das pessoas, deixou-se de usar o rastreamento entre câmeras e começou-se a usar apenas o vetor de atributos para re-identificar as pessoas. O vetor de atributos obtido por apenas uma imagem não era muito robusto à variações de domínios, tais como iluminação, posição, distância das pessoas, portanto foram criadas técnicas que utilizavam vídeos para obter vetores defasados no tempo. Com vários vetores distintos, a descrição da pessoa ficou mais robusta. Por volta 2014, as redes neurais profundas (*deep neural networks*) começaram a ser utilizadas para resolver problemas de visão computacional e se tornaram a técnica utilizada pelo estado da arte nos dias de hoje. A Figura 2.2 mostra como foi a evolução dessas técnicas ao longo do tempo.

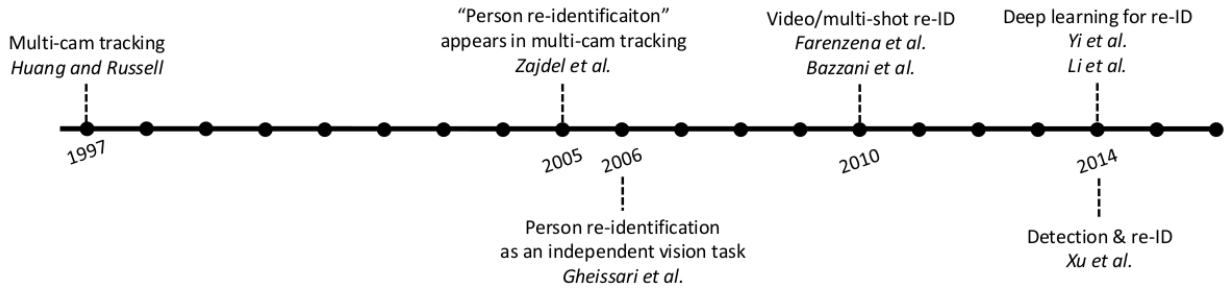

Figura 2.2: Linha do tempo das técnicas de re-identificação de pessoas. Reproduzida de [3].

Inicialmente, em 1997, Huang and Russell [25] propuseram uma técnica de rastreamento por múltiplas câmeras que não era específica para o desafio de re-identificação de pessoas. Essa técnica utiliza uma formulação Bayesiana para predizer a aparição de um objeto em uma câmera dado que ele já foi visto em outra câmera. O modelo inclui fatores como cor, comprimento, altura, largura, velocidade e momento de observação para cada objeto.

Em 2005, Zajdel, Zivkovic e Krose [26] utilizaram uma rede Bayesiana dinâmica especificamente para o problema de re-identificação de pessoas. Eles utilizaram as características como cor e movimentação para acompanhar as pessoas na câmera e perceber se uma pessoa que já havia sido identificada voltasse a aparecer na câmera.

Já em 2006, Gheissari [27] propôs um algoritmo de segmentação espaço-temporal para gerar assinaturas das pessoas em cada câmera. Esse algoritmo utiliza informações de cores e histogramas para gerar as assinaturas das pessoas, o algoritmo busca gerar assinaturas que são robustas à possíveis mudanças na vestimenta das pessoas.

A técnica utilizada por Farenzena e Bazzani [4] em 2010, consiste na extração de atributos da imagem da pessoa que modelem três aspectos complementares da imagem de uma pessoa, o conteúdo cromático geral, o arranjo espacial das cores no corpo da pessoa e a recorrência de padrões locais de alta entropia. Combinando essas métricas, a técnica promete ser robusta contra imagens de baixa resolução, oclusão e variações de pose e iluminação. A Figura 2.3 mostra as etapas de extração das métricas para essa técnica.

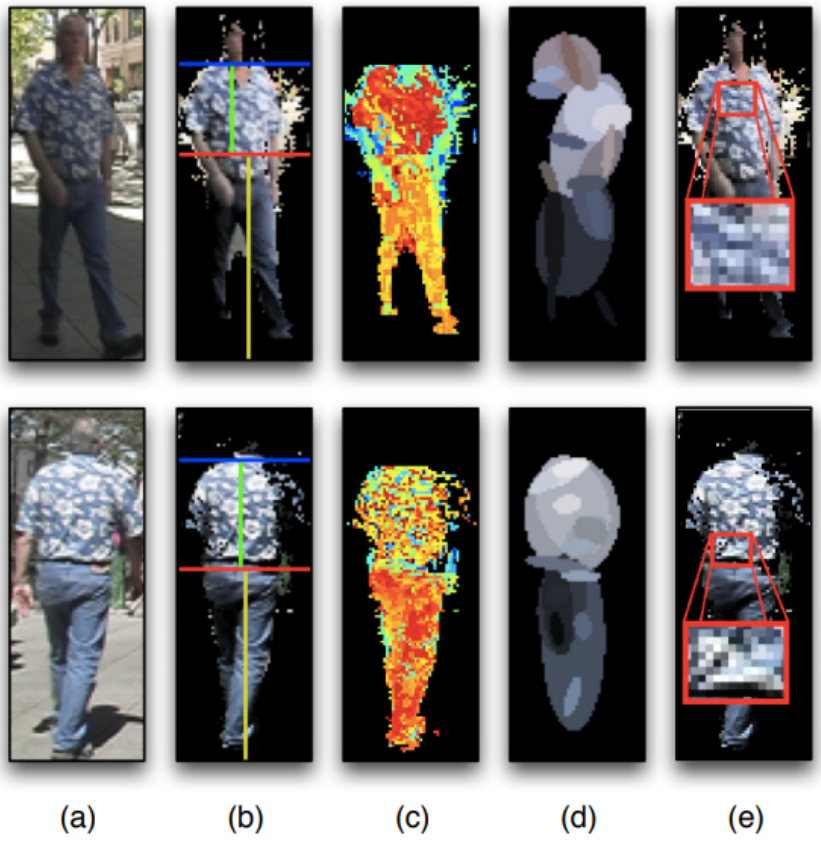

Figura 2.3: Etapas de extração das métricas utilizadas na técnica de Farenzena e Bazzani [4]. a) Mesma pessoa em duas vistas distintas; b) Eixos x e y de simetria e assimetria; c) Histograma ponderado, onde as cores mais quentes indicam regiões de maior interesse; d) Regiões de onde o arranjo espacial das cores é estável; e) Padrões locais recorrentes e altamente estruturados. Reproduzida de [4]. ©2010 IEEE.

A partir de 2014, com o sucesso no uso de redes neurais para resolver problemas de visão computacional, começaram a aparecer técnicas utilizando esse artifício para o problema de re-identificação de pessoas. Li e Zhao [2] propuseram uma rede neural de 6 camadas para tratar problemas de desalinhamento, transformações fotométricas e geométricas entre as vistas, oclusões e ruído de fundo. Eles também criaram uma nova base de dados pública para o treinamento de

redes neurais, chamada *CUHK03*, que ainda é usada na literatura nos dias de hoje. A Figura 2.4 mostra a arquitetura da rede criada por eles.

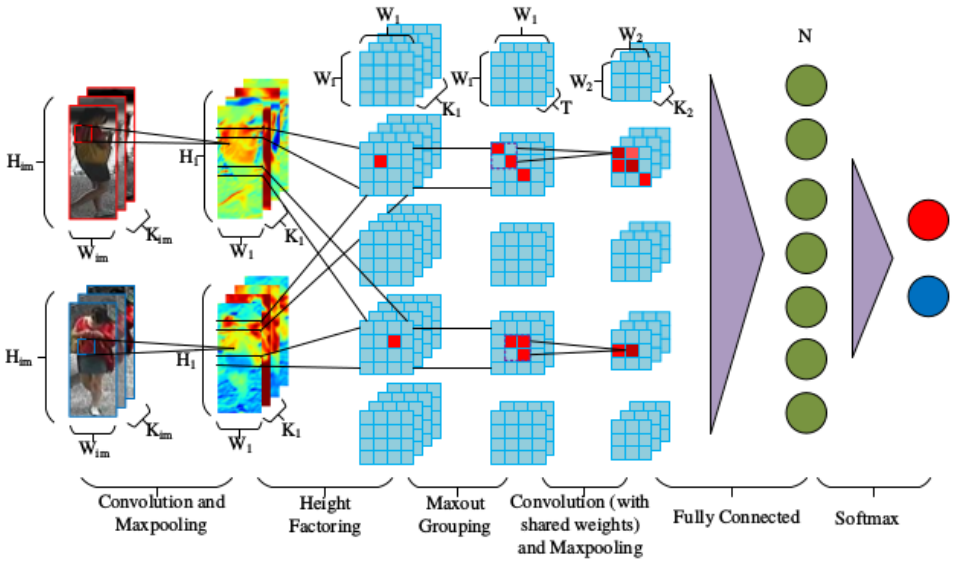

Figura 2.4: Arquitetura proposta por Li e Zhao [2]. A primeira camada combina uma convolução com uma operação de *max pooling* para extrair atributos robustos ao desalinhamento, enquanto a segunda camada cria uma matriz de deslocamento que indica como alinhar as partes do corpo. A terceira camada é uma camada de ativação, já quarta camada é novamente de convolução com uma operação de *max pooling*. A quinta camada é formada por uma operação que conecta todos os atributos da camada anterior para compensar o peso baixo que foi dado na análise do par de imagens em conjunto. Por fim, a última camada utiliza uma ativação *softmax* para traduzir o resultado final em uma probabilidade. Reproduzida de [2]. ©2014 IEEE.

Ainda em 2014, outro trabalho no campo das redes neurais foi proposto por Yi e Lei [28]. Nesse trabalho, eles propuseram uma rede neural siamesa, ou seja, que recebe duas imagens de entrada, passa essas imagens por duas sub-redes neurais com os pesos compartilhados e compara as saídas dessas duas redes, a função de comparação deve resultar em um valor baixo caso as entradas sejam imagens da mesma pessoa e um valor alto caso sejam imagens de pessoas distintas. Com isso, o método proposto consegue aprender atributos de cores, textura e métricas em uma estrutura unificada.

Recentemente, as técnicas baseadas em redes neurais superam bastante todas as outras técnicas, tendo obtido melhorias de mais de 50% nos resultados, como indicado em [3].

2.1.2 Trabalhos recentes que se assemelham à este

Os trabalhos recentes de re-identificação de pessoas têm apostado no poder das redes modularizadas e alcançado bons resultados. A rede *Spindle Net* [22] propõe uma etapa de detecção de partes do corpo (cabeça, tronco, braços, pernas, quadril) e geração de uma imagem para cada parte do corpo. Essas imagens passam por convoluções distintas e são agrupadas em uma camada final de

concatenação para gerar o vetor de características da imagem da pessoa. Já a rede *Attention-Aware Compositional Network*(AACN) [29] é bem parecida com a *Spindle Net*. No entanto, as partes do corpo são detectadas utilizando segmentação para diminuir a influência do fundo das imagens. A rede *Multi-Level Factorisation Net* (MLFN) [6] inclui ativações intermediárias da rede em uma camada de fusão final. Isso ajuda a rede levar parâmetros mais gerais da pessoa em consideração na geração do vetor de características.

Neste trabalho, será utilizada a rede *resnet 50* como *backbone* em todos os testes, pois por mais que as arquiteturas modularizadas apresentem bons resultados quanto a avaliações no mesmo domínio, acredita-se que os pontos abordados pelas redes modularizadas (segmentação, detecção, uso de informações mais gerais) podem ser aprendidos por uma rede não modularizada em um treinamento bem sucedido.

Outra linha de estudo é o uso de GANs para auxiliar nos treinamentos das redes neurais de re-identificação de pessoas. Zhong et al. [30] propuseram o uso de uma GAN cíclica para aproximar as imagens de uma câmera para as outras câmeras de uma mesma base de dados. Essa transformação é utilizada como técnica de aumento de dados durante o treinamento para ajudar a rede a aprender um vetor de características que seja indiferente as vistas da base de dados. Já Deng [20] é proposto o método que utiliza uma GAN cíclica para aproximar imagens entre duas base de dados. Esse método é utilizado como pré-processamento de uma base de dados na tentativa de aprender a gerar vetores de características em outra base de dados de uma forma não supervisionada. Neste trabalho, as técnicas de Zhong não serão utilizadas, pois o uso das GANs será feito com enfoque na adaptação de domínio de forma não supervisionada como proposto por Deng.

Na tentativa de aprender parâmetros mais robustos durante o treinamento das redes neurais, Zhong et al. [31] propuseram um método de aumento de dados, apagando uma região da imagem, de forma aleatória, para simular possíveis oclusões. Já Xiao et al. [32] propuseram o método que consiste em treinar uma rede base em uma super base de dados composta por várias bases agrupadas. Depois aplica-se um *dropout* guiado para diminuir a influência de neurônios que são muito específicos de uma base de dados só. O objetivo é eliminar os neurônios que só são ativados em imagens de uma base de dados específica, ficando apenas com os neurônios gerais que conseguem gerar vetores de características discriminantes entre todas as bases de dados.

2.2 Arquiteturas de redes neurais

Como indicado na seção anterior, as técnicas atuais que apresentam os melhores resultados, para o desafio de re-identificação de pessoas, são as técnicas que utilizam redes neurais. No entanto, existe uma grande quantidade de redes neurais diferentes na literatura, cada uma com suas características próprias. Nesta seção serão apresentadas dois tipos de arquiteturas que são utilizadas para esse desafio: 2.2.1) as arquiteturas residuais (estado da arte em vários desafios de visão computacional) e 2.2.2) arquiteturas modularizadas (apresentam ótimos resultados para o desafio de re-identificação de pessoas).

2.2.1 Arquiteturas residuais (*residual networks*)

Com o crescente uso de redes neurais, cada vez mais profundas, foi detectado um fenômeno relacionado a profundidade das redes. Esse fenômeno indica que, para as redes simples, quanto mais profunda for a rede, mais a acurácia do treinamento satura e os resultados são inferiores aos das redes mais rasas. Em [5], foi detectado que a substituição dos blocos simples de aprendizado por blocos residuais resolvia esse problema, alcançando resultados melhores ao utilizar redes neurais mais profundas. A Figura 2.5 mostra a diferença entre uma rede neural simples que era utilizada e a mesma rede com os blocos residuais adicionados.

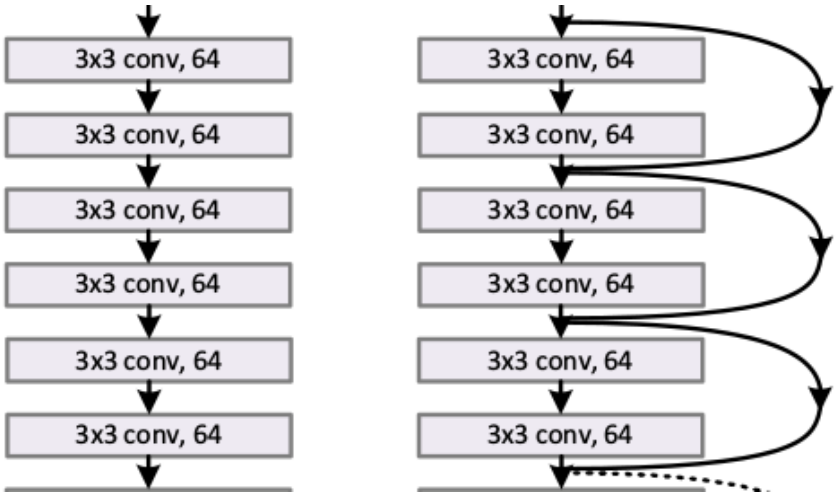

Figura 2.5: Diferença entre redes neurais simples e redes neurais que utilizam blocos residuais. A imagem da esquerda mostra os blocos simples de convolução que eram utilizados antes das arquiteturas residuais, já a imagem da direita mostra como funcionam as redes com blocos residuais. Reproduzida de [5]. ©2016 IEEE.

A ideia dos blocos residuais é que ao invés de aprender uma função $H(x)$ que mapeia a entrada x diretamente na saída y, o bloco residual define uma função $F(x) = H(x) - x$ que também pode ser vista como $H(x) = F(x) + x$. A hipótese apresentada em [5] é que, se o *identity mapping* (mapeamento da identidade) for ótimo, pode-se aprender $F(x) = 0$ e com isso teria-se $H(x) = x$. Ou seja, a saída seria igual a entrada que é o resultado esperado quando o mapeamento é ótimo, e é muito mais simples para a rede aprender do que precisar aprender uma função de mapeamento nova. A Figura 2.6 ilustra o funcionamento de um bloco residual.

2.2.2 Arquiteturas modularizadas

Grande parte das redes neurais é utilizada de forma que o usuário ao inserir uma imagem de entrada analisa o resultado na saída da rede neural. No entanto, as camadas intermediárias das redes neurais também podem aprender algumas características interessantes da imagem e que poderiam ser úteis para o resultado final. Por exemplo, uma rede neural treinada para gerar uma assinatura digital da imagem da pessoa vai apresentar a assinatura como resultado final, mas pode ser que em camadas intermediárias a rede neural aprendeu que aquela imagem era de uma mulher,

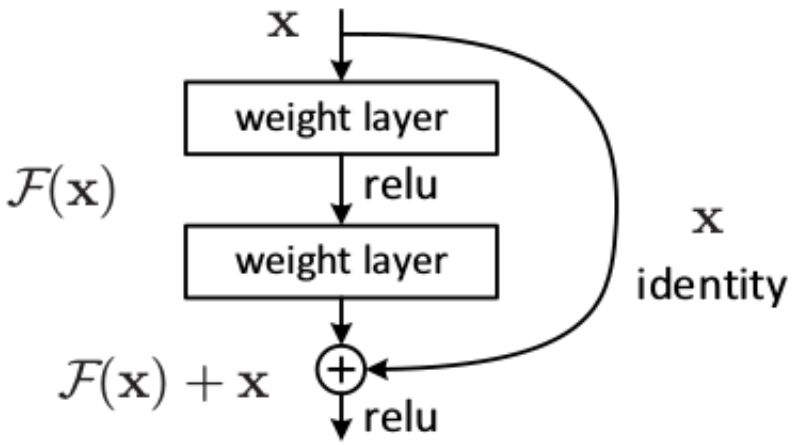

Figura 2.6: Funcionamento de um bloco residual. Reproduzida de [5]. ©2016 IEEE

loira e magra. Esse tipo de informação pode ser benéfico se concatenado ou mesclado com a assinatura digital final.

A ideia das arquiteturas modularizadas é, também, utilizar os resultados intermediários das redes neurais para compor a assinatura (saída). Um exemplo ótimo de arquitetura modularizada seria uma rede neural que em suas camadas mais rasas, ou em seus módulos iniciais, aprenda a segmentar a imagem da pessoa. Ou seja, aprenda a criar um mapa de atenção no qual os *pixels* que pertencem à pessoa propriamente dita têm um peso alto e os *pixels* que pertencem ao fundo (*background*) da imagem têm um peso baixo. Desta forma, o mapa de atenção poderia ser aplicado na assinatura digital (saída) para filtrar e retirar os atributos que não pertencem à pessoa propriamente dita.

Em [6], Chang propõe uma rede neural altamente modularizada que utiliza informações de várias camadas superficiais para gerar a assinatura digital da pessoa. Ele alcança o estado da arte em várias bases de dados para provar a eficiência das arquiteturas modularizadas no desafio de re-identificação de pessoas. A Figura 2.7 mostra a arquitetura de rede neural proposta por Chang.

2.3 Divisão dos dados

Para treinar as redes neurais, separa-se os dados das bases de dados em três conjuntos disjuntos. Seguem os conjuntos normalmente utilizados e a funcionalidade de cada um deles:

- **Conjunto de Treinamento:** É o conjunto que armazena as imagens que serão utilizadas para o treinamento da rede neural, ou seja, são imagens que são colocadas na entrada da rede neural e que sabe-se a resposta esperada para comparar com a saída da rede. A função de custo é calculada a partir da diferença dos resultados. Esse conjunto foi formado por 65% das identidades (pessoas distintas) da base de dados *CyberQueue*;
- **Conjunto de Validação:** É um conjunto de suporte utilizado durante o treinamento da

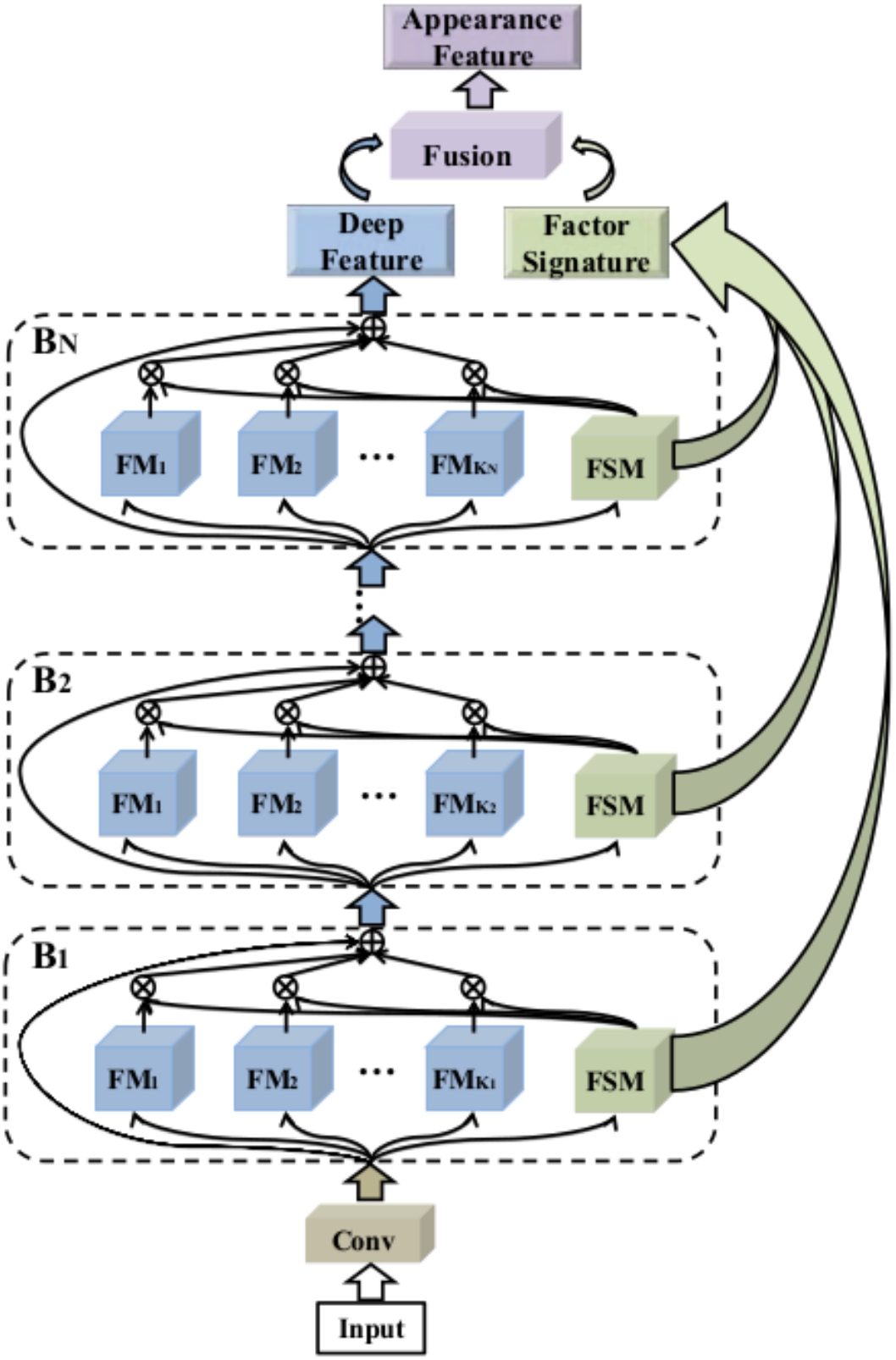

Figura 2.7: *Multi-Level Factorisation Net* (MLFN). Reproduzida de [6]. ©2018 IEEE.

rede neural. Ele funciona como um conjunto de testes durante a etapa do treinamento. Ou seja, ao final de cada época de treinamento, realiza-se um teste da acurácia no conjunto de validação para verificar se a rede neural está alcançando uma boa generalização ou se está ficando viciada nas imagens do conjunto de treinamento. Esse conjunto foi formado por 5% das pessoas da base de dados *CyberQueue*;

- **Conjunto de Testes:** Como o nome diz, esse é o conjunto de testes utilizado para verificar a eficiência alcançada pela rede neural. É importante que esse conjunto seja sempre o mesmo para facilitar a comparações dos resultados de diferentes redes neurais. Esse conjunto foi formado por 30% das pessoas da base de dados *CyberQueue*.

Para as bases de dados públicas, é interessante utilizar a mesma divisão de conjuntos que os outros autores utilizam, porque isso permite comparar os resultados obtidos com o estado da arte.

2.4 Técnicas de aumento de dados (*data augmentation*)

As técnicas de aumento de dados são muito utilizadas para ajudar as redes neurais a conseguir generalizar o aprendizado durante o treinamento, para que não fique muito viciada nas imagens do conjunto de treinamento. A ideia das técnicas de aumento de dados está em aplicar algumas transformações na imagem de entrada de forma que ela não perca sua essência, mas apresente a mesma informação de uma maneira diferente. Algumas das principais técnicas de aumento de dados utilizadas em redes neurais são:

- ***Flipping*:** Se a classe que deseja-se aprender apresentar uma simetria vertical ou horizontal, pode-se aplicar um espelhamento na imagem para gerar novas imagens que pertencem a mesma classe;
- ***Rescaling*:** Pode-se aplicar aleatoriamente um aumento ou diminuição na imagem de forma que mude o tamanho da região de interesse onde a classe está presente na imagem;
- **Rotação:** Ao rotacionar, com um baixo grau, a imagem, essa não costuma perder suas características essenciais

No entanto, dessas técnicas tradicionais, a única interessante para o desafio de re-identificação de pessoas é a de *flipping* horizontal, pois isso não vai mudar a essência da aparência da pessoa. A técnica de *rescaling* não se aplica porque as imagens são geradas utilizando um detector de pessoas invariante à escala. Já a técnica de rotação não faz sentido, pois assume-se que as câmeras têm uma orientação padrão.

As técnicas de aumento de dados utilizadas para este desafio estão relacionadas as dificuldades de re-identificação de pessoas. Uma das grandes dificuldades relacionadas a esse problema está na ocorrência de grandes oclusões na imagem. Para combater essa dificuldade, uma técnica interessante de aumento de dados é retirar, aleatoriamente, uma parte da imagem e substituir ela

por *pixels* aleatórios para gerar oclusões na imagem da pessoa, esse método foi proposto em [31]. Outra grande dificuldade está no aprendizado de vetores de características que sejam robustos às variações de vistas, para isso Zhong [30] propôs a utilização de uma GAN (*Generative adversarial network*) que projeta as imagens de uma câmera na vista das outras câmeras da base de dados.

2.5 Funções de custo

Várias aplicações de redes neurais no campo da visão computacional tem a classificação como objetivo final. No entanto, o desafio de re-identificação de pessoas não é um desafio de classificação, mas sim um desafio de encontrar similaridades entre amostras. Para isso, deve-se utilizar funções de custo que promovam esse tipo de análise.

Funções siamesas são utilizadas para comparar as respostas para duas amostras distintas. O objetivo dessas funções é que as amostras de classes diferentes sejam distantes e as amostras da mesma classe sejam próximas. No entanto, essas funções de custo apresentam uma falha. Pois, garantir que duas amostras de mesma classe sejam próximas, não garante que elas sejam distantes de amostras de outras classes, podendo induzir a erros.

Logo, a função de custo mais utilizada na literatura de re-identificação de pessoas é a *triplet*. Pois, essa função leva em consideração uma imagem de mesma classe e outra imagem de classe distinta quando vai treinar uma *anchor* (imagem). Com isso, ela consegue aproximar os semelhantes e afastar os diferentes ao mesmo tempo, gerando uma maior segmentação na resposta. Qualquer função de comparação pode ser utilizada junto à *triplet*, contudo não há ganhos ao se usar funções de comparação muito complexas, logo a função de comparação utilizada pela *triplet* é a distância euclidiana, definida na Equação 2.1.

$$D(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{P} (x_i - y_i)^2} \tag{2.1}$$

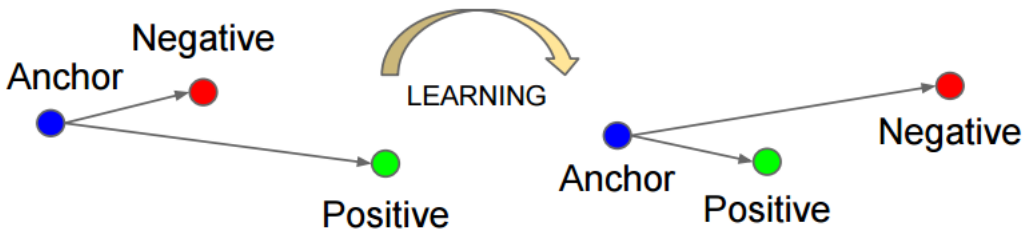

Figura 2.8: Durante o treinamento, a função de custo *triplet* faz com que a distância entre a imagem de treinamento (*anchor*) diminua para outras imagens da mesma pessoa e aumente para imagens de outras pessoas. Reproduzida de [7]. ©IEEE 2015.

A Figura 2.8 ilustra o comportamento que a função de custo *triplet* gera no treinamento da rede neural. Em 2014, Wang et al. [33] propuseram o primeiro treinamento de uma rede neural com base

na função de custo *triplet*. Para obter o comportamento da Figura 2.8, foi utilizada a Equação 2.2 que define o valor da função de custo *triplet*, onde a função $f(\cdot)$ representa a rede neural, logo sua saída será um vetor de características de 128 dimensões. Já a função $D(\cdot)$ é a distância euclidiana definida na Equação 2.1, p é o *anchor*, p^+ é o exemplo positivo e p^- é o exemplo negativo. Por fim, m é um *offset* para evitar que o erro seja zerado quando for assumido que os exemplos negativos e os exemplos positivos tiverem a mesma distância em relação ao *anchor*.

$$L = \max\Big(0\ ,\ m + D\Big(f(p), f(p^+)\Big) - D\Big(f(p), f(p^-)\Big)\Big) \tag{2.2}$$

2.6 Taxa de Aprendizado (*learning rate*)

A taxa de aprendizado é um parâmetro que determina o tamanho do passo de atualização dos pesos da rede neural. Esse é um dos hiper parâmetros que devem ser escolhidos com cuidado na hora de treinar uma rede neural, pois ela pode ter uma grande influência se o seu treinamento irá convergir ou não. Taxas de aprendizado muito baixas são ótimas para encontrar mínimos locais e taxas de aprendizado mais altas são interessantes para procurar outros locais que possam ter mínimos menores que o local anterior. No entanto, ambos os casos apresentam seus ônus, no caso das taxas de aprendizado muito pequenas, elas podem levar o resultado para um mínimo local pior que o mínimo global da função de custo. A taxa de aprendizado muito alta pode gerar uma divergência no treinamento, ambos os casos estão ilustrados na Figura 2.9.

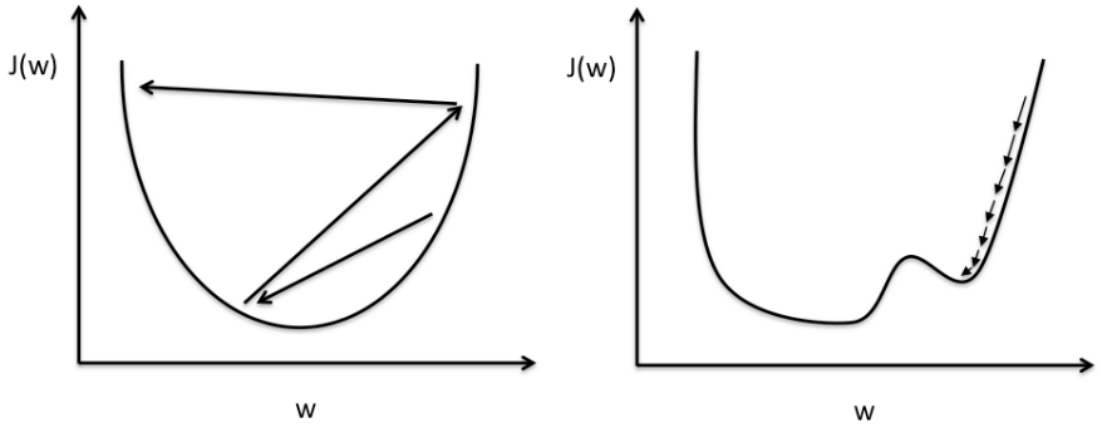

Figura 2.9: Efeitos da taxa de aprendizado na minimização da função de custo. O gráfico da esquerda mostra uma divergência no treinamento causado por uma taxa de aprendizado alta, e o gráfico da direita mostra a perda de um mínimo global por causa do uso de uma taxa de aprendizado muito baixa. Reproduzida de [8].

2.6.1 Taxas de aprendizado cíclicas

Os métodos tradicionais de escolha da taxa de aprendizado são baseados em força bruta (testar várias taxa de aprendizado diferentes e analisar qual entrega uma melhor qualidade) e variam

muito com o com a experiência do usuário, logo esses métodos não são ótimos e apresentam um grande custo computacional e uma demora para otimizar esse parâmetro.

Visto isso, Smith [9] apresentou um novo método para escolha da taxa de aprendizado inicial e para determinar a atualização dessa durante o treinamento. O seu método consiste em criar uma atualização cíclica da taxa de aprendizado conforme dito a seguir:

1. **Otimização da taxa de aprendizado:** consiste em iniciar o treinamento com uma taxa de aprendizado bem pequena e ir aumentando essa taxa de aprendizado linearmente por algumas épocas até que a taxa de aprendizado se torne tão alta que o treinamento comece a divergir, nesse momento o treinamento deve ser parado e deve-se gerar um gráfico que relacione a acurácia do treinamento com a taxa de aprendizado.

2. A partir do gráfico gerado no item 1, deve-se escolher o limite superior e inferior da taxa de aprendizado no ciclo de atualização. O limite superior do ciclo será definido pelo maior valor da taxa de aprendizado que ainda fazia o treinamento convergir e o limite inferior será definido pelo menor valor de taxa de aprendizado que fazia o treinamento convergir. Esses limites podem ser visualizados na Figura 2.10 como *max_lr* e *base_lr*.

3. Escolher o tamanho do passo *(stepsize)*. O *stepsize* diz quantas iterações (ou épocas) o treinamento vai fazer antes da *learning rate* voltar a aumentar/diminuir. A Figura 2.10 ilustra esse parâmetro em verde.

4. Treinar sua rede neural com uma taxa de aprendizado cíclica por alguns ciclos (Smith acredita que de 3 e 5 ciclos devem ser o suficiente para a convergência do treinamento).

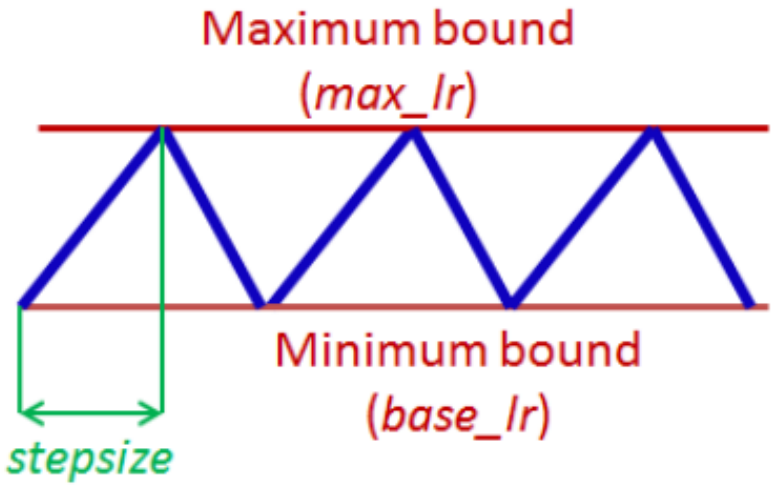

Figura 2.10: A taxa de aprendizado triangular. Reproduzida de [9]. ©2017 IEEE.

É intuitivo entender o porque as taxas de aprendizado cíclicas funcionam quando se pensa na topologia da função de custo. Pois, a parte do ciclo que contém altas taxas de aprendizado permite que o treinamento "salte" entre dois vales da topologia da função de custo, e as baixas taxas de aprendizado permitem explorar os mínimos locais desses vales. Logo, a cada ciclo um novo vale

pode ser investigado e o treinamento guarda informações sobre a acurácia no mínimo local de cada vale. Portanto, ao fim do treinamento os pesos da rede que levaram ao vale com melhor mínimo local são salvos.

No entanto, Dauphin [34] argumenta que o problema das taxas de aprendizado baixas está principalmente nos pontos de sela e não em mínimos locais ruins, pois os pontos de sela apresentam um gradiente pequeno e "passar"por essas regiões com uma taxa de aprendizado pequena é um processo lento.

Em seu trabalho, Smith [9] testou vários métodos diferentes para reduzir/aumentar a taxa de aprendizado dentro de um ciclo e para diminuir a amplitude em ciclos secundários, no entanto todos esses testes apresentaram resultados parecidos, logo ficou recomendado utilizar a taxa triangular com uma redução/aumento linear para facilitar a implementação.

2.6.2 Super convergência

Apesar de todo o sucesso que as redes neurais profundas têm tido em diversos campos nos últimos anos, Smith acredita que esses resultados têm sido obtidos com o uso de hiper parâmetros sub-ótimos e propõe diversas técnicas para determinar esses hiper parâmetros de modo a reduzir o tempo de treinamento e aumentar a performance das redes neurais [35].

Quanto a taxa de aprendizado, Smith propõe um método chamado super convergência que é muito parecido com o método da taxa de aprendizado cíclica, no entanto na super convergência apenas 1 ciclo é realizado. Logo, a taxa de aprendizado é inicializada com o valor do limite inferior e cresce linearmente até atingir um valor de ordem de grandeza maior que o limite superior, por fim a taxa de aprendizado é reduzida linearmente até alcançar valores menores que o limite inferior e se mantém nessa ordem de grandeza até o fim do treinamento. Smith define esse método de super convergência como um tipo de regularização que para funcionar corretamente deve estar em harmonia com outros métodos de atualização utilizados como, por exemplo, *weight decay*.

2.7 Transferência de Aprendizado (*transfer learning*)

O processo de treinamento de uma rede neural consiste em aprender os pesos ótimos para cada um dos neurônios. Os pesos ótimos são aqueles que fazem a saída da rede neural segmentar corretamente as classes que desejamos classificar, ou seja, para um dado da classe A de entrada, a saída da rede neural apresentará uma resposta muito característica da própria classe A. A função de custo aponta a direção que deve ser seguida para atualizar os pesos e o otimizador faz essa atualização dos pesos em busca de pesos ótimos. No entanto, quais pesos são ideais para inicializar a rede neural?

Não existem pesos ideais para se inicializar uma rede neural. Quaisquer que sejam os pesos iniciais, o otimizador vai adapta-los a cada época com o intuito de reduzir a função de custo. No entanto, se os pesos iniciais estiverem mais perto dos pesos ótimos, esse processo de minimizar a função de custo será mais rápido, e pode gerar um resultado ainda melhor que os resultados

gerados por inicializações aleatórias.

Para melhorar a inicialização dos pesos pode ser utilizada a técnica de *fine tunning* que é uma técnica de transferência de aprendizado. Uma vez que as camadas mais superficiais das redes neurais aprendem características gerais da imagem. Espera-se que ao utilizar os pesos dessas camadas treinadas em um domínio A, em um domínio B, vão produzir resultados razoáveis. Portanto, esta técnica de transferência de aprendizado consiste em inicializar os pesos da rede neural com pesos de outra rede neural que foi treinada a priori.

2.7.1 Adaptação de domínio

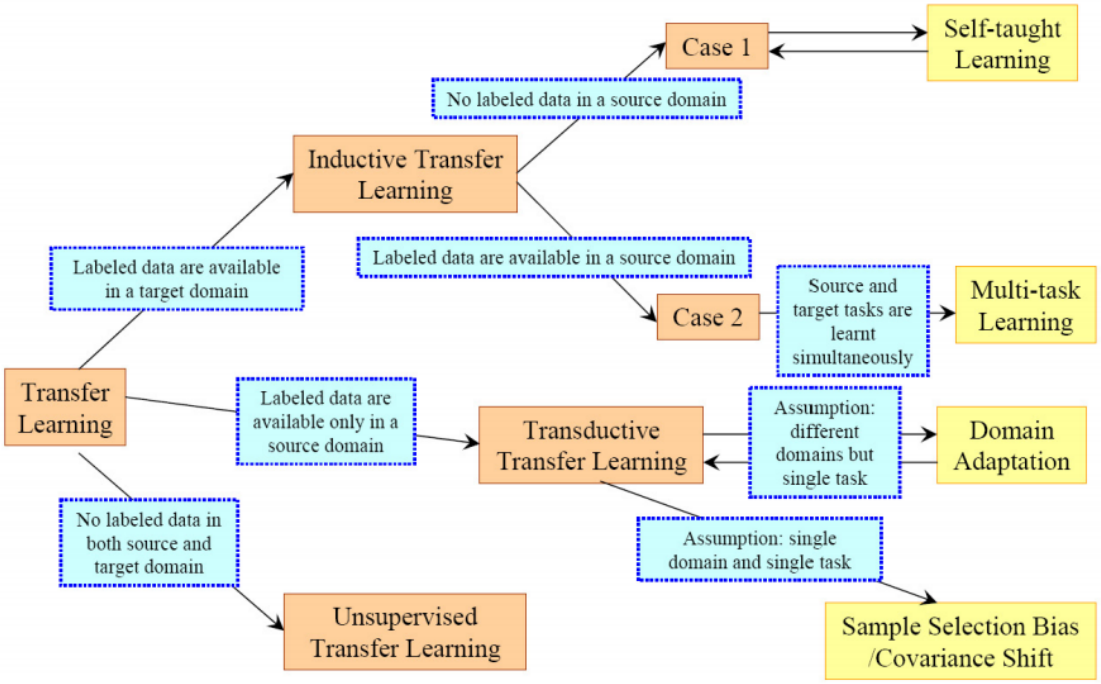

Figura 2.11: Classificação dos tipos de transferência de aprendizado definidos por Pan e Yang. Reproduzida de [10]. ©2010 IEEE.

Em 2009, Pan e Yang [10] realizaram uma pesquisa profunda no tema transferência de aprendizado e classificaram suas diversas formas, conforme a Figura 2.11. Portanto, a adaptação de domínio pode ser vista como uma subcategoria da transferência de aprendizado, onde se tem dois domínios distintos mas uma mesma tarefa.

Seguindo a notação de Pan e Yang, tem-se que um domínio D é composto por um espaço d-dimensional $X \subset R^d$ com uma função de distribuição de probabilidade $P(X)$ e uma tarefa τ é definida por um espaço de rótulos Y e uma função de distribuição de probabilidade $P(Y|X)$.

Portanto, de acordo com a Figura 2.11 e a notação definida por Pan e Yang, a adaptação de domínio é uma transferência de aprendizado transdutiva, onde é assumida a mesma tarefa $\tau^f = \tau^a$. Todavia, normalmente esse método é associado com uma tarefa de classificação, onde tanto o espaço de rótulos quanto a função de densidade de probabilidade são assumidas como

iguais entre os domínios fonte e alvo. Ou seja, é assumido que $Y^f = Y^a$ e $P(Y|X^f) = P(Y|X^a)$. No entanto, em casos reais, como o caso de re-identificação de pessoas estudado nesse trabalho, essa segunda afirmação não costuma ser verdadeira segundo Csurka [36]. Em sua pesquisa, Csurka relaxou a definição de adaptação de domínio para os casos onde apenas seja verdade a afirmação $Y^f = Y^a$.

2.8 Generative Adversarial Networks (GANs)

As GANs, primeiramente propostas por Goodfellow et al. em [37], são um *framework* para treinamento de redes geradoras a partir de um processo contraditório. O processo consiste em treinar, simultaneamente, duas redes: uma rede geradora G que aprende a função de densidade de probabilidade dos dados de treinamento, e uma rede discriminativa D que estima a probabilidade de um dado pertencer aos dados de treinamento ao invés de ter sido gerado por G. Diz-se que esse processo é contraditório, pois o objetivo do treinamento da rede geradora G é maximizar a probabilidade da rede discriminativa D cometer um erro, ou seja, o objetivo é que G aprenda a representar o conjunto de dados tão bem que a rede D não consegue distinguir se um dado é proveniente de G ou do conjunto real de dados.

A entrada de G é um ruído aleatório e sua saída é um dado que tenta representar o conjunto de dados de treinamento, já em D, tem-se um dado na entrada (esse dado pode ser proveniente tanto de G quanto do conjunto de treinamento) e a saída é um valor no intervalo $(0, 1)$ que representa a probabilidade do dado ser real do conjunto de treinamento. A Figura 2.12 ilustra essa relação entre as redes G e D, em um contexto onde o uso da GANs foi feito para gerar imagens.

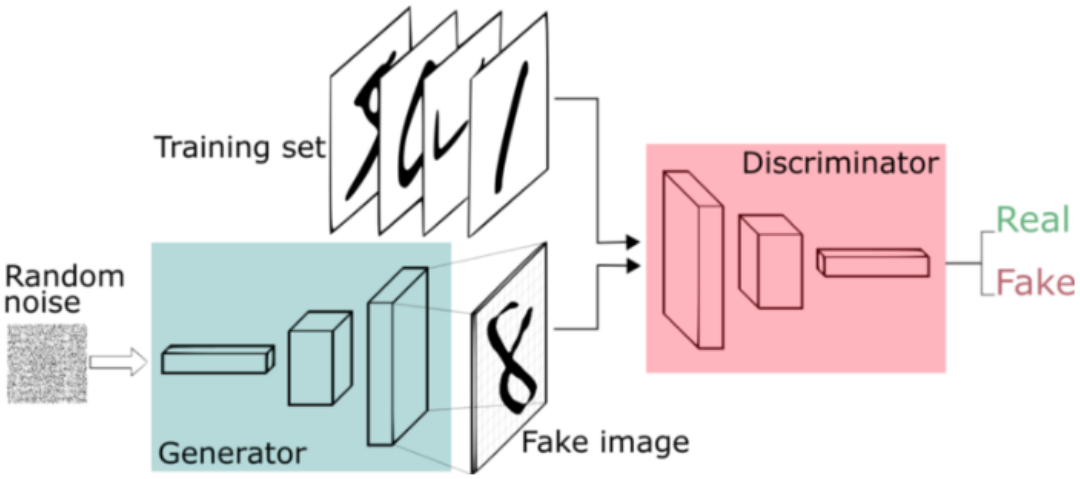

Figura 2.12: Fluxo de dados nas redes geradora e discriminativa de uma GAN. Reproduzida de [11].

Portanto, G pode ser vista como uma função de mapeamento $G(z, \theta_g)$, onde z é um ruído aleatório proveniente de uma distribuição p_z, e θ_g são os pesos da rede G, que mapeia p_z em p_g ao mesmo tempo que aproxima p_g de p_{dados} (distribuição real dos dados de treinamento). Enquanto

D é uma função $D(x, \theta_x)$, onde x é um dado que pode ser de p_g ou p_{dados}, e θ_d são os pesos da rede D. Logo, o treinamento se dá a partir de um jogo *minimax*, onde deseja-se minimizar a diferença entre p_g e p_{dados}, e maximizar o erro de $D(x, \theta_x)$, como definido na Equação 2.3.

$$\min_{\theta_g} \max_{\theta_d} \left[E_{x \sim p_{dados}} \log D_{\theta_d}(x) + E_{z \sim p_z(z)} \log \left(1 - D_{\theta_d}(G_{\theta_g}(z))\right)\right] \tag{2.3}$$

Durante o treinamento, a Equação 2.3 é dividida nas equações 2.4 e 2.5. Pois, o treinamento vai alternar em momentos de adaptação dos pesos para maximizar o erro de D (utilizando subida de gradiente na Equação 2.4), e em momentos que visam minimizar a diferença entre p_g e p_{dados} (utilizando descida de gradiente na Equação 2.5).

$$\max_{\theta_d} \left[E_{x \sim p_{dados}} \log D_{\theta_d}(x) + E_{z \sim p_z(z)} \log \left(1 - D_{\theta_d}(G_{\theta_g}(z))\right)\right] \tag{2.4}$$

$$\min_{\theta_g} \left[E_{z \sim p_z(z)} \log \left(1 - D_{\theta_d}(G_{\theta_g}(z))\right)\right] \tag{2.5}$$

2.8.1 GAN cíclica

Um problema conhecido da visão computacional é o desafio de transformação de uma imagem de um domínio fonte X^f (ex: imagens de desenhos) em uma imagem de um domínio alvo X^a (ex: imagens do objeto real), como ilustrado na Figura 2.13. No entanto, em várias situações reais não há disponibilidade de imagens pareadas entre os dois domínios, ou seja, tem-se várias imagens de ambos domínios. Mas essas imagens intra-domínios não tem relações umas com as outras. Portanto, é necessário aprender uma função de mapeamento $G : X^f \rightarrow X^a$ que aproxime a distribuição $G(X^f)$ da distribuição X^a sem ter exemplos de como seria a imagem de X^f em X^a.

Figura 2.13: Exemplo de transformação de uma imagem de desenho para uma imagem do objeto real. Reproduzida de [12]. ©2017 IEEE.

O fato de não ter imagens pareadas entre os dois domínios aumenta muito a complexidade do problema, pois há infinitas funções $G(\cdot)$ que podem fazer esse mapeamento e apresentar um resultado quantitativo interessante durante o treinamento de uma GAN. No entanto, não necessariamente esses resultados serão qualitativamente aceitáveis. Para tentar resolver esse problema e apresentar resultados qualitativamente aceitáveis, Zhu et al. [13] propuseram o conceito da GAN cíclica (Figura 2.14), onde além de aprender a função $G : X^f \rightarrow X^a$ há também o aprendizado de uma função $F : X^a \rightarrow X^f$ e adiciona-se á função de custo um erro cíclico para que $F(G(X^f)) \approx X^f$ e vice-versa. Dessa forma foi possível obter ótimos resultados de transformação de imagens entre domínios sem a necessidade de uma anotação das imagens intra-domínios.

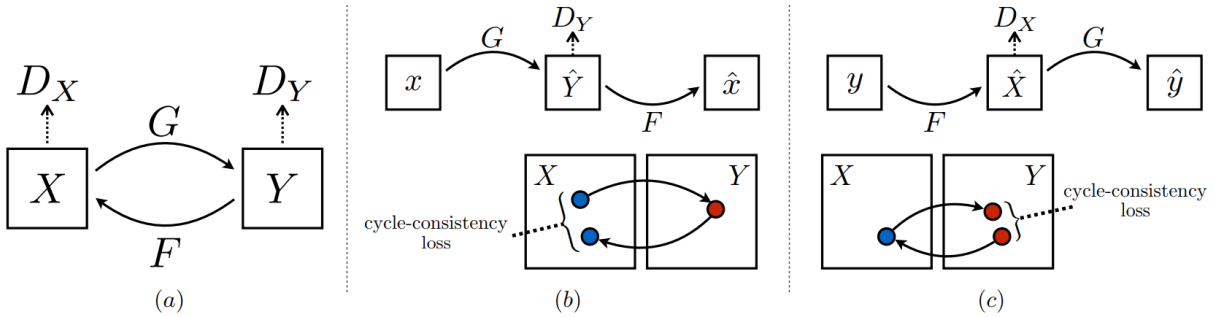

Figura 2.14: Descrição do funcionamento do modelo da GAN cíclica. (a) Modelo da GAN cíclica com duas funções geradoras G e F e duas funções discriminativas D_X e D_{X^a}, onde D_X aprende a distinguir entre imagens de X e de $F(Y)$ e D_Y aprende a distinguir entre imagens de Y e de $G(X)$. (b) Exemplo do efeito do erro cíclico para aproximar $F(G(X)) \approx X$. (c) Exemplo do efeito do erro cíclico para aproximar $F(F(Y)) \approx Y$. Reproduzida de [13]. ©2017 IEEE.

Logo, a função de custo da GAN cíclica é dada pela Equação 2.6, onde L_{GAN} é definida pela Equação 2.3, λ é uma variável para controlar o peso do erro cíclico e L_{cyc} é o erro cíclico que é definido pela Equação 2.7.

$$L(G, F, D_{X^f}, D_{X^a}) = L_{GAN}(G, D_{X^a}, X^f, X^a) + LGAN(F, D_{X^f}, X^a, X^f) + \lambda L_{cyc}(G, F) \quad (2.6)$$

$$L_{cyc}(G, F) = E_{x^f \sim p_{dados}(x^f)}\left[\left\|F(G(x^f)) - x^f\right\|_1\right] + E_{x^a \sim p_{dados}(x^a)}\left[\|G(F(x^a)) - x^a\|_1\right] \quad (2.7)$$

Capítulo 3

Bases de Dados

Cada base de dados utilizada para se treinar uma rede neural apresenta características diferentes. Portanto, é importante fazer um estudo prévio das bases de dados para facilitar o entendimento dos resultados obtidos. Foram utilizadas 4 bases de dados, 3 delas são públicas e foram analisadas na seção 3.1. A seção 3.2 explica o processo de criação de uma base de dados proprietária (*CyberQueue*) e caracteriza essa base criada.

3.1 Estudo das bases de dados utilizadas

Para o treinamento e avaliação das redes neurais desenvolvidas, foram utilizadas quatro bases de dados. Três dessas bases de dados são públicas e a quarta é a base de dados proprietária apresentada na seção 3.2.

Cada base de dados tem características específicas e entende-las é um fator essencial para entender a complexidade de utilizar uma rede em um domínio distinto daquele que foi utilizado para o treinamento. Portanto, segue um estudo das características de cada uma dessas bases de dados que foram utilizadas.

3.1.1 Base de dados *CUHK03*

A base de dados *CUHK03* proposta em [2] conta com 13164 imagens de 1360 pessoas diferentes (uma média de 4.8 imagens para cada vista de cada pessoa), essas imagens foram obtidas por 6 câmeras de segurança diferentes. Mesmo sendo usadas 6 câmeras distintas para a criação da base de dados, cada pessoa teve imagens em apenas 2 câmeras distintas. Essa base de dados apresenta duas variações, uma feita com pessoas anotando a posição da pessoas na imagem e outra com essa anotação sendo feito por um algoritmo de detecção. A variação utilizada foi a com anotação manual.

Uma vez que as imagens são obtidas por câmeras de segurança, elas apresentam variação de iluminação e perdas de partes do corpo das pessoas (esse problema é amenizado quando utiliza-se a versão anotada manualmente). Há uma variação na resolução das imagens dessa base de dados,

devido as vistas diferentes. Elas apresentam uma média de 100x300 *pixels*. A Figura 1.2 mostra uma mesma pessoa vista em duas vistas diferentes nesse base de dados.

3.1.2 Base de dados *Viper*

A base de dados *Viper*, proposta em [14], é a mais antiga utilizada, logo esta é a base de dados que contém o menor número de imagens e pessoas, com apenas 1264 imagens de 632 pessoas distintas. Foram usadas 2 câmeras distintas para a criação da base de dados. No entanto os autores realocaram essas câmeras diversas vezes durante a criação, gerando várias vistas diferentes. Cada pessoa dessa base de dados apresenta imagens de apenas 2 vistas distintas, com uma média de 1 imagem para cada vista de cada pessoa.

A principal variação, que os autores forçaram acontecer, foi na angulação da câmera. Eles utilizaram angulações de 45, 90, 135 e 180 graus para adquirir as imagens das bases de dados. Como o processo de desenvolvimento da base de dados durou vários dias, há variações na iluminação também. Todas as imagens foram extraídas de arquivo de vídeo pré-comprimidos e foram transformadas para terem a mesma resolução de 48x128 *pixels*, mesmo que essa transformação pudesse gerar uma distorção nas imagens. A Figura 3.1 mostra uma mesma pessoa vista em duas vistas diferentes nesse base de dados.

Figura 3.1: Exemplo de imagens de uma mesma pessoa em diferentes câmeras na base de dados *Viper*. Fonte: Base de dados *Viper* [14]. ©2007 IEEE.

3.1.3 Base de dados *Market1501*

A base de dados *Market 1501*, publicada em [15], é a maior e mais recente base de dados pública neste trabalho. Ela contém mais de 32 mil imagens de 1501 pessoas distintas e vistas de 6 câmeras distintas, portanto são aproximadamente 3.6 imagens de cada pessoa por vista. Há pessoas nessa base de dados que foram identificadas em todas as 6 vistas. Ela é a única base de dados utilizada que apresenta essa característica, pois as outras bases podiam ter mais de 2 vistas, mas só se tinha

exemplos da mesma pessoa em 2 vistas diferentes.

Os autores dessa base de dados tinham como objetivo resolver 3 problemas quando a criaram, esses problemas eram: 1) As outas bases de dados eram pequenas para treinar redes muito profundas; 2) As imagens das outras bases de dados eram cortadas e anotadas à mão, diminuindo a realidade do problema; 3) Tinham poucas imagens de exemplos para cada pessoa. Para resolver esses problemas, a base de dados *Market 1501* agrupou um conjunto com mais de 32 mil imagens anotadas e mais de 500 mil imagens de distração (não representam uma pessoa). Todas essas imagens foram adquiridas e anotadas por um detector de pedestres. Por fim, as imagens foram coletadas em um ambiente aberto, adquirindo imagem das pessoas nas 6 vistas distintas. Todas as imagens dessa base de dados apresentam uma resolução de 64x128 *pixels*. A Figura 3.2 mostra uma mesma pessoa vista em todas as 6 vistas diferentes nessa base de dados.

Figura 3.2: Exemplo de imagens de uma mesma pessoa em diferentes câmeras na base de dados *Market 1501*. Fonte: Base de dados *Market 1501* [15]. ©2015 IEEE.

3.2 Criação da base de dados *CyberQueue*

Como indicado na seção 1.3, um dos objetivos desse trabalho era resolver o problema da *CyberLabs*, utilizando técnicas de re-identificação de pessoas. O problema de automatizar o cálculo de tempo de fila para a *CyberLabs* faz parte de um projeto protegido por um acordo de não divulgação. Portanto, nenhuma imagem tanto da base de dados quanto do ambiente onde foi realizado o projeto pode ser divulgada. No entanto, pode-se descrever as características do ambiente e das

imagens coletadas para facilitar a compreensão dos dados e dos resultados obtidos.

3.2.1 Características físicas do ambiente do desafio

Para analisar o tempo de fila, foram utilizadas duas câmeras, uma na entrada e outra na saída da fila. Com isso, podia-se ver as pessoas passando pela entrada e depois rever as pessoas passando pela saída e re-identifica-las para cronometrar o tempo. A Figura 3.3 não é do ambiente real, no entanto representa muito bem como estavam alocadas as câmeras para o projeto. As regiões em cinza ilustram as áreas de visão das câmeras, a área A representa a câmera na entrada da fila e a área B representa a câmera na saída da fila.

Ambas as câmera utilizadas tinham resolução *full HD* (1920×1080 *pixels*), no entanto as laterais da vista da câmera filmam as paredes e/ou regiões fora da fila, logo as áreas de interesse tinham uma resolução de 800×1080 *pixels* e 400×1000 *pixels* na entrada e na saída, respectivamente.

A base de dados resultante apresenta vistas de 3 câmeras, no entanto o projeto inteiro foi realizado com apenas 2 câmeras. A terceira vista existente na base de dados foi resultado de um reposicionamento que houve na câmera de saída durante o projeto. Portanto as imagens da posição antiga ficaram como uma vista e as imagens da nova posição entraram na base de dados como uma vista nova.

3.2.2 Detectores de pessoas e de rostos

Na abordagem proposta para a *CyberLabs*, o primeiro passo é a anotação ou a regionalização automática de uma pessoa na imagem. Esse passo costuma ser um grande desafio ao utilizar os códigos de re-identificação de pessoas no mundo real. Na maioria das bases de dados acadêmicas, as imagens foram obtidas por uma anotação manual que é mais precisa e com erros desprezíveis. No entanto, o processo de identificação de pessoas utilizando redes neurais não é tão preciso e os erros são inevitáveis. Logo, essa é a primeira característica diferencial da base de dados *CyberQueue*, pois ela contém imagens de pessoas "cortadas"(imagens que só pegam o corpo da pessoa da cintura pra cima, ou que só pegam a parte direita/esquerda do corpo).

Para identificar as pessoas nas imagens foi utilizada uma arquitetura de rede neural chamada *faster R-CNN* [17], com a rede neural base *resnet 50* [5]. Esta rede neural faz parte do repositório público do *Google*.

Historicamente, arquiteturas bases de redes neurais, como a *resnet 50*, eram arquiteturas treinadas para fazer classificação de imagens, ou seja, dada certa imagem, essas arquiteturas diziam a qual classe essa imagem pertencia. No entanto, as redes neurais existentes não conseguiam indicar onde na imagem se encontravam aquelas classes. Por isso foram criadas redes neurais que propunham regiões interessantes nas imagens (RPN - *region proposal network*). Portanto, para detectar um objeto na imagem eram usadas duas redes neurais. A rede RPN usada para propor regiões da imagem onde poderiam haver objetos e a rede com arquitetura base, como a *resnet 50*, que classificava cada uma das regiões, indicando se o objeto desejado estava naquela região ou não.

Figura 3.3: Vista superior da fila de entrada para um *show* do cantor Ed Sheeran na Inglaterra que apresenta características muito parecidas com o ambiente onde foi desenvolvida a base de dados *CyberQueue*. A região A indica a vista da câmera de entrada da fila (ponto inicial de cronometragem) e a região B indica a vista da câmera de saída da fila. Reproduzida de [16].

No entanto, esse processo de utilizar duas redes neurais distintas para fazer a detecção de objetos, era muito custoso computacionalmente e necessitava do treinamento de duas redes neurais distintas. Logo, a arquitetura *faster R-CNN* propôs fazer ambos os processos de classificação e regionalização das áreas interessantes na mesma rede neural. Com isso, o custo computacional foi reduzido significantemente, tanto para o treinamento quanto para a execução das redes neurais. A Figura 3.4 ilustra o funcionamento de um rede *faster R-CNN* para detectar objetos em uma imagem.

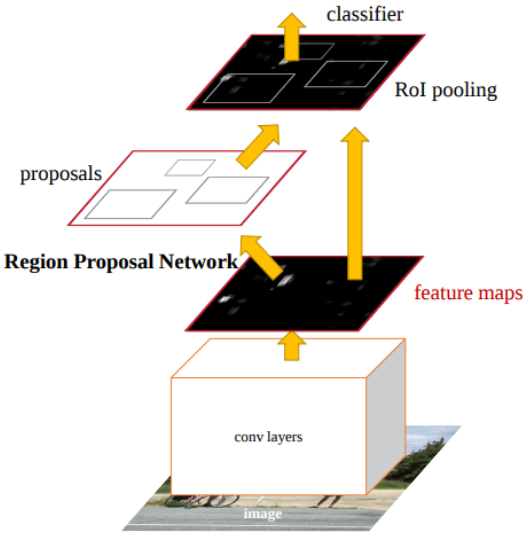

Figura 3.4: Exemplo do funcionamento de uma rede *faster R-CNN*, onde os atributos da rede base convolucional são combinados com a RPN para encontrar regiões interessantes na imagem e classificar essas regiões de acordo com o objeto existente nelas. Reproduzida de [17]. ©2016 IEEE.

Para a primeira etapa desse projeto, foi utilizado reconhecimento facial para resolver o problema. Desta maneira, o próprio funcionamento do programa criado iria gerar imagens para a base de dados de re-identificação de pessoas. Inicialmente, foi utilizada uma *faster R-CNN* de detecção de pessoas , a partir da imagem das pessoas foi utilizada outra *faster R-CNN* para localizar o rosto das pessoas. No entanto, os rostos da pessoas nas imagens reais e dinâmicas das câmeras tinham uma resolução média de apenas 40×40 *pixels* e a rede de localização de rostos utilizada havia sido treinada para imagens entre 80×80 e 160×160 *pixels*. Portanto o resultado não era muito preciso.

3.2.3 Rede neural para fazer o reconhecimento facial

Após identificar os corpos e rostos das pessoas, havia a necessidade de gerar uma assinatura digital da pessoa. Para isso, foi utilizada uma rede neural convolucional com função de custo *triplet*. Essa rede foi treinada para que sua saída fosse um vetor de atributos que representassem uma assinatura digital do rosto da pessoa. A rede neural foi treinada na base de dados pública LFW (*labeled faces in the wild* - faces anotadas em ambiente reais) e atingiu acurácia de 99.13%.

Portanto, o programa recebia as imagens das câmeras e utilizava o detector de pessoas para

gerar sub imagens da imagem principal, onde estava só a pessoa. Na imagem da pessoa gerada era aplicado o detector de faces para gerar outra sub imagem, a imagem do rosto da pessoa. Por fim, a imagem do rosto da pessoa era dada como entrada de uma rede neural convolucional que gerava uma assinatura digital da pessoa. Assim, utilizando-se distância euclidiana nas assinaturas digitais das pessoas era possível agrupar as imagens semelhantes, provavelmente da mesma pessoa (distância euclidiana pequena).

As imagens da câmera de entrada eram utilizadas para gerar esses agrupamentos de assinaturas digitais que indicavam uma pessoa que entrou na fila. Já as imagens da câmera de saída eram utilizadas para gerar assinaturas digitais das pessoas saindo da fila. Ao comparar essas assinaturas com os grupamentos realizados na entrada, quando uma assinatura de saída apresentava uma distância euclidiana muito pequena em relação a um grupamento da entrada, ocorria a re-identificação da pessoa e o tempo que ela permaneceu na fila era medido e armazenado.

3.2.4 Limpeza da base de dados

A base de dados LFW, que foi utilizada, é composta por imagens frontais dos rostos das pessoas. Há pouca variação de iluminação e a resolução das imagens está entre 80×80 e 160×160 *pixels*. Como o projeto utilizava câmeras de segurança, posicionadas acima dos rostos, funcionando o dia inteiro, havia uma variação significativa do domínio. A iluminação variava com a hora do dia e a resolução do corte da imagem nos rostos das pessoas variava com a distância da pessoa até a câmera. Em média um rosto ficava com resolução de 40×40 *pixels*. Neste cenário, o programa obteve resultados aproximados de 40% de acertos na precisão de re-identificação. Logo, a base de dados estaria com muitos falsos positivos.

Para filtrar a base de dados e eliminar os falsos positivos , foi criado um programa em linguagem de programação *Python* para a validação manual do reconhecimento facial. O programa consistia em mostrar a imagem da primeira vez que a pessoa foi vista na entrada ao lado da imagem da última vez que a pessoa foi vista na saída, acompanhadas de algumas informações como horário de entrada, horário de saída e uma predição feita por uma rede de re-identificação de pessoas. O usuário analisava os dados mostrados e pressionava uma tecla para indicar se ambas as imagens eram da mesma pessoa ou não. As imagens corretas eram mantidas na base de dados e as erradas eram deletadas. A Figura 3.5 mostra a interface do programa criado.

3.2.5 Características da base de dados *CyberQueue*

Como indicado nessa seção, a base de dados *CyberQueue* apresenta várias características próprias que a difere bastante das outras bases de dados utilizadas e torna o desafio da criação de uma rede de re-identificação de pessoas ainda mais complexo. Primeiramente, foi utilizado apenas algoritmos para detectar as pessoas, portanto são normais as imagens que aparece só a parte de cima do corpo da pessoa ou que omita outras partes do corpo da pessoa. Além disso, como a base de dados foi criada utilizando reconhecimento facial, há imagens que contém mais de uma pessoa. A identidade da pessoa relacionada a imagem pode não ser da pessoa que aparece em primeiro

Figura 3.5: Interface do programa criado para realizar a validação manual dos resultados do reconhecimento facial. As imagens utilizadas são de bases de dados públicas e foram usadas apenas para ilustração.

plano, mas do rosto de uma pessoa que está em um plano de fundo na imagem.

Outros pontos desafiadores dessa base de dados são a variação na iluminação e a angulação alta da vista das pessoas. O ponto mais desafiador dessa base de dados é que ela contém falsos positivos, relacionados a erros humanos na validação e a uma falha existente no programa de validação. Como dito o programa de validação só apresenta para o usuário a primeira imagem da entrada e a última imagem da saída, no entanto se há mais de uma imagem na entrada ou na saída, essas imagens podem não pertencer a mesma pessoa, mas ela vai entrar na base de dados da mesma maneira.

As imagens dessa base de dados apresentam uma resolução fixa de 150×300 *pixels* que é uma boa resolução quando comparada com as outras bases de dados utilizadas. A base de dados criada contém 56867 imagens de 6261 pessoas diferentes com vistas de 3 câmeras distintas, mas todas as pessoas apresentam apenas imagens de 2 vistas distintas, portanto há uma média de 4.54 imagens de cada pessoa por câmera.

3.2.6 Solução do problema utilizando re-identificação de pessoas

Após a criação da base de dados *CyberQueue*, essa foi utilizada para treinar uma rede de re-identificação de pessoas utilizando a imagem do corpo da pessoa. A metodologia utilizada para treinar essa nova rede de re-identificação de pessoas utilizou a arquitetura *resnet 50* e a função de erro *triplet*. Portanto, a *pipeline* do programa final continuou praticamente igual ao que utilizava o reconhecimento facial, com a única diferença que a rede de detecção de faces e a rede que gerava assinaturas digitais dos rostos foram substituídas pela rede que gera assinaturas digitais das imagens dos corpos das pessoas.

A rede de re-identificação de pessoas criada aumentou a performance do programa que era de 40% para aproximadamente 85%, no entanto erros ainda acontecem. O programa criado para a limpeza da base de dados continua sendo utilizado, eliminando os falsos positivos do resultado final e acrescentando mais imagens a base de dados para que novos treinamentos, que sejam feitos na rede e o resultado dessa melhore ainda mais.

Capítulo 4

Metodologia

Esse capítulo visa explicar os métodos escolhidos para resolver o desafio de re-identificação de pessoas e o porque esses métodos são interessantes. Na seção 4.1 é feita uma comparação entre os métodos de reconhecimento facial e de re-identificação de pessoas e é explicado o porque é necessário estudar o desafio de re-identificação. A seção 4.2 vai apresentar a rotina criada para o treinamento de redes que serão treinadas e avaliadas no mesmo domínio. A seção 4.3 mostra todos os métodos de adaptação de domínio que serão usados. Já a seção 4.4 apresenta um método proposto para resolver um problema comum no treinamento de redes neurais, quando se usa a função de custo *triplet*. Por fim, a seção 4.5 apresenta todas as métricas que serão utilizadas para avaliar os resultados gerados.

4.1 Falha do reconhecimento facial

Hoje, a tecnologia do reconhecimento facial se encontra muito mais desenvolvida que a tecnologia de re-identificação de pessoas. Pode-se afirmar isso dado que a rede neural *DeepFace* [24] criada por Taigman et al. em 2014 (5 anos atrás) obteve uma performance praticamente igual a de um ser humano quando avaliada na base de dados LFW (*Labeled Faces in the Wild*) [18]. A base de dados LFW contém 13233 imagens de 5749 pessoas distintas sendo todas essas imagens pertencentes a situações reais (vide Figura 4.1). A acurácia da rede *DeepFace* nessa base de dados foi de 97.35% enquanto a acurácia humana nessa base de dados é de 97.53%.

Figura 4.1: Exemplos de imagens da base de dados LFW. Reproduzida de [18].

Enquanto o estado da arte de reconhecimento facial está muito próximo ou até um pouco melhor do que a capacidade humana nessa tarefa, o desafio de re-identificação de pessoas atinge taxas de erros próximas a 8% para as bases de dados *CUHK03* e *Market1501*. Então, porque não

usar apenas o reconhecimento facial, ao invés de tentar desenvolver técnicas de re-identificação de pessoas?

A grande diferença entre esse dois desafios é que o reconhecimento facial necessita de imagens de alta qualidade (resolução) e que peguem bem o rosto das pessoas. Enquanto para a re-identificação de pessoas, pode-se trabalhar com imagens de menor qualidade e sem restrição quanto a pose da pessoa, podendo, por exemplo, utilizar imagens de pessoas de costas. Como o foco do desafio de re-identificação de pessoas é utilizar imagens provenientes de CFTV (câmeras de segurança), as técnicas de reconhecimento facial não funcionam bem para esse tipo de dados.

Utilizando imagens da base de dados *CUHK03* e a biblioteca de código aberto *dlib* [38], pode-se visualizar o resultado dos algoritmos de reconhecimento facial nas imagens de CFTV. A Figura 4.2 demonstra o caso onde a pose da pessoa não permite a visualização do rosto (imagem mais a esquerda) e uma situação onde o rosto é visível, mas com baixa qualidade (imagem central). Na imagem mais a direita da Figura 4.2, pode-se ver que os pontos chaves do rosto (rosto da imagem do meio) que foram encontrados estão um pouco amontoados, mas corretos. No entanto, para conseguir detectar esses pontos foi necessário fazer uma interpolação na imagem deixando ela 4 vezes maior que seu tamanho original. Portanto, houve uma inserção de ruído desnecessário na imagem e um aumento no custo computacional do algoritmo.

Figura 4.2: Ilustração da dificuldade de usar reconhecimento facial nas bases de dados de re-identificação de pessoas. *Esquerda:* Exemplo de imagem onde o rosto não é visível. *Meio:* Exemplo de imagem onde o rosto é visível, mas com uma baixa qualidade. *Direita:* Pontos chaves encontrados no rosto da imagem do meio, a imagem do rosto teve que ser interpolada para a detecção desses pontos, por isso apresenta-se distorcida.

As dificuldades apresentadas demonstram que mesmo o reconhecimento facial obtendo ótimos resultados e funcionando para uma gama de casos, há espaço para desenvolver o desafio de re-identificação de pessoas. Pois, em diversos casos, o reconhecimento facial não será suficiente e a re-identificação de pessoas poderá resolver o problema com maestria. Por exemplo, em casos onde

deseja-se fazer rastreamento de pessoas usando as imagens de CFTV.

4.2 Treinamento das redes neurais de re-identificação de pessoas

Como visto na seção anterior, há a necessidade de treinar redes neurais especialistas no desafio de re-identificação de pessoas, portanto o primeiro passo desse trabalho será treinar essas redes. Será treinada uma rede neural para cada base de dados apresentada no capítulo 3 de forma a analisar o potencial do aprendizado supervisionado, utilizando redes profundas nessas bases de dados.

Todas as redes treinadas utilizaram a arquitetura *Resnet 50* que é uma arquitetura residual. Para a função de custo e o otimizador foram escolhidos a *triplet* e o Adam, respectivamente. Antes de fazer o treinamento, todas as imagens de todas as bases de dados foram convertidas para a resolução de 128×256 *pixels*. Esse é um valor médio entre as resoluções das bases de dados utilizadas. A criação de uma resolução padrão facilita os testes e as avaliações em domínios distintos.

Nesses treinamentos, os pesos da rede neural foram inicializados utilizando os pesos de uma rede treinada na base de dados *ImageNet* [39]. Portanto, foi utilizada a técnica de *fine tunning* em todos os treinamentos. Essa escolha foi feita porque essa base de dados é muito ampla pela quantidade de classes que ela contém. Os pesos de suas camadas superficiais são gerais e funcionam para praticamente qualquer problema e com isso consegue-se iniciar o treinamento de uma etapa mais avançada, sem precisar aprender esses pesos das camadas mais superficiais [40].

Foi utilizada uma taxa de aprendizado inicial de 0.0002, que é uma taxa de aprendizado relativamente pequena. Para garantir a convergência, foi utilizado um *weight decay* (parâmetro que penaliza os pesos grandes) de 0.0005 para dar uma pequena penalização nos maiores pesos, e uma rotina de diminuição da taxa de aprendizado, definida na Equação 4.1, a partir da época 100. Os treinamentos foram programados para realizar 150 épocas, no entanto os pesos salvos no fim foram aqueles que alcançaram melhor resultado parcial no conjunto de validação. Cada época do treinamento foi dividida em vários *batches*, onde cada *batch* continha 64 imagens, sendo 4 imagens por pessoas e imagens de 16 pessoas distintas por *batch*.

$$lr = lr \ (0.001)^{(\text{época}-100)/50} \tag{4.1}$$

4.3 Métodos de adaptação de domínio

Para o desafio de re-identificação de pessoas, pode-se ver cada uma das bases de dados utilizadas como um domínio, pois como visto no capítulo 3 cada base apresenta parâmetros únicos e distintos das outras, como ângulo, número de câmeras, iluminação, qualidade da imagem, distância das pessoas para a câmera. Portanto, é interessante analisar os resultados de uma rede neural treinada em uma base fonte nos dados de uma base alvo e tentar melhorar esses resultados utilizando

técnicas de transferência de aprendizado.

A re-identificação de pessoas tem uma grande proximidade com o desafio de reconhecimento facial e ambas as técnicas só serão bem aceitas para aplicações reais genéricas quando se demonstrarem robustas quanto ao local de aplicação da tecnologia. Ou seja, não adianta ter a melhor tecnologia de re-identificação de pessoas se ela só funciona em seu laboratório ou na sua base de dados, pois o custo para adaptar essa tecnologia em um novo ambiente seria muito caro se toda a parte de aquisição e anotação de dados para gerar uma nova base de dados e de treinamento das redes neurais precisasse ser repetida.

Portanto, nessa seção vamos analisar vários métodos para adaptar uma rede neural treinada numa base de dados fonte de forma que ela performe melhor nos dados de uma base alvo. Analisaremos tanto métodos supervisionados quanto métodos não supervisionados para ver a diferença entre eles e entender o quanto de trabalho se têm para se fazer essa adaptação de domínio.

4.3.1 Método 1 - Transferência Direta *(Direct transfer)*

O método de transferência direta consiste em utilizar uma rede neural pré-treinada em um domínio fonte para analisar o seus resultados em um domínio alvo. Ou seja, utiliza-se as redes neurais treinadas na seção 4.2 para avaliar os seus resultados em outras bases de dados que não a utilizada para o treinamento.

Esse método assume que para os dois domínios a tarefa (geração de vetor de características) e os espaços de rótulos (pessoas) são iguais, portanto $\tau^f = \tau^a$ e $Y^f = Y^a$. Essa afirmativa é verdadeira, pois a rede neural treinada com a função de custo *triplet*, como mencionado na seção 4.2, aprende uma métrica que aproxima imagens da mesma pessoa e distancia imagens de pessoas distintas. Logo, mesmo que as bases de dados tenham pessoas distintas o objetivo do algoritmo é o mesmo (aproximar/afastar imagens de pessoas). No entanto, por se tratar de domínios diferentes, cada um com suas particularidades (iluminação, angulação, distância da câmera para as pessoas), espera-se que a maneira de gerar o vetor de características das imagens seja distinta ($P(Y|X^f) \neq P(Y|X^a)$). Por isso, não se espera um bom resultado para esse método.

Em nenhum momento de treinamento foram apresentados dados anotados da base alvo. Mesmo que o resultado esperado desse método não seja muito bom, ele é muito interessante, pois permite que seja feita uma análise de quão distantes são duas bases de dados.

4.3.2 Método 2 - *Fine Tunning*

Essa técnica é configurada quando se copia os pesos de uma rede neural treinada numa base dados fonte para iniciar o treinamento de uma rede neural em uma base de dados alvo. O nome *fine tunning* é utilizado, pois acredita-se que o aprendizado geral já ocorreu e o treinamento com essa técnica representa apenas um ajuste fino do aprendizado.

Esse mesmo método foi utilizado nos treinamentos das redes neurais para avaliação no próprio domínio de treinamento, como descrito na seção 4.2. No entanto, no caso anterior utilizamos pesos

de uma rede treinada na *ImageNet* [39], portanto a camada final de classificação da rede teve que ser retirada, pois trata-se de uma tarefa distinta. Para a situação atual não há necessidade de retirar essa última camada por se tratar da mesma tarefa e do mesmo espaço de rótulos.

O objetivo do uso dessa técnica no treinamento anterior tinha sido de acelera-lo, seguindo a premissa que as camadas superficiais aprendem representações superficiais e genéricas, logo só seria necessário aprender as representações mais específicas do desafio em questão nas camadas mais profundas. Agora, acredita-se que esse método irá adaptar essas representações mais específicas da re-identificação de pessoas de um domínio fonte para um domínio alvo, portanto o objetivo desse método é aprender as particularidades de um novo domínio.

Espera-se que esse método apresente os melhores resultados dentre todos os métodos de adaptação de domínio que serão estudados nesse trabalho. Pois, ele terá todos os dados do domínio alvo devidamente anotados e partirá de um estágio avançado de treinamento, podendo, inclusive, ultrapassar os resultados adquiridos com o método da seção 4.2. Porém, esse é o método mais caro de adaptação de domínio, pois há a necessidade de obter uma grande quantidade de dados anotados manualmente no domínio alvo para o novo treinamento.

4.3.3 Método 3 - *Pseudo* rótulos

Esse método consiste em utilizar uma rede neural, treinada em uma base de dados fonte, para classificar os dados de uma base de dados alvo. Assume-se que essa classificação é perfeita e utiliza-a para anotar os dados da base de dados alvo, o treinamento é refeito utilizando os pseudo rótulos gerados pela anotação. No entanto, a re-identificação de pessoas não é um desafio de classificação, portanto surge o questionamento: Como utilizar a saída de uma rede neural de re-identificação para anotar os dados de outra base de dados?

Uma vez que essas redes neurais foram treinadas usando a função de custo *triplet*, a saída da rede é um vetor de características que pertence a um espaço vetorial euclidiano (dado que a *triplet* foi treinada utilizando a distância euclidiana como função de comparação). Portanto, para anotar os dados de outra base a partir das saídas da rede neural, precisa-se agrupar os vetores de características de forma que cada grupamento represente uma pessoa.

O algoritmo de agrupamento escolhido foi o *k-means* [41]. Para um espaço vetorial contendo N amostras (cada amostra representa o vetor de características de uma imagem da base de dados alvo), esse algoritmo consiste em inicializar K *clusters*, de forma aleatória, dentro do espaço vetorial e agrupar n $(0 < n \leq N)$ amostras para cada *cluster*. O agrupamento é feito utilizando a política do vizinho mais próximo, ou seja, cada amostra pertencerá ao *cluster* com a menor distancia euclidiana para ela (a Equação 4.2 demonstra esse agrupamento).

$$S_{k_i} = \{x_p : \|x_p - k_i\|^2 < \|x_p - k_j\|^2 \ \ \forall j, 1 \leq j \leq K, \ j \neq i, 1 \leq p \leq N\}, \tag{4.2}$$

S_{k_i}é o conjunto de vetores de características associados ao *cluster* k_i

x_p representa cada um dos vetores de características

No entanto, um primeiro agrupamento não é ótimo, pois a posição dos K *clusters* iniciais é feita de forma aleatória. Logo, após um primeiro agrupamento, pode-se atualizar a posição dos K *clusters* para a posição do centroide do grupo associado a ele, conforme demonstra a Equação 4.3. Com essa nova posição dos K *clusters*, aplica-se novamente a Equação 4.2 para gerar um novo agrupamento e esse processo se repete por um número pré-determinado de iterações ou até atingir convergência (não mudar os agrupamentos após a atualização da posição dos *clusters*). A Figura 4.3 ilustra o processo de agrupamento do algoritmo *k-means*.

$$k_i = \frac{1}{|S_{k_i}|} \sum_{x_j \in S_{k_i}} x_j, \tag{4.3}$$

k_i é a nova posição do *cluster* i

$|S_{k_i}|$ *é a quantidade de vetores pertencentes ao cluster* k_i

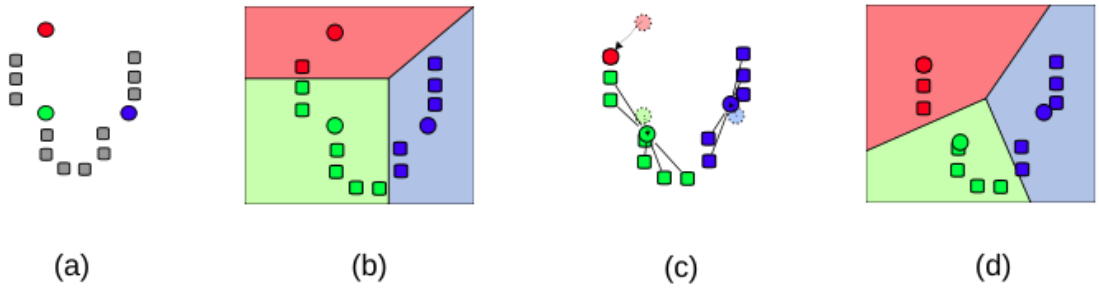

Figura 4.3: Exemplo do funcionamento do algoritmo *k-means*. (a) Inicialização inicial aleatória dos *cluster*. (b) Primeiro agrupamento, utilizando a inicialização aleatória. (c) Correção da posição dos *clusters* utilizando os centroides dos grupos. (d) Novo agrupamento a partir da correção da posição dos *clusters*. As etapas (c) e (d) são repetidas até atingir convergência ou um número máximo, pré-determinado de iterações. Reproduzida de [19].

Esse é um método de aprendizado não supervisionado, no entanto, percebeu-se que se ele fosse usado da maneira descrita a taxa de erro seria altíssima, não ajudando no resultado final. Portanto, após algumas observações foram criadas as seguintes restrições:

- **K > Qtd. ids:** O número de *clusters* K define a quantidade de grupos que serão formados, portanto esse número precisa ser maior que a quantidade de pessoas distintas na base de

Tabela 4.1: Tabela de relação entre o número de pessoas distintas nas bases de dados e o k utilizado para o algoritmo *k-means* nessa base de dados.

Base de dados alvo	**Qtd. ids**	**K**
CUHK03	1360	2000
Market1501	1501	1600
Viper	632	632
CyberQueue	6261	6500

dados alvos. Pois, para $K < Qtd.ids$ com certeza serão formados grupos com imagens de mais de uma pessoa, e para $K > Qtd.ids$ pode-se formar mais de um grupo com imagens de uma mesma pessoa que representa um problema menor para o aprendizado. A Tabela 4.1 mostra a relação da quantidade de pessoas distintas com a quantidade de grupos formados em cada base de dados estudada. Para a base de dados *Viper*, tem-se $K = Qtd.\ ids$, pois essa base apresenta apenas 1 imagem de cada pessoa por câmera.

- ***K-means* para cada câmera:** Percebeu-se que as imagens de duas pessoas distintas, mas vistas pela mesma câmera apresentam vetores de características mais próximos que imagens da mesma pessoa vista em câmeras diferentes. Portanto, os grupos formados estavam apresentando imagens apenas de pessoas na mesma câmera, então foi feito um agrupamento de *k-means* para cada câmera e depois utilizou-se o algoritmo de vizinhos mais próximos para agrupar os grupos entre câmeras.

Após esse processo de agrupamento dos vetores de características de uma base de dados alvo, espera-se que cada grupo represente uma pessoa única, portanto pode-se criar uma base de dados alvo modificada utilizando esses agrupamentos como *pseudo* rótulos. A partir dessa base de dados alvo modificada é feito um treinamento e uma avaliação dos resultados na base de dados alvo original para analisar se houve uma ganho de performance ou não.

4.3.4 Método 4 - Uso de GAN cíclica como pré-processamento de dados

Esse método foi proposto por Deng et al. [20] e consiste em treinar uma GAN [37] para tentar aproximar um domínio fonte de um domínio alvo. Assim melhorar o resultado da rede neural no domínio alvo, mesmo ela sendo treinada no domínio fonte. No entanto, esse método apresenta a dificuldade de não existirem imagens pareadas entre duas bases de dados distintas, por isso surge a ideia de usar a GAN cíclica de Zhu et al. [13].

O primeiro passo desse método consiste em treinar uma GAN cíclica de forma que ela aprenda a mapear imagens entre os domínios fonte e alvo. Nessa etapa, a GAN aprende uma função geradora G que, dada uma imagem x do domínio fonte com função de densidade de probabilidade (fdp) p_{fonte}, a transformação $G(x)$ dela vai apresentar uma fdp p_g que aproxima a fdp do domínio alvo (p_{alvo}), logo para uma imagem qualquer $x \in p_{fonte}$ a transformação $G(x)$ gera $p_g \approx p_{alvo}$. Por

propriedade, a GAN cíclica também irá aprender uma função F que para $y \in p_{alvo}$ a transformação $F(y)$ gera $p_f \approx p_{fonte}$. A Figura 4.4 ilustra o funcionamento das funções geradoras.

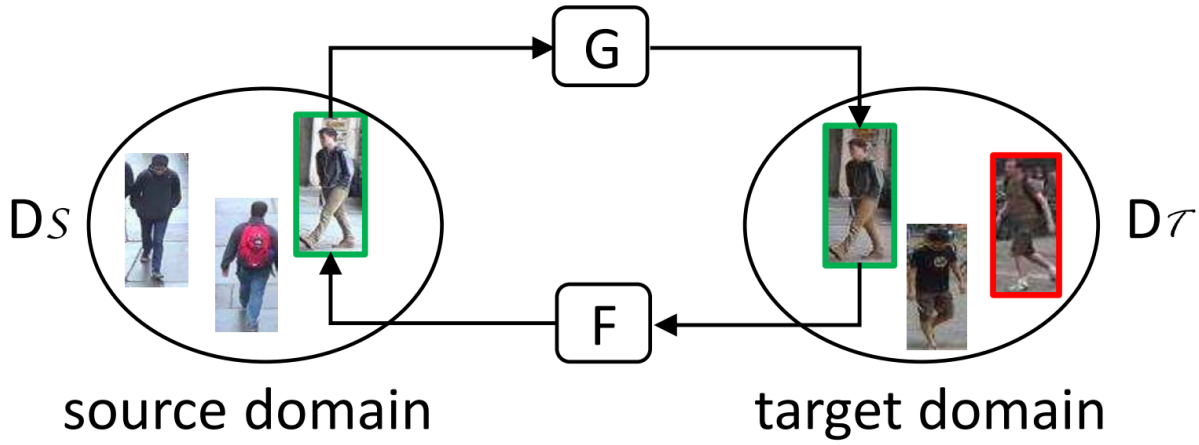

Figura 4.4: Exemplo do funcionamento das funções geradoras F e G aprendidas por uma GAN cíclica. Onde, $D_\tau = d_{alvo}$ (domínio alvo) e $D_s = D_{fonte}$ (domínio fonte). Reproduzida de [20]. ©2018 IEEE.

Uma vez que as funções geradoras aprendem a transformar as imagens de um domínio fonte de forma que elas fiquem mais próximas de um domínio alvo, utiliza-se essa técnica como pré-processamento dos dados antes do treinamento. Portanto, espera-se que esse treinamento gere uma rede neural que performe melhor no domínio alvo. A GAN cíclica é utilizada nesse caso por sua característica de facilitar o treinamento sem imagens pareadas.

O treinamento das GANs para esse método seguiu as estruturas de redes propostas por Zhu et al. [13], utilizando a Equação 2.6 com $\lambda = 10$ para 25 épocas (com taxa de aprendizado inicial de 0.0002 e *weight decay* a partir da época 15). As imagens utilizadas para esse treinamento foram de 128×256 *pixels*, para manter o mesmo padrão utilizado no treinamento das redes de re-identificação. No conjunto de treinamento foram inseridas, no máximo, 1000 images por vista de câmera, pois se mostrou mais interessante iterar sobre o conjunto de treinamento várias vezes, ao invés de iterar por mais imagens distintas.

4.4 Problemas de convergência da *triplet*

Os treinamentos iniciais foram feitos de forma supervisionada, utilizando dados de apenas uma base de dados. Eles foram feitos de forma bem pragmática, como foi determinado na seção 4.2. No entanto, para o restante dos treinamentos não foi possível definir um padrão de parâmetros utilizados, pois como os métodos de adaptação alteram as bases de dados, essas não se comportam mais tão bem quanto a original. Isso pode gerar diversas dificuldades durante o treinamento.

A maior dificuldade encontrada estava relacionada à convergência da função de custo *triplet*. Para cada *anchor* (imagem) do *batch* de treinamento era escolhida como seu exemplo positivo outra imagem do *batch* que pertencia a mesma pessoa. De forma que essa fosse a imagem que

apresentasse a maior distância do *anchor* (essa imagem representa o exemplo positivo mais difícil do *batch*). Para o exemplo negativo era escolhida uma imagem de outra pessoa, que apresentasse a menor distância da *anchor*, caracterizando-se como o exemplo negativo mais difícil do *batch*. Esse método de sempre selecionar os exemplos mais difíceis durante o treinamento se chama *batch hard*. Hermans et al. [42] realizaram um estudo sobre diversas maneiras de escolher os exemplos para acompanhar uma *anchor*. O *batch hard* apresentou os melhores resultados.

O grande problema do *batch hard* é quando o treinamento chega em uma etapa onde a distância do *anchor* para o exemplo negativo é aproximadamente igual a distância do mesmo para o exemplo positivo. Nessa etapa, se os dados forem muito difíceis, a rede pode acabar aprendendo que se ela gerar o mesmo vetor de características para todas as entradas, a função de custo sempre será reduzida para o valor da margem, independente da dificuldade apresentada pelo exemplo positivo/negativo. No entanto, chegar nesse ponto significa que todos os pesos da rede convergiram para o valor 0. Contudo, ela não aprendeu nada e o treinamento fica preso nesse ponto de não aprendizado.

Para tentar resolver esse problema foram testadas várias técnicas distintas:

- Reduzir muito a taxa de aprendizado para ver se um aprendizado mais lento poderia passar desse ponto sem problemas, no entanto só demorava mais para chegar no mesmo ponto;
- Utilizar diferentes tipos de otimizadores, com isso foram observadas diferenças na convergência durante o começo do treinamento, mas todos chegaram no mesmo ponto com mudanças apenas no tempo;
- Usar taxas de aprendizado cíclicas [9] para testar se uma rápida subida na taxa de aprendizado iria retirar a rede desse ponto de não aprendizado, mas com o passar do ciclo seguinte ela voltava para o mesmo ponto;
- Acrescentar um ruído aleatório a alguns pesos da rede de forma que eles não convergissem para zero, mas isso só fez com que o treinamento oscilasse ao redor do ponto de não aprendizado sem que a convergência ocorresse.

Após testar todos os métodos citados acima, chegou-se a conclusão de que devido as alterações feitas nos dados pelos métodos de adaptação de domínio, a tarefa tinha ficado muito complicada para ser aprendida utilizando *batch hard* e um *batch* grande (o valor padrão nos testes foi de 64 imagens por *batch*). Portanto, o método encontrado para resolver esse problema foi o método de aprendizado por etapas [1]. Esse método consiste em reduzir o *batch* no início do treinamento, pois em um universo de imagens reduzido a tarefa é mais simples de se aprender. Quando a tarefa simples é aprendida aumenta-se o *batch* dificultando um pouco a tarefa e dando continuidade no aprendizado. Esse processo se repete até alcançar a convergência no treinamento.

O algoritmo a seguir, em pseudo código, representa uma implementação básica desse método.

[1] Método inspirado em discussão encontrada em `https://github.com/VisualComputingInstitute/triplet-reid/issues/4`

Algorithm 1 Algoritmo do aprendizado por etapas.

$batch\_size = 8$
$loss\_margin = 0.5$
for $i = 0$ to num_epochs **do**
  $loss = treino(i, batch\_size)$
  if $loss < 0.8 \times loss\_margin$ **then**
    $batch\_size = batch\_size + 8$
  end if
end for

4.5 Métricas utilizadas para avaliação

Para avaliar os resultados obtidos, por cada rede neural treinada, foi utilizado sempre o mesmo procedimento, calculando algumas métricas que permitem a comparação dos resultados. As métricas escolhidas na avaliação são muito usadas na literatura para o desafio de re-identificação de pessoas, são elas: *mean Average Precision* (mAP) e *Cumulative Matching Characteristics* (CMC). Ambas as métricas são calculadas em cima do conjunto de testes.

4.5.1 *Top-k Predições*

Para analisar os resultados obtidos não é necessário que a rede neural aponte sempre o resultado correto com a maior probabilidade. Às vezes se o resultado correto está com a terceira maior probabilidade ele ainda pode ser útil. Portanto, para casos como esses, são utilizadas as *top-k* predições, onde k representa até qual posição a predição para a resposta pode ser considerada correta.

Tomando o caso de re-identificação de pessoas como exemplo, pode-se imaginar um conjunto de 300 pessoas onde deseja-se saber a qual das pessoas uma certa imagem pertence. Portanto, analisa-se essa imagem na rede neural, obtendo uma lista das pessoas mais prováveis são (onde a pessoa correta é a que está em negrito):

1. Pessoa 295 -> 21% de probabilidade;
2. Pessoa 072 -> 19.5% de probabilidade;
3. **Pessoa 198** -> 18% de probabilidade;
4. Pessoa 210 -> 15.2% de probabilidade;
5. Pessoa 004 -> 12.3% de probabilidade;
6. Restante das pessoas somadas -> 14% de probabilidade.

Para esse caso, em específico, se for adotado um $k < 3$, tem-se que o resultado da rede está incorreto. No entanto, para qualquer $k \geq 3$ esse resultado seria considerado correto.

4.5.2 *Mean Average Precision* (mAP)

A métrica mAP é dada pelo valor médio das métricas AP (*Average Precision*) por classe do conjunto de teste. No caso desse trabalho cada pessoa representa uma classe no conjunto de testes. Portanto, precisa-se entender como é feito o cálculo da AP para entender o que é a mAP.

Para entender como funciona o cálculo da AP é preciso conhecer os conceitos de precisão e *recall* (re chamada). A precisão é o percentual das suas predições que estão corretas, já a *recall* calcula o quão boa é a rede para encontrar os exemplos corretos, por exemplo, se a rede consegue encontrar 75% dos exemplos corretos em uma abordagem de *top-5* predições, a *recall* vai ter um valor de 0.75.

Para cada predição, pode-se obter quatro resultados diferentes, eles são:

- **Falso positivo (FP):** Quando a rede neural diz que aquele exemplo é verdadeiro, mas é um exemplo negativo.
- **Falso negativo (FN):** Quando a rede neural diz que aquele exemplo é negativo, mas é um exemplo verdadeiro.
- **Verdadeiro positivo (VP):** Quando a rede neural diz que aquele exemplo é verdadeiro e é, realmente, um exemplo verdadeiro.
- **Verdadeiro negativo (VN):** Quando a rede neural diz que aquele exemplo é negativo e é, realmente, um exemplo negativo.

Portanto, utilizando esses quatro resultados diferentes que podem ser obtidos em uma predição de um classificador, pode-se definir a precisão como indicado na Equação 4.4 e a re chamada como indicado na Equação 4.5.

$$\text{Precisão} = \frac{VP}{VP + FP} \tag{4.4}$$

$$\text{Recall} = \frac{VP}{VP + FN} \tag{4.5}$$

A Tabela 4.2 exemplifica como é feito o cálculo da precisão e da *recall* para um caso onde há 3 exemplos positivos que devem ser encontrados e está sendo utilizada uma abordagem de *top-5* predições. A primeira coluna da Tabela indica qual o *rank* da predição, a segunda coluna indica se aquela predição foi correta ou não, já a terceira e quarta colunas representam os valores da precisão e *recall* naquele momento.

Se for plotado um gráfico de *recall* $\times$ precisão, conceitualmente, o valor da AP pode ser encontrado calculando a área abaixo desse gráfico. No entanto, o cálculo mais utilizado da AP é feito discretizando a *recall* em intervalos de 0.1, calculando a maior precisão para cada intervalo (como apresentado na Equação 4.6) e fazendo uma média aritmética do valor da precisão em todos os intervalos (Equação 4.7).

Tabela 4.2: Tabela exemplo para ilustrar o cálculo da precisão e *recall*.

Rank	Correto?	Precisão	*recall*
1	Sim	1.0	0.33
2	Sim	1.0	0.67
3	Não	0.67	0.67
4	Não	0.5	0.67
5	Sim	0.6	1.0

$$AP_r(r) = \max(\text{precisão})\ \forall r^{'} > r \tag{4.6}$$

$$AP = \frac{1}{11}\sum_{n=0}^{1.0} AP_r(n) AP = \frac{1}{11}(AP_r(0) + AP_r(0.1) + ... + AP_r(0.9) AP_r(1.0)) \tag{4.7}$$

Por fim, a métrica mAP é calculada fazendo-se a média aritmética da AP para cada classe do conjunto de dados. Portanto, para um conjunto de dados com C classes (C pessoas distintas), a mAP pode ser calculada como na Equação 4.8.

$$MAP = \frac{1}{C}\sum_{n=0}^{C} AP_n \tag{4.8}$$

4.5.3 *Cumulative Matching Characteristics* (CMC)

A CMC é a métrica mais importante para ser analisada nesse trabalho, pois essa métrica é a mais popular entre as métricas utilizadas para avaliar os métodos de re-identificação de pessoas.

Para um conjunto de dados de teste com N imagens, cada imagem de teste será comparada com todas as outras $N-1$ imagens existentes no conjunto e ordenadas por quais são as mais prováveis de pertencerem a mesma pessoa até as menos prováveis. Caso tenha uma imagem da mesma pessoa nas *top-k* predições, considera-se que o resultado foi correto (positivo) e atribui-se 1 ao acumulador parcial (A_n) da CMC, caso contrário atribui-se 0 (Equação 4.9). Após aplicar esse procedimento para todas as N imagens do conjunto, soma-se os valores de todos os acumuladores parciais para se obter o acumulador total A_{cc} (4.10) e uma média aritmética é aplicada ao acumulador total para calcular o índice CMC (Equação 4.11).

$$A_n = \begin{cases} 1, \text{ se o resultado esperado estiver nas } \textit{top-k} \text{ predições} \\ 0, \text{ caso contrário} \end{cases} \tag{4.9}$$

$$A_{cc} = \sum_{n=1}^{N} A_n \tag{4.10}$$

$$CMC = \frac{A_{cc}}{N} \tag{4.11}$$

A Figura 4.5 mostra um exemplo da diferença entre as métricas CMC e AP. Nesta Figura, tem-se que os quadrados verdes representam as comparações positivas e os quadrados vermelhos representam as comparações negativas, e no exemplo (a) havia apenas uma comparação positiva para ser encontrada enquanto nos exemplos (b) e (c) haviam duas comparações positivas para serem encontradas em cada.

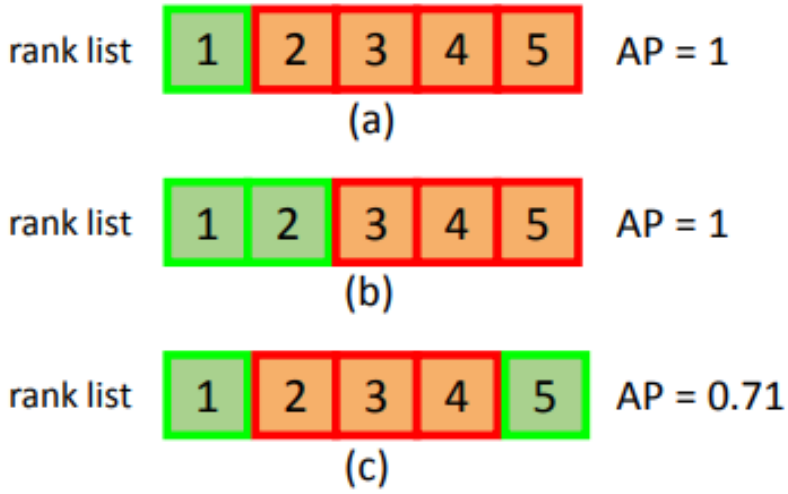

Figura 4.5: Comparação entre as métricas AP e CMC. Para esses três exemplos e utilizando uma abordagem de *top-5* predições, tem-se que a métrica CMC vale 1 para todos, enquanto a métrica AP tem resultados que variados. Reproduzida de [15]. ©2015 IEEE.

4.5.4 Tipos de comparações

O conjunto de testes contém sempre um número N de pessoas diferentes. Para cada pessoa há um número M de vistas de câmeras distintas, ou seja, cada uma das N pessoas pode ter imagens de 1 até M vistas de câmeras distintas. Esse tipo de organização pode gerar uma confusão sobre como devemos selecionar as imagens para fazer a comparação, logo alguns autores já fizeram comparações diferentes. Portanto, nesse trabalho os testes utilizaram três abordagens diferentes quanto às comparações.

Abordagem *Allshots*: Para cada imagem do conjunto de testes, faz-se uma comparação dela com um conjunto contendo uma imagem aleatória de cada pessoa de cada vista, excluindo apenas imagens da pessoa e vista em análise. No geral, é a abordagem que relata os piores resultados, pois é a que seleciona um volume maior de imagens para comparação.

Abordagem *CUHK03*: Para cada imagem do conjunto de testes, faz-se uma comparação dela com um conjunto contendo uma imagem de cada pessoa e de uma vista diferente. Por limitar bastante o universo de comparação, aumenta as chances do resultado ser positivo e, portanto, é a abordagem que relata os valores mais altos.

Abordagem *Market1501*: Cada imagem do conjunto de testes é comparada com o restante do conjunto excluindo as imagens que são da mesma pessoa e da mesma vista. Esse método é um

pouco mais difícil que o *CUHK03*, pois ele tem um universo de comparação maior, incluindo todas as vistas. No entanto, ele é mais fácil que o *Allshots*, pois inclui todas as imagens da mesma pessoa em outras vistas, o que garante que os exemplos fáceis sempre estarão presentes no universo de comparação.

A Figura 4.6 ilustra cada um dos tipos de comparação utilizados para facilitar o entendimento desses.

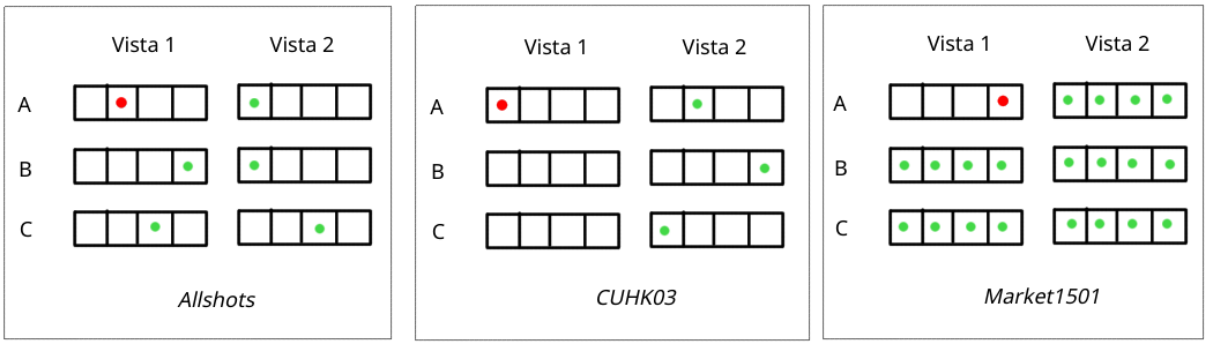

Figura 4.6: Exemplos dos tipos de comparação utilizados. Onde, A, B e C representam pessoas distintas, cada pessoa tem imagens em duas vistas e cada divisão do retângulo representa uma imagem. A divisão com um ponto vermelho representa a imagem a ser comparada e as divisões com pontos em verde representam as imagens que fazem parte do universo de comparação.

Capítulo 5

Resultados

Este capítulo apresenta os resultados obtidos por todas as técnicas utilizadas neste trabalho e faz uma análise do resultado dessas. A seção 5.1 apresenta os resultados para redes treinadas e avaliadas no mesmo domínio, e faz uma comparação dos resultados obtidos com o estado da arte. A seção 5.2 apresenta todos os resultados obtidos com os métodos de adaptação de domínio, avalia o potencial de cada um deles e faz uma comparação entre os métodos.

5.1 Avaliações no mesmo domínio

Nesta seção, apresenta-se os resultados das redes treinadas e avaliadas no mesmo domínio. Avalia-se a capacidade do método implementado de resolver o problema de re-identificação de pessoas, sem entrar no problema de adaptação de domínio. A Tabela 5.1 apresenta os resultados das avaliações nos mesmos domínios de treinamento, utilizando-se as métricas mAP para predições *top-1* e as curvas CMC para predições *top-1*, *top-5* e *top-10*.

A base de dados *CyberQueue* apresentou os piores resultados na Tabela 5.1. Um fator relevante para esse resultado é que ela tem o maior conjunto de testes (1879 pessoas distintas), por isso tem a menor probabilidade de um acerto aleatório. Outros três fatores determinantes dessa diferença são as anotações erradas, muita oclusão e imagens com um péssimo alinhamento.

Por todos os fatores apresentados, o resultado inferior na base de dados *CyberQueue* em relação as outras bases de dados já era esperado. No entanto para uma análise de *top-10* e com o método de comparação da *CUHK03* (mais próximo de uma aplicação real), tem-se uma acurácia de 67.4% mesmo considerando o universo de 1879 pessoas distintas. Portanto, se considerarmos a situação real do capítulo 3, no qual o universo é 80% menor, esse método tem bastante potencial para resolver problemas reais que apresentem um certo controle do ambiente.

Já os testes nas outras bases de dados apresentaram resultados intermediários. Ou seja, resultados superiores ao aleatório e aos primeiros resultados apresentados quando essas bases de dados foram publicadas. No entanto, ainda são resultados inferiores ao estado da arte. Uma análise mais detalhada será feita na seção 5.1.1 a seguir, onde os resultados do estado da arte serão apresentados.

Tabela 5.1: Resultados para as redes *Resnet50* treinadas em cada uma das bases de dados e avaliadas na mesma base de dados em que foram treinadas.

	Tipo de comparação (CMC)			
Base de Dados	**Allshots**	**CUHK03**	**Market1501**	**mAP**
	Top-1			
Viper	34.3%	63.4%	78.5%	61.4%
CyberQueue	18.9%	38.2%	34.5%	31.2%
CUHK03	60.8%	81.4%	79.5%	77.2%
Market1501	34.9%	64.2%	79.1%	61.8%
	Top-5			
Viper	51.4%	88.4%	91.5%	61.4
CyberQueue	31.2%	60.4%	50.6%	31.2
CUHK03	74.5%	97.2%	88.6%	77.2
Market1501	51.9%	88.5%	91.3%	61.8
	Top-10			
Viper	60.0%	93.5%	94.5%	61.4
CyberQueue	38.1%	67.4%	57.7%	31.2
CUHK03	81.3%	98.3%	93.1%	77.2
Market1501	60.5%	93.7%	94.7%	61.8

5.1.1 Estado da arte

Para entender melhor a qualidade dos resultados obtidos, estes serão comparados com os resultados do estado da arte no desafio de re-identificação de pessoas. A Tabela 5.2 indica os resultados do estado da arte na base de dados pública *CUHK03* e a Tabela 5.3 indica os resultados do estado da arte na base de dados pública *Market 1501*, ambas Tabelas foram retiradas de [6]. Já a Tabela 5.4 indica os resultados do estado da arte na base de dados pública *Viper*, essas informações foram retiradas de [22].

O método que atinge o estado da arte na base de dados *CUHK03* foi proposto por Liu et al. [21]. A rede *HP-net*, proposta por eles, pode ser vista na Figura 5.1. Essa rede consiste em dois módulos: um módulo principal (*Main net*) e um módulo de mapeamento de atenção (*Attentive Feature Net*). O módulo principal é responsável pelas convoluções principais que irão gerar um vetor de características da imagens. Já o módulo de mapeamento de atenção irá gerar vários mapas de atenção, de diversos níveis semânticos diferentes (profundidade de convoluções distintas), para depois indicar quais partes dos vetores de características devem ganhar mais ou menos pesos.

Para a base de dados *Market1501*, o método que atinge o estado da arte é a rede *MLFN* que foi utilizada como exemplo para a explicação de arquiteturas modularizadas no capítulo 2. Essa rede foi proposta por Chang et al. [6] e consiste em uma rede neural altamente modularizada. Ela utiliza informações de várias camadas superficiais para gerar a assinatura digital (vetor de características) da pessoa. A Figura 2.7 mostra a arquitetura de rede neural proposta por Chang

Tabela 5.2: Resultados do estado da arte na base de dados pública *CUHK03*. Encontra-se em itálico o resultado encontrado neste trabalho e em negrito o melhor resultado de cada coluna. A métrica utilizada para calcular esses resultados foi a curva CMC *top-1* com comparação *CUHK03*. Reproduzida de [6]. ©2018 IEEE.

CUHK03	Top-1
DGD [32]	75.3%
Spindle [22]	88.5%
HP-net [21]	**91.8%**
LSRO [43]	84.6%
SVDNet [44]	81.8%
DPFL [45]	82.0%
MLFN [6]	89.2%
Nosso	*81.4%*

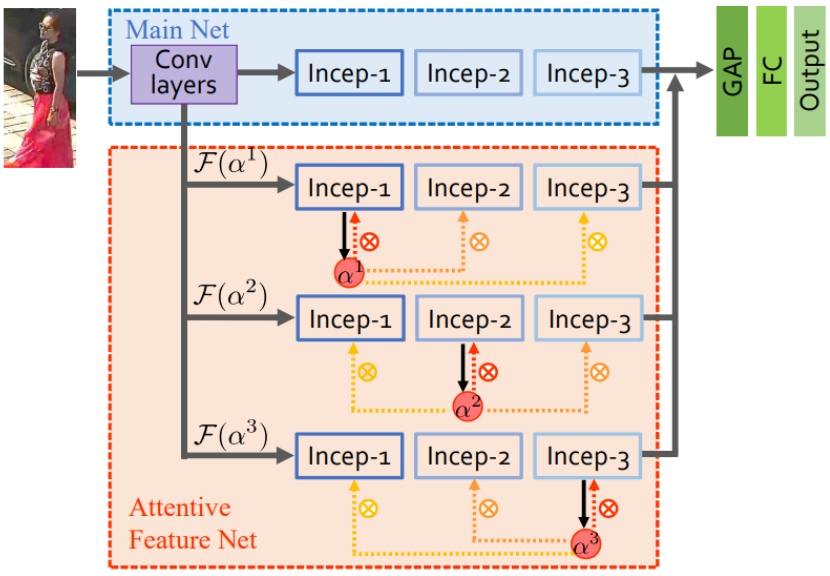

Figura 5.1: Rede *HP-net* responsável pelo melhor resultado já publicado na base de dados *CUHK03*. Reproduzida de [21]. ©2017 IEEE.

Tabela 5.3: Resultados do estado da arte na base de dados pública *Market 1501*. Encontra-se em itálico os resultados encontrados nesse trabalho e em negrito o melhor resultado de cada coluna. A métrica utilizada para calcular esses resultados foram a curva CMC *top-1* com comparação *Market1501* e a mAP. Reproduzida de [6]. ©2018 IEEE.

Market 1501	Top-1	mAP
Context [46]	86.8%	66.7%
JLML [47]	89.7%	74.5%
LSRO [43]	88.4%	76.1%
SSM [48]	88.2%	76.2%
DML [49]	91.7%	77.1%
DPFL [45]	92.2%	80.4%
MLFN [6]	**92.3%**	**82.4%**
Nosso	*79.1%*	*61.8%*

et al.

A rede *Spindle* proposta por Zhao [22] apresentou os melhores resultados para a base de dados *Viper*. Mesmo não sendo um trabalho muito recente (de 2017), essa rede continua sendo o estado da arte na base de dados *Viper*. Por ser uma base de dados com poucas imagens, a *Viper* tem sido pouco utilizada nos estudos mais recentes.

A rede *Spindle* consiste de dois módulos principais: uma rede de extração de features (FEN) e uma rede de fusão de features (FCN). O módulo de extração de features é responsável por detectar diferentes partes do corpo da pessoa (braços, pernas, tronco, cabeça e quadril) e gerar um vetor de características para cada parte do corpo. O módulo de fusão de features irá receber esses vetores de características de cada parte do corpo e determinar como esses vetores serão agrupados para formar o vetor de características da pessoa. A Figura 5.2 ilustra a rede *Spindle*.

Os resultados apresentados foram inferiores ao estado da arte. Para a base de dados *CUHK03* o resultado foi 10.4% pior do que o estado da arte. Para a base de dados *Market 1501* essa diferença foi de 13.2%. Para a base de dados *Viper* o resultado foi 19.5% inferior ao estado da arte.

Ao observar as redes que atingiram o estado da arte em cada base de dados estudada, ficou claro que os resultados ótimos de re-identificação de pessoas são obtidos por redes que utilizam arquiteturas modularizadas. Isso faz sentido ao pensar que a imagem de uma pessoa contém informações importantes em vários níveis semânticos. Por exemplo, inicialmente é importante detectar as cores da imagem, depois a posição da pessoa na imagem, depois tamanho da pessoa, depois formato de roupas, depois se é um homem ou mulher, etc. As arquiteturas modularizadas facilitam o aprendizado desses níveis semânticos. Nas arquiteturas residuais, como a *ResNet50*, os módulos residuais ajudam a aprender e a observar esses variados níveis semânticos, mas eles sozinhos não bastam para atingir a performance da arquitetura modularizada.

Tabela 5.4: Resultados do estado da arte na base de dados pública *Viper*. Encontra-se em itálico os resultados encontrados nesse trabalho e em negrito o melhor resultado de cada coluna. A métrica utilizada para calcular esses resultados foi as curvas CMC com comparação *Allshots*. Reproduzida de [22]. ©2017 IEEE.

Viper	Top-1	Top-5	Top-10
TMA [50]	48.2%	-	87.7%
NFST [51]	51.2%	82.1%	90.5%
SCSP [52]	53.5%	**82.6%**	**91.5%**
SSDAL+XQDA [53]	43.5%	**82.6%**	81.5%
LOMO+XQDA [54]	40.0%	-	80.5%
MLAPG [55]	40.7%	82.3%	-
GOG+XQDA [56]	49.7%	79.7%	88.7%
TCP [57]	47.8%	74.7%	84.8%
Spindle [22]	**53.8%**	74.1%	83.2%
Nosso	*34.3%*	*51.4%*	*60.0%*

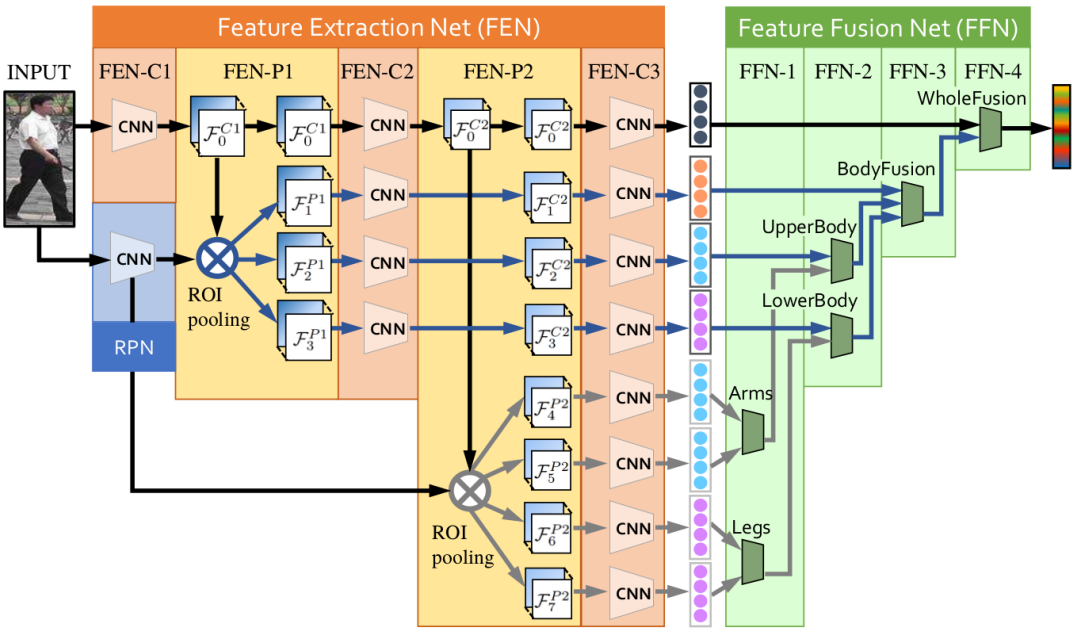

Figura 5.2: Rede *Spindle* responsável pelo melhor resultado já publicado na base de dados *Viper*. Reproduzida de [22]. ©2017 IEEE.

5.2 Avaliações em domínios distintos

5.2.1 Transferência direta

Todas as redes neurais desenvolvidas foram avaliadas em todas as bases de dados utilizadas nesse trabalho. Essa seção tem o intuito de mostrar a avaliação das redes treinadas, sendo aplicadas em bases de dados distintas daquelas onde elas foram treinadas, sem nenhum processamento extra. A Tabela 5.5 apresenta os resultados das curvas CMC para o método de transferência direta, utilizando o método de comparação *Allshots*.

Tabela 5.5: Resultados ao aplicar o método de Transferência direta nas bases de dados estudadas.

		Domínio Alvo			
Domínio fonte	**CMC**	**CUHK03**	**Market1501**	**Viper**	**CyberQueue**
CUHK03	**Top - 1**	X	3.7%	10.1%	0.1%
	Top - 5	X	8.1%	22.5%	0.4%
	Top - 10	X	11.3%	29.0%	0.5%
Market1501	**Top - 1**	5.2%	X	12.5%	0.2%
	Top - 5	10.2%	X	25.0%	0.5%
	Top - 10	14.3%	X	33.1%	0.8%
Viper	**Top - 1**	4.3%	34.3%	X	0.3%
	Top - 5	9.2%	51.4%	X	0.6%
	Top - 10	14.1%	60.0%	X	0.9%
CyberQueue	**Top - 1**	0.3%	0.5%	1.1%	X
	Top - 5	1.1%	1.4%	4.0%	X
	Top - 10	1.6%	2.1%	6.6%	X

Os resultados da apresentados na Tabela 5.5 mostram a complexidade de se treinar redes neurais que sejam robustas à variações de domínios. No entanto, para melhor analisar esses resultados, precisa-se saber que as probabilidades de um acerto aleatório nas bases de dados utilizados são de 0.027% para a base de dados *CyberQueue*, 0.500% para a *CUHK03*, 0.067% para a *Market 1501* e 0.158% para a *Viper*, utilizando a abordagem *allshots*.

Os testes de na base de dados *CyberQueue* foram os que obtiveram os piores resultados. Esse resultado era esperado, pois como analisado no capítulo 3, essa base de dados foi criada utilizando algoritmos de detecção de pessoas que resultou em várias imagens onde só parte da pessoa é vista. Também foram utilizadas técnicas de reconhecimento facial, portanto há imagens que contém mais de uma pessoa, logo a quantidade de oclusões nessa base de dados é muito grande.

No entanto, mesmo os resultados do método de transferência direta sendo muito aquém do esperado e muito inferior aos resultados das avaliações no mesmo domínio, se quando comparados com a probabilidade de acerto aleatório, esses resultados mostraram que aprendizados robustos ocorreram. Pois, os resultados apresentados são pelo menos 5 vezes melhores do que a chance de

acerto aleatório, considerando o pior caso.

Como descrito na seção 4.3.1, o método de transferência direta é interessante para analisar a distância entre duas bases de dados. Pois, uma vez que a tarefa e o espaço de rótulos são os mesmos, tem-se que os resultados distintos são causados pela diferença entre as funções de densidade de probabilidade $\big(P(Y|X^f) \neq P(Y|X^a)\big)$. Logo, pode-se dizer que as bases de dados *Viper* e *Market1501* são as mais próximas entre si, e que a base de dados *CyberQueue* é realmente muito distante de todas as outras.

5.2.2 *Fine Tunning*

A Tabela 5.6 apresenta os resultados das curvas CMC obtidos com o método de *fine tunning*. Todos os resultados apresentados na Tabela 5.6 utilizam a abordagem de comparação *Allshots*, apresentada no capítulo 4.

Tabela 5.6: Resultados ao aplicar o método de *fine tunning* nas bases de dados estudadas.

		Domínio Alvo			
Domínio fonte	**CMC**	**CUHK03**	**Market1501**	**Viper**	**CyberQueue**
CUHK03	**Top - 1**	X	31.8%	24.8%	9.8%
	Top - 5	X	48.7%	53.0%	18.7%
	Top - 10	X	57.3%	66.5%	24.8%
Market1501	**Top - 1**	53.7%	X	22.3%	13.2%
	Top - 5	67.4%	X	50.6%	23.7%
	Top - 10	74.8%	X	63.1%	30.3%
Viper	**Top - 1**	56.7%	35.1%	X	15.5%
	Top - 5	71.8%	52.0%	X	26.2%
	Top - 10	78.6%	60.5%	X	33.0%
CyberQueue	**Top - 1**	46.1%	23.2%	12.7%	X
	Top - 5	55.2%	40.7%	32.8%	X
	Top - 10	64.7%	48.1%	46.0%	X

Os resultados da Tabela 5.6 não demonstram um ganho em relação aos resultados apresentados na Tabela 5.1, exceto pela rede treinada na base de dados *Viper* e feita transferência de aprendizado para a base de dados *Market1501*, que apresentou um ganho de 0.2%. No entanto, nota-se que quase todos os resultados alcançados com o método de *fine tunning* são muito próximos daqueles alcançados com o treinamento em apenas um domínio.

Esse resultado aponta que, provavelmente, as camadas superficiais da rede treinada na *ImageNet*, que foram utilizadas para treinar as redes na seção 5.1, conseguem inicializar o treinamento melhor do que as camadas das redes treinadas em outras bases do mesmo desafio. Isso demonstra que, por mais que visualmente as bases de re-identificação de pessoas sejam parecidas, cada uma delas contém nuances que à dificultam bastante. No entanto, por se tratar da mesma tarefa, os treinamentos dessa seção convergiram para o resultado apresentado na Tabela 5.6 aproximadamente

três vezes mais rápido que os treinamentos realizados na seção 5.1.

5.2.3 *Pseudo* rótulos

Um ponto importante de ser mencionado antes de analisar os resultados gerados pelo método de treinamento que utiliza *pseudo* rótulos é a qualidade do agrupamento feito. Pois, o fato de tratar-se de bases de dados muito grandes, com milhares de imagens, aumenta muito o universo de aplicação das redes. Essas que mostraram resultados pífios para o método de transferência direta. Além disso, tem-se o fato de o método de agrupamento *k-means* ser heurístico e ter uma queda de rendimento diretamente relacionada ao tamanho do K. A Tabela 5.7 foi montada analisando quantos dos grupos criados realmente apresentam, em sua maioria, imagens de apenas uma pessoa.

Tabela 5.7: Resultados do agrupamento da imagens de uma base de dados utilizando os vetores de características extraídos a partir de uma outra base de dados e o método *k-means* para agrupamento. Foi utilizada a métrica CMC top-1 para gerar esses resultados.

	Domínio Alvo			
Domínio fonte	**CUHK03**	**Market1501**	**Viper**	**CyberQueue**
CUHK03	X	1.50%	10.13%	0.22%
Market1501	2.75%	X	10.28%	0.23%
Viper	3.40%	4.50%	X	0.23%
CyberQueue	0.25%	0.12%	1.42%	X

Os resultados da Tabela 5.7 mostram o quão ruim foram os agrupamentos formados. Há uma discrepância leve nos dados para a base *Viper* como alvo, isso ocorre porque essa base só tem uma 2 câmeras e 1 imagem de pessoa por câmera, facilitando muito o agrupamento. O resultado de 4.50% da base *Viper* para a base *Market1501* é um reflexo claro da proximidade entre elas que ficou clara na Tabela 5.5. Pode-se ver, também, os péssimos resultados sempre que a base de dados *CyberQueue* está envolvida, mostrando mais uma vez o quanto ela é diferente das outras bases de dados utilizadas.

A Tabela 5.8 apresenta os resultados das curvas CMC quando usado o método de *pseudo* rótulos para treinamento das redes neurais. Todos os resultados apresentados na Tabela 5.8 utilizam a abordagem de comparação *Allshots*.

Ao comparar os resultados da Tabela 5.8 com os da Tabela 5.5, nota-se um incremento na acurácia de alguns testes. No entanto, fica a dúvida sobre como a rede conseguiu melhorar os resultados se os agrupamentos foram tão ruins?

Há dois fatores chaves para a melhoria nos resultados mesmo com esses valores tão ruins de agrupamento, seguem:

- **Aprendizado com imagens do domínio alvo:** Mesmo com o alto índice de erro do agrupamento, esse treinamento é feito utilizando imagens do domínio alvo. Portanto, a rede

Tabela 5.8: Resultados ao aplicar o método de uso de *pseudo* rótulos nas bases de dados estudadas.

		Domínio Alvo			
Domínio fonte	**CMC**	**CUHK03**	**Market1501**	**Viper**	**CyberQueue**
CUHK03	**Top - 1**	X	5.7%	22.6%	0.1%
	Top - 5	X	11.9%	28.5%	0.3%
	Top - 10	X	16.4%	35.0%	0.4%
Market1501	**Top - 1**	6.5%	X	11.4%	0.2%
	Top - 5	11.8%	X	22.5%	0.5%
	Top - 10	16.9%	X	33.3%	0.8%
Viper	**Top - 1**	2.0%	14.6%	X	0.2%
	Top - 5	5.5%	26.5%	X	0.4%
	Top - 10	8.9%	33.9%	X	0.7%
CyberQueue	**Top - 1**	0.7%	0.3%	3.0%	X
	Top - 5	1.6%	1.0%	4.0%	X
	Top - 10	2.6%	1.6%	6.6%	X

tem capacidade de aprender parâmetros específicos do domínio alvo que não dependem de um agrupamento correto. Por exemplo, aprender a identificar e segmentar as pessoas nas imagens da base de dados alvo;

- **Sentido semântico nas imagens do grupamento:** Os agrupamentos apresentam um alto índice de erro, pois a análise foi feita apenas verificando se as imagens pertenciam a mesma pessoa. No entanto, não foi levada em consideração a semântica entre as imagens de um agrupamento errôneo. Ao olhar a Figura 5.3, nota-se que mesmo o agrupamento apresentando imagens de pessoas distintas, há um sentido semântico no grupo, uma vez que todas as pessoas do grupo apresentam roupas da mesma cor, por exemplo.

Contudo, mesmo com esses fatores da rede ver imagens do domínio alvo e elas apresentarem um sentido semântico dentro de um mesmo grupo, o fato do agrupamento não ser tão bom fica claro quando analisamos o ganho de performance comparado com o método de transferência direta. Para algumas bases de dados mais próximas o resultado até piorou. Provavelmente, as redes aprenderam características erradas a partir dos grupos errados. Por fim, é interessante observar que a relação dos resultados da Tabela 5.7 e 5.8, onde os melhores resultados de uma Tabela refletem diretamente nos melhores resultados da outra.

5.2.4 Uso de GAN cíclica como pré-processamento

O uso da GAN cíclica como pré-processamento dos dados de treinamento é um método de adaptação de domínio não supervisionada. Esse método foi utilizado para tentar aproximar as imagens de uma base de dados fonte às imagens de uma base de dados alvo, num momento prévio

Figura 5.3: Exemplo de um grupo que contém imagens de mais de uma pessoa, porém com um sentido semântico nas cores das roupas das pessoas. Esse grupo foi formado utilizando *Viper* como domínio fonte e *Market1501* como domínio alvo.

ao treinamento. A Tabela 5.9 mostra os resultados obtidos quando utilizado esse método de adaptação de domínio. Todos os resultados apresentados nessa Tabela correspondem ao método *Allshots* de comparação.

É interessante notar que esse é o primeiro método não supervisionado que apresenta, no geral, um ganho de acurácia na base de dados *CyberQueue*, que é a mais distante das demais. E, também, melhora os resultados de redes treinadas na base de dados *CyberQueue* quando aplicada em outras bases. Logo, acredita-se que a transformação realizada pelas GANs realmente conseguiu alcançar o seu objetivo de aproximar as imagens de dois domínios distintos.

No entanto, há uma leve perda de performance quando as as duas bases de dados já são próximas. Como, por exemplo, na rede treinada nas base *Viper* e avaliada na base *Market1501*, e vice-versa. Provavelmente, isso ocorre pois ao tentar aproximar as imagens das duas bases de dados, a GAN insere um ruído na imagem original e esse ruído pode ter distanciado as duas bases.

A Figura 5.4 mostra as transformações da GAN cíclica entre as bases *CUHK03* e *Market1501*. É interessante notar que a GAN tenta preservar as informações da pessoa, mas altera bastante o fundo da imagem para aproximar os domínios (simula o piso granulado da *CUHK03* nas imagens da *Market1501*, e simula a presença de grama no fundo das imagens da *CUHK03* para ficarem mais parecidas com o parque onde foram obtidas as imagens da *Market1501*). No entanto, as pessoas nas imagens também sofrem algumas alterações. Isso pode estar relacionado a tentar aproximar condições de iluminação, por exemplo, entre as imagens. Porém, esse tipo de alteração pode ser

Tabela 5.9: Resultados ao aplicar o método de uso de GAN cíclica como pré-processamento nas bases de dados estudadas.

		Domínio Alvo			
Domínio fonte	**CMC**	**CUHK03**	**Market1501**	**Viper**	**CyberQueue**
CUHK03	**Top - 1**	X	6.2%	11.6%	0.7%
	Top - 5	X	12.8%	25.5%	1.8%
	Top - 10	X	17.4%	34.7%	2.8%
Market1501	**Top - 1**	13.9%	X	9.8%	0.3%
	Top - 5	23.1%	X	26.9%	0.8%
	Top - 10	29.8%	X	36.4%	1.3%
Viper	**Top - 1**	9.5%	21.8%	X	0.4%
	Top - 5	18.1%	36.9%	X	0.9%
	Top - 10	24.6%	45.1%	X	1.3%
CyberQueue	**Top - 1**	1.3%	1.1%	1.7%	X
	Top - 5	3.0%	2.7%	7.3%	X
	Top - 10	4.2%	3.8%	9.8%	X

maléfico para o aprendizado da rede se acabar mudando a morfologia da imagem da pessoa.

5.2.5 Comparação das técnicas de adaptação de domínio

Para facilitar a comparação entre os métodos utilizados, a Tabela 5.10 compila os resultados apresentados nas Tabelas 5.6, 5.5, 5.8 e 5.9. Para montar a Tabela de comparação foram retirados todos os resultados de acurácia *top-1* das demais Tabelas. A correspondência entre os métodos e o número atribuído a ele na Tabela encontra-se na legenda a seguir:

- **Método 1:** Transferência direta;
- **Método 2:** *Fine tunning*;
- **Método 3:** *Pseudo* rótulos;
- **Método 4:** GAN cíclica como pré-processamento.

O método de *fine tunning* apresenta, em todos os casos, os melhores resultados para a adaptação de domínio. Isso já era esperado por esse método ser supervisionado e ter todas as informações do domínio alvo disponíveis. No entanto, a não superação desse método quando comparado com o treinamento em apenas um domínio é um pouco desapontante, pois esse método inicia o treinamento de um ponto avançado e tem capacidade para obter os melhores resultados possíveis.

Ao analisar os métodos de aprendizado não supervisionado, pode-se ver um pior resultado do método 2. Isso está dentro do esperado, pois esse é um método que não tenta aprender características do domínio alvo, ele simplesmente é avaliado nesse. O método 3 apresenta um resultado

Figura 5.4: Exemplo da transformação feita pela GAN cíclica utilizando imagens das bases de dados *CUHK03* e *Market1501*.

Tabela 5.10: Comparação entre todos os métodos de adaptação de domínio.

		Target domain			
Source domain	**Método**	**CUHK03**	**Market1501**	**Viper**	**CyberQueue**
CUHK03	1	X	3.7%	10.1%	0.1%
	2	X	31.8%	24.8%	9.8%
	3	X	5.7%	22.6%	0.1%
	4	X	6.2%	11.6%	0.7%
Market1501	1	5.2%	X	12.5%	0.2%
	2	53.7%	X	22.3%	13.2%
	3	6.5%	X	11.4%	0.2%
	4	13.9%	X	9.8%	0.3%
Viper	1	4.3%	34.3%	X	0.3%
	2	56.7%	35.1%	X	15.5%
	3	2.0%	14.6%	X	0.2%
	4	9.5%	21.8%	X	0.4%
CyberQueue	1	0.3%	0.5%	1.1%	X
	2	46.1%	23.2%	12.7%	X
	3	0.7%	0.3%	3.0%	X
	4	1.3%	1.1%	1.7%	X

intermediário, pois ele já tem um contato com as imagens de um domínio alvo mesmo que essas contenham vários erros (como mostrado na Tabela 5.7). Por fim, o método 4 apresenta o melhor dos resultados, pois ele consegue combinar a anotação correta dos dados (usa a anotação do domínio fonte) e uma imagem que é aproximada das imagens do domínio fonte.

A base de dados *Viper* apresenta um comportamento um pouco diferente das outras quanto aos métodos que melhor performam. Isso, provavelmente, está relacionado com o fato de ela ser uma base muito pequena para o treinamento de redes neurais profundas. O fato de apresentar poucas imagens e apenas 1 imagem por pessoa por câmera, também explica o motivo dessa base apresentar resultados tão superiores ao utilizar o método 3, pois o agrupamento para essa base era muito mais simples que para as demais.

É interessante notar que esses métodos conseguem incrementar os resultados na base de dados *CyberQueue*. Pois, essa base é muito distinta das demais. Ela apresenta resultados gerais inferiores. No entanto, pode-se ver que os métodos de adaptação de domínio conseguem aumentar a performance da rede em até 100% em casos que a base de dados *CyberQueue* é utilizada, seja como domínio fonte ou alvo.

Capítulo 6

Conclusões

Foi apresentado neste trabalho uma breve revisão da literatura de re-identificação de pessoas, alguns resultados e técnicas utilizadas no atual estado da arte. Portanto, o objetivo proposto de familiarização com a literatura foi cumprido.

O problema proposto, pela *CyberLabs*, de cronometrar o tempo de uma fila utilizando a re-identificação de pessoas na entrada e na saída foi solucionado com sucesso. Mesmo os resultados da rede treinada na base de dados *CyberQueue* apontando apenas 18.9% para as *top-1* predições, esse resultado não atrapalhou na aplicação prática desse algoritmo. Pois, quando é feita uma comparação de teste na base de dados, essa comparação é feita com quase 2000 outras pessoas distintas, no entanto num caso real esse universo de comparação é muito reduzido, uma vez que a capacidade da fila é de menos de 200 pessoas. Nesse universo menor de comparação e utilizando restrições temporais para auxiliar, foi possível cronometrar o tempo das pessoas que passavam pela fila com uma taxa de acerto acima de 90%.

Os resultados de treinamento e avaliação num mesmo domínio alcançados nesse trabalho ficaram um pouco aquém do estado da arte. Isso ocorreu, provavelmente, pelo não uso das redes de arquiteturas modularizadas que se mostraram eficientes para o desafio de re-identificação de pessoas. Esses resultados foram base para as técnicas de adaptação de domínio. Logo, é interessante num trabalho futuro que essa *baseline* inicial seja melhorada, para poder estudar todo o potencial das técnicas de adaptação de domínio.

A técnica de adaptação de domínio utilizando *pseudo* rótulos apresentou algumas falhas, mas mesmo assim obteve-se resultados interessantes. Portanto, poderia ser interessante estudar variações dessa técnica. Por exemplo, usar outras técnicas de agrupamento ou aplicar essa técnica várias vezes (de forma recursiva). Além do mais, essa técnica é muito interessante, pois ela reduz muito o custo do treinamento em um domínio alvo, uma vez que não há a necessidade de anota-lo. Porém, se fosse feita uma anotação parcial no domínio alvo e iniciasse a técnica de *pseudo* rótulos já garantindo um percentual de acerto dado por essa anotação parcial, seriam alcançados resultados ainda melhores e sem perder a principal vantagem que é a diminuição no custo de anotação da base de dados.

No geral, os melhores resultados das técnicas de adaptação de domínio foram apresentados pela

técnica que utilizou as GANs como pré-processamento. No entanto, viu-se que essa técnica pode, as vezes, alterar a morfologia da imagem da pessoa e atrapalhar o aprendizado. Deng et al. [20] propuseram um incremento na função de custo da GAN cíclica para penalizar os casos onde ela altera a morfologia da imagem das pessoas, logo um estudo desse tipo de técnica pode ser benéfico para o problema em questão.

Por fim, nenhuma das técnicas de adaptação de domínio apresentadas é excludente. Logo, seria interessante utilizar combinações dessas técnicas e avaliar os resultados.

6.1 Perspectivas Futuras

Ainda há muito espaço para melhorias e estudos no tema deste trabalho, como foi visto na seção anterior. Logo, alguns tópicos principais são propostos para um futuro trabalho que tenha em vista melhorar os resultados apresentados:

- Melhorar a *baseline* do trabalho explorando o potencial das redes modularizadas para esse problema;
- Utilizar mais métodos de aumento de dados, como o método de apagar parte da imagem ou utilizar GANs que aproximem imagens de duas câmeras da mesma base de dados;
- Estudar como melhorar a GAN cíclica para preservar mais a imagem da pessoa, como foi feito por Deng et al. [20] incrementando o erro da GAN cíclica;
- Fazer experimentos com combinações de técnicas não excludentes e recursivas.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] CLOSED Circuit Television System (CCTV). `http://www.ertembilisim.com/kapali-devre-televizyon-sistemi-cctv.html`. Online; version of 24 March 2014.

[2] LI, W.; ZHAO, R.; XIAO, T.; WANG, X. Deepreid: Deep filter pairing neural network for person re-identification. In: *Proc of the IEEE Conf on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2014. p. 152–159. ISSN 1063-6919.

[3] ZHENG, L.; YANG, Y.; HAUPTMANN, A. G. Person re-identification: Past, present and future. *CoRR*, abs/1610.02984, 2016. Disponível em: <https://arxiv.org/abs/1610.02984>.

[4] FARENZENA, M.; BAZZANI, L.; PERINA, A.; MURINO, V.; CRISTANI, M. Person re-identification by symmetry-driven accumulation of local features. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2010. p. 2360–2367.

[5] HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016.

[6] CHANG, X.; HOSPEDALES, T. M.; XIANG, T. Multi-level factorisation net for person re-identification. In: *Proc of the IEEE Conf on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2018.

[7] SCHROFF, F.; KALENICHENKO, D.; PHILBIN, J. Facenet: A unified embedding for face recognition and clustering. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2015.

[8] KULKARNI, C. *Learning Rate Tuning and Optimizing*. `https://medium.com/@ck2886/learning-rate-tuning-and-optimizing-d03e042d0500`. Online; version of 19 February 2018.

[9] Smith, L. N. Cyclical learning rates for training neural networks. In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. [S.l.: s.n.], 2017. p. 464–472.

[10] Pan, S. J.; Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, v. 22, n. 10, p. 1345–1359, Oct 2010. ISSN 1041-4347.

[11] GENERATIVE Adversarial Networks - Explained. `https://towardsdatascience.com/generative-adversarial-networks-explained-34472718707a`. Online; version of 10 May 2018.

[12] ISOLA, P.; ZHU, J.-Y.; ZHOU, T.; EFROS, A. A. Image-to-image translation with conditional adversarial networks. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2017.

[13] ZHU, J.-Y.; PARK, T.; ISOLA, P.; EFROS, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: *The IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2017.

[14] GRAY, D.; BRENNAN, S.; TAO, H. Evaluating appearance models for recognition, reacquisition, and tracking. In: *In IEEE International Workshop on Performance Evaluation for Tracking and Surveillance, Rio de Janeiro*. [S.l.: s.n.], 2007.

[15] ZHENG, L.; SHEN, L.; TIAN, L.; WANG, S.; WANG, J.; TIAN, Q. Scalable person re-identification: A benchmark. In: *IEEE International Conference on Computer Vision*. [S.l.: s.n.], 2015.

[16] BRITS know how to queue. www.express.co.uk. Online; version of 3 May 2017.

[17] REN, S.; HE, K.; GIRSHICK, R.; SUN, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In: CORTES, C.; LAWRENCE, N. D.; LEE, D. D.; SUGIYAMA, M.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems (NIPS) 28*. Curran Associates, Inc., 2015. p. 91–99. Disponível em: <http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf>.

[18] HUANG, G. B.; MATTAR, M.; BERG, T.; LEARNED-MILLER, E. Labeled Faces in the Wild: A Database forStudying Face Recognition in Unconstrained Environments. In: Erik Learned-Miller and Andras Ferencz and Frédéric Jurie. *Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition*. Marseille, France, 2008. Disponível em: <https://hal.inria.fr/inria-00321923>.

[19] K-MEANS clustering. `https://en.wikipedia.org/wiki/K-means_clustering`. Online; version of 27 June 2019.

[20] DENG, W.; ZHENG, L.; YE, Q.; KANG, G.; YANG, Y.; JIAO, J. Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2018.

[21] LIU, X.; ZHAO, H.; TIAN, M.; SHENG, L.; SHAO, J.; YI, S.; YAN, J.; WANG, X. Hydraplus-net: Attentive deep features for pedestrian analysis. In: *The IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2017.

[22] ZHAO, H.; TIAN, M.; SUN, S.; SHAO, J.; YAN, J.; YI, S.; WANG, X.; TANG, X. Spindle net: Person re-identification with human body region guided feature decomposition and fusion. In: *Proc 30th IEEE Conf on Computer Vision and Pattern Recognition (CVPR), Honolulu, Hawaii, July 21-26*. [S.l.: s.n.], 2017.

[23] Wei Niu; Jiao Long; Dan Han; Yuan-Fang Wang. Human activity detection and recognition for video surveillance. In: *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*. [S.l.: s.n.]. v. 1, p. 719–722 Vol.1.

[24] TAIGMAN, Y.; YANG, M.; RANZATO, M.; WOLF, L. Deepface: Closing the gap to human-level performance in face verification. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2014.

[25] HUANG, T.; RUSSELL, S. Object identification in a bayesian context. In: *Proceedings of the Fifteenth International Joint Conference on Artifical Intelligence (ISCAI) - Volume 2*. San Francisco, CA, USA: [s.n.], 1997. (IJCAI'97), p. 1276–1282.

[26] ZAJDEL, W.; ZIVKOVIC, Z.; KROSE, B. J. A. Keeping track of humans: Have i seen this person before? In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. [S.l.: s.n.], 2005. p. 2081–2086. ISSN 1050-4729.

[27] GHEISSARI, N.; SEBASTIAN, T. B.; HARTLEY, R. Person reidentification using spatio-temporal appearance. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) - Volume 2*. Washington, DC, USA: IEEE Computer Society, 2006. p. 1528–1535.

[28] YI, D.; LEI, Z.; LIAO, S.; LI, S. Z. Deep metric learning for person re-identification. In: *Proceedings of the 2014 22Nd International Conference on Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2014. (ICPR), p. 34–39.

[29] XU, J.; ZHAO, R.; ZHU, F.; WANG, H.; OUYANG, W. Attention-aware compositional network for person re-identification. In: *Proc of the IEEE Conf on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2018.

[30] ZHONG, Z.; ZHENG, L.; ZHENG, Z.; LI, S.; YANG, Y. Camera style adaptation for person re-identification. In: *Proc of the IEEE Conf on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2018.

[31] ZHONG, Z.; ZHENG, L.; KANG, G.; LI, S.; YANG, Y. Random erasing data augmentation. *CoRR*, abs/1708.04896, 2017. Disponível em: <http://arxiv.org/abs/1708.04896>.

[32] XIAO, T.; LI, H.; OUYANG, W.; WANG, X. Learning deep feature representations with domain guided dropout for person re-identification. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016.

[33] WANG, J.; SONG, Y.; LEUNG, T.; ROSENBERG, C.; WANG, J.; PHILBIN, J.; CHEN, B.; WU, Y. Learning fine-grained image similarity with deep ranking. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2014.

[34] DAUPHIN, Y. N.; VRIES, H. de; CHUNG, J.; BENGIO, Y. Rmsprop and equilibrated adaptive learning rates for non-convex optimization. *CoRR*, abs/1502.04390, 2015. Disponível em: <http://arxiv.org/abs/1502.04390>.

[35] SMITH, L. N. A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay. *CoRR*, abs/1803.09820, 2018. Disponível em: <https://arxiv.org/abs/1803.09820>.

[36] CSURKA, G. A comprehensive survey on domain adaptation for visual applications. In: CSURKA, G. (Ed.). *Domain Adaptation in Computer Vision Applications*. Cham: Springer International Publishing, 2017. p. 1–35. ISBN 978-3-319-58347-1.

[37] GOODFELLOW, I.; POUGET-ABADIE, J.; MIRZA, M.; XU, B.; WARDE-FARLEY, D.; OZAIR, S.; COURVILLE, A.; BENGIO, Y. Generative adversarial nets. In: GHAHRAMANI, Z.; WELLING, M.; CORTES, C.; LAWRENCE, N. D.; WEINBERGER, K. Q. (Ed.). *Advances in Neural Information Processing Systems (NIPS) 27*. Curran Associates, Inc., 2014. p. 2672–2680. Disponível em: <http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf>.

[38] DLIB C++ Library. `http://dlib.net/`. Online; version of 10 March 2019.

[39] DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. ImageNet: A Large-Scale Hierarchical Image Database. In: *CVPR09*. [S.l.: s.n.].

[40] DONAHUE, J.; JIA, Y.; VINYALS, O.; HOFFMAN, J.; ZHANG, N.; TZENG, E.; DARRELL, T. Decaf: A deep convolutional activation feature for generic visual recognition. In: XING, E. P.; JEBARA, T. (Ed.). *Proceedings of the 31st International Conference on Machine Learning*. Bejing, China: PMLR, 2014. Disponível em: <http://proceedings.mlr.press/v32/donahue14.html>.

[41] HARTIGAN, J. A.; WONG, M. A. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 1979. Disponível em: <http://www.jstor.org/stable/2346830>.

[42] HERMANS, A.; BEYER, L.; LEIBE, B. In defense of the triplet loss for person re-identification. *CoRR*, abs/1703.07737, 2017. Disponível em: <http://arxiv.org/abs/1703.07737>.

[43] Zheng, Z.; Zheng, L.; Yang, Y. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.]. p. 3774–3782. ISSN 2380-7504.

[44] Sun, Y.; Zheng, L.; Deng, W.; Wang, S. Svdnet for pedestrian retrieval. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.]. p. 3820–3828. ISSN 2380-7504.

[45] Chen, Y.; Zhu, X.; Gong, S. Person re-identification by deep learning multi-scale representations. In: *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*. [S.l.: s.n.]. p. 2590–2600. ISSN 2473-9944.

[46] Li, D.; Chen, X.; Zhang, Z.; Huang, K. Learning deep context-aware features over body and latent parts for person re-identification. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.]. p. 7398–7407. ISSN 1063-6919.

[47] LI, W.; ZHU, X.; GONG, S. Person re-identification by deep joint learning of multi-loss classification. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI*. [S.l.]: AAAI Press, 2017. p. 2194–2200. ISBN 978-0-9992411-0-3.

[48] Bai, S.; Bai, X.; Tian, Q. Scalable person re-identification on supervised smoothed manifold. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.]. p. 3356–3365. ISSN 1063-6919.

[49] Zhang, Y.; Xiang, T.; Hospedales, T. M.; Lu, H. Deep mutual learning. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.]. p. 4320–4328. ISSN 2575-7075.

[50] MARTINEL, N.; DAS, A.; MICHELONI, C.; ROY-CHOWDHURY, A. K. Temporal model adaptation for person re-identification. In: LEIBE, B.; MATAS, J.; SEBE, N.; WELLING, M. (Ed.). *Computer Vision – ECCV 2016*. Cham: Springer International Publishing. p. 858–877.

[51] ZHANG, L.; XIANG, T.; GONG, S. Learning a discriminative null space for person re-identification. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016.

[52] CHEN, D.; YUAN, Z.; CHEN, B.; ZHENG, N. Similarity learning with spatial constraints for person re-identification. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016.

[53] SU, C.; ZHANG, S.; XING, J.; GAO, W.; TIAN, Q. Deep attributes driven multi-camera person re-identification. In: LEIBE, B.; MATAS, J.; SEBE, N.; WELLING, M. (Ed.). *Computer Vision – ECCV 2016*. Cham: Springer International Publishing. p. 475–491.

[54] LIAO, S.; HU, Y.; ZHU, X.; LI, S. Z. Person re-identification by local maximal occurrence representation and metric learning. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2015.

[55] Liao, S.; Li, S. Z. Efficient psd constrained asymmetric metric learning for person re-identification. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.]. p. 3685–3693. ISSN 2380-7504.

[56] MATSUKAWA, T.; OKABE, T.; SUZUKI, E.; SATO, Y. Hierarchical gaussian descriptor for person re-identification. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016.

[57] CHENG, D.; GONG, Y.; ZHOU, S.; WANG, J.; ZHENG, N. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016.