UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

ESCOLA DE ENGENHARIA

DEPARTAMENTO DE ENGENHARIA QUÍMICA

ENG07053 - TRABALHO DE DIPLOMAÇÃO EM ENGENHARIA QUÍMICA

Análise da Similaridade de imagens com redes neurais

Autor: Angelo Carmignani

Orientador: Prof. Dr. Jorge Otávio Trierweiler

Porto Alegre, maio de 2021

Autor: Angelo Carmignani

Análise da Similaridade de imagens com redes neurais

Trabalho de Conclusão de Curso apresentado à COMGRAD/ENQ da Universidade Federal do Rio Grande do Sul como parte dos requisitos para a obtenção do título de Bacharel em Engenharia Química

Orientador: Prof. Dr. Jorge Otávio Trierweiler

Banca Examinadora:

Prof. Dr. Marcelo Farenzena, Dequi/UFRGS

Doutorando Rafael Martello, Dequi/UFRGS

Porto Alegre

2021

AGRADECIMENTOS

Agradeço à Universidade Federal do Rio Grande do Sul pela oportunidade da minha formação em Engenharia Química e realização desse trabalho.

Aos professores e funcionários da universidade, pelos ensinamentos, auxílio e orientações, principalmente ao meu orientador Prof. Dr. Jorge Otávio Trierweiler.

Ao meu pai, Attilio, que sempre me incentivou a curiosidade e o estudo. Trouxe ao meu carácter valores imprescindíveis de ética e moral.

A minha mãe, Márcia, com todo amor e paciência que teve comigo no último ano e todo apoio que me deu, mesmo à distância.

A Heloísa, pelo apoio ao longo de toda graduação, pela convivência que tivemos juntos que ajudou muito para que esse trabalho fosse realizado de maneira mais tranquila.

Aos meus colegas de trabalho que instigaram minha curiosidade por ciência de dados e por todos ensinamentos trocados.

Muito obrigado.

RESUMO

Aprendizado por métricas de distância é um campo de extrema importância devido ao grande volume dados que é possível extrair atualmente. Para uma maior interpretação e utilização de menos recursos computacionais é necessário reduzir a dimensionalidade desses dados. A utilização de métricas de distância juntamente com redes neurais apresentam uma ferramenta poderosa para agrupar imagens que podem ser utilizadas em diversos campos, como na indústria, medicina e segurança. O objetivo em questão do trabalho é avaliar diferentes técnicas para similaridade de imagem baseadas na distância, principalmente voltada para funções de otimização, e como é possível melhorá-las com diferentes topologias de redes neurais e novos parâmetros adicionais. A principal ferramenta utilizada para estudo foi o *tensorflow*, com *API Keras,* a qual já apresenta modelos pré-treinados utilizados para estudo como *ResNet50* e *InceptionV3.* Foi realizada uma comparação entre essas duas estruturas como espinha dorsal, como também a adição de novas camadas (normalização, dropout e dimensionalidade de *embedding*). A principal base utilizada foi a *Standford Online Products*, em que dada a estrutura da rede neural com os parâmetros definidos foi avaliado principalmente a métrica de revogação das *k* imagens mais próximas como também a velocidade de convergência da rede. A principal perda utilizada foi a *Proxy NCA,* com um ajuste para maximizar a probabilidade de certa imagem pertencer a *Proxy,* como também a adição de um parâmetro ajustável conhecido como temperatura que apresentou melhoras na revogação. O modelo final apresentou resultados muito próximos em revogação frente aos atuais estados de arte.

Palavras-chave: *Redes neurais*, *Aprendizado de métricas de distância*, *similaridade de imagens*

ABSTRACT

Distance metric learning is an essential field nowadays due to the large volume of data extracted today. For a better interpretation and use of fewer computational resources, it is necessary to reduce the dimensionality of this data. The use of distance metrics and neural networks presents a powerful tool for clustering images that can be used in different fields, such as in industry, medicine, and security. The present work aims to evaluate different techniques for image similarity based on distance, mainly focused on loss functions and how it is possible to improve them with different neural networks layout and new additional parameters. The main tool for the study is the tensorflow, with API Keras, which already presents pre-trained models such as ResNet50 and InceptionV3. A comparison was made between these two structures as a backbone and the addition of new layers (normalization, dropout, and embedding dimensionality). The main dataset studied is the Standford Online Products, in which, given the structure of the neural network with the defined parameters, it was evaluated mainly the recall metric of the nearest k images as well as the network convergence speed. The main loss used was the Proxy NCA, with an adjustment to maximize the probability of a specific image belonging to Proxy and the addition of an adjustable parameter known as temperature that showed improvements in the recall. The final model showed very similar results in recall compared to the current state of the art.

Keywords: *Neural Network*, *Distance metric learning*, *Image comparison*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 Introdução

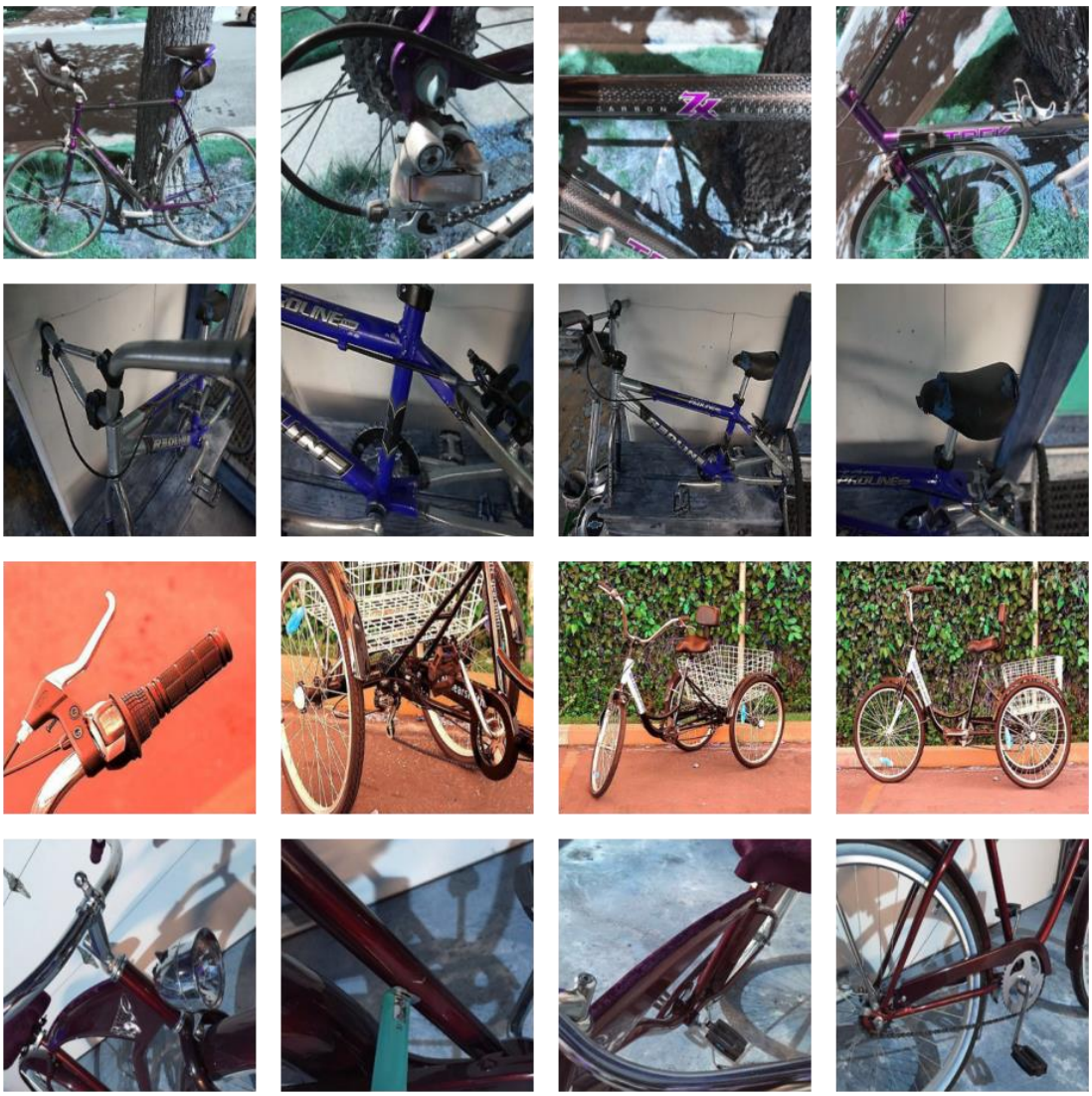
Redes convolucionais têm sido amplamente utilizadas para classificação de imagem em larga escala, principalmente devido ao acesso a bases públicas ricas em informação diversificadas, tal como *ImageNet* (Jia Deng et al., 2009) que apresenta cerca de 50 milhões de imagens totalmente classificadas. A partir dessas bases foi possível treinar redes neurais mais complexas de maneira eficaz e, hoje, o reconhecimento de imagens com redes convolucionais já é um assunto bastante dominado.

No entanto, esse domínio é limitado a base utilizada para treinar. Para o reconhecimento de imagens funcionar de maneira efetiva, as imagens que vão ser reconhecidas precisam, necessariamente, terem sido utilizadas na base de treino. Dessa forma, se existir uma base extremamente heterogênea ou que apresente atualizações constantes, tal metodologia começa a apresentar uma menor eficiência.

Logo, uma metodologia chamada aprendizado na primeira tentativa (*zero-shooting learning*), tem crescido nos últimos tempos. Ao invés de treinar e reconhecer somente imagens que estavam na base do treino, essa metodologia tem por objetivo treinar uma rede de tal forma que consiga distinguir imagens jamais vistas na mesma base, e a principal maneira de realizar isso é treinar uma rede que aprenda a semântica das imagens e as compare por similaridade. Suponha o seguinte exemplo da Figura 1, em que é apresentado diferentes tipos de bicicletas, se a rede for treinada para identificar bicicleta, a rede neural irá retornar que todas as imagens têm alta probabilidade de ser uma bicicleta, mas não irá conseguir distingui-las. Pode ser realizado um treino específico para diferentes tipos de bicicleta, mas também seria limitado a constante atualização do modelo como também a necessidade de inúmeras classificações para os diferentes tipos de bicicletas.

Para tornar esse reconhecimento mais efetivo, ao invés de simplesmente determinar se é uma bicicleta de tal tipo ou não com o retorno de uma probabilidade, é possível comparar uma imagem frente a todas apresentadas em um banco de dados, por exemplo, e retornar as imagens mais similares. Para que isso seja possível é necessário que a rede neural aprenda a semântica das imagens e seja indiferente ao tratamento da imagem (rotacionada, recortada, contraste alterado, saturação).

Figura 1: Exemplo de imagens de bicicletas com semânticas e tratamentos diferentes



Fonte: Do autor

A principal estrutura da rede, diferentemente do utilizado para classificação (camada final com único neurônio e ativação *softmax*) é a presença de uma camada de incorporação (neurônios totalmente conectados) com uma ativação linear e dimensionalidade ajustável que possibilita uma redução de dimensionalidade das imagens. Com essa redução é possível comparar a posição de cada imagem no espaço de maneira mais eficaz e, dessa forma, otimizar essa posição com uma função objetivo, a qual faz com que as imagens mais próximas serão mais semelhantes e as mais afastadas menos. Essa definição de imagens mais próximas ou distantes podem ser feitas através de uma distância euclidiana ou similaridade de cossenos, por exemplo.

Algumas técnicas, como incorporação linear local (Hou et al., 2009), foram propostas para redução de dimensionalidade em imagens, porém tais técnicas apresentam problemas como a dificuldade de bons resultados em imagens novas que não foram utilizadas no treino ou a existência de uma métrica de distância precisa para calcular as posições no espaço. Esse segundo ponto é principalmente limitado devido ao uso de combinações lineares que somente é efetivo para imagens padrões (sem rotação, corte ou alteração de luminosidade) que apresentam uma grande similaridade.

Dado essa problemática, novas técnicas foram propostas chamadas de redução de dimensionalidade com mapeamento invariante que utilizam exaustivamente a não-linearidade que modelos de redes neurais proporcionam. Desta forma, o presente estudo tem por objetivo avaliar essas técnicas da literatura e aplicá-las, com uma estrutura de rede neural adequada, a fim de criar um modelo com resultado próximo dos principais estados de arte de hoje em dia.

2 Revisão Bibliográfica

Para avaliar a melhor estratégia para se obter um modelo efetivo em que seja possível definir as imagens mais similares dentro de um grupo, existem dois principais fatores que foram revisados na literatura com destaque para o tipo de modelo utilizado como espinha dorsal e as funções objetivas de otimização, que são discutidos a seguir.

2.1 Tipo de modelo e revisão de redes neurais

Para que seja possível obter melhores resultados ao treinar uma rede neural é possível utilizar uma rede pré-treinada como espinha dorsal do modelo e realizar um ajuste fino nela a fim de extrair informações provenientes de uma base mais rica. Dessa forma, foi avaliado para o estudo presente três estruturas principais de redes neurais, de tal forma que fosse possível utilizá-las sem requerer um grande esforço computacional.

2.1.1 VGG

A primeira topologia de rede apresentada é a *VGG* (Simonyan e Zisserman, 2015) que se difere de topologias anteriores como *AlexNet* (Krizhevsky et al., 2017) e *LeNet 5* (Lecun et al., 1998), as quais apresentavam camadas convolucionais com filtros de grande campo de receptividade, como 7x7. Além disso, para esses casos geralmente era alternado entre uma camada convolucional e de agrupamento.

Porém, Simonyan e Zisserman, 2015, observeram que tais camadas poderiam ser substituídas por camadas empilhadas de filtros menores, como por exemplo substituir a camada com filtro de 7x7 por três empilhadas de filtro 3x3. Além disso, como não há perdas no campo de receptividade, a operação matemática requer menos recursos computacionais. Supondo que todas as camadas apresentem a mesma dimensão de entrada e saída nos canais, C, o total de pesos a serem calculados, sem considerar o peso do viés, para um filtro 7x7 seria $7x7xCxC=49xC^2$ e, para três camadas 3x3, seria de $3x3x3xCxC=27xC^2$, apresentando uma eficiência 45% melhor no total de pesos calculados. Por fim, o uso das camadas empilhadas torna o modelo mais discriminativo.

Levando essa vantagem das camadas empilhadas, a topologia do *VGG* consiste em multiplos blocos de camadas convolucionais empilhadas, seguidas de uma camada de

agrupamento máximo, tal qual a Figura 2 apresenta, e no final dela três camadas densas totalmente conectadas. Conforme a rede vai se desenvolvendo, o número de filtros das camadas convolucionais vão crescendo 64/128/256/512, além disso, a última camada apresenta duas camadas de 4096 neurônios totalmente conectados. Devido a isso, a rede apresenta uma grande quantidade de pesos a serem calculados, o que leva a ser computacionalmente exigente e com a necessidade de um uso de memória muito elevada.

Figura 2: Topologia da rede *VGG 16,* conv seriam as camadas convolucionais seguidas pelo filtro aplicado e a quantidade de filtros aplicados, e densa seriam as camada totalmente conectadas seguida do tamanho da saída.

VGG 16
conv 3x3, 64
conv 3x3, 64
polimento máximo
conv 3x3, 128
conv 3x3, 128
polimento máximo
conv 3x3, 256
conv 3x3, 256
conv 3x3, 256
polimento máximo
conv 3x3, 512
conv 3x3, 512
conv 3x3, 512
polimento máximo
conv 3x3, 512
conv 3x3, 512
conv 3x3, 512
polimento máximo
densa-4096
densa-4096
densa-1000
Softmax

Fonte: Adaptado de (Okatani, 2015)

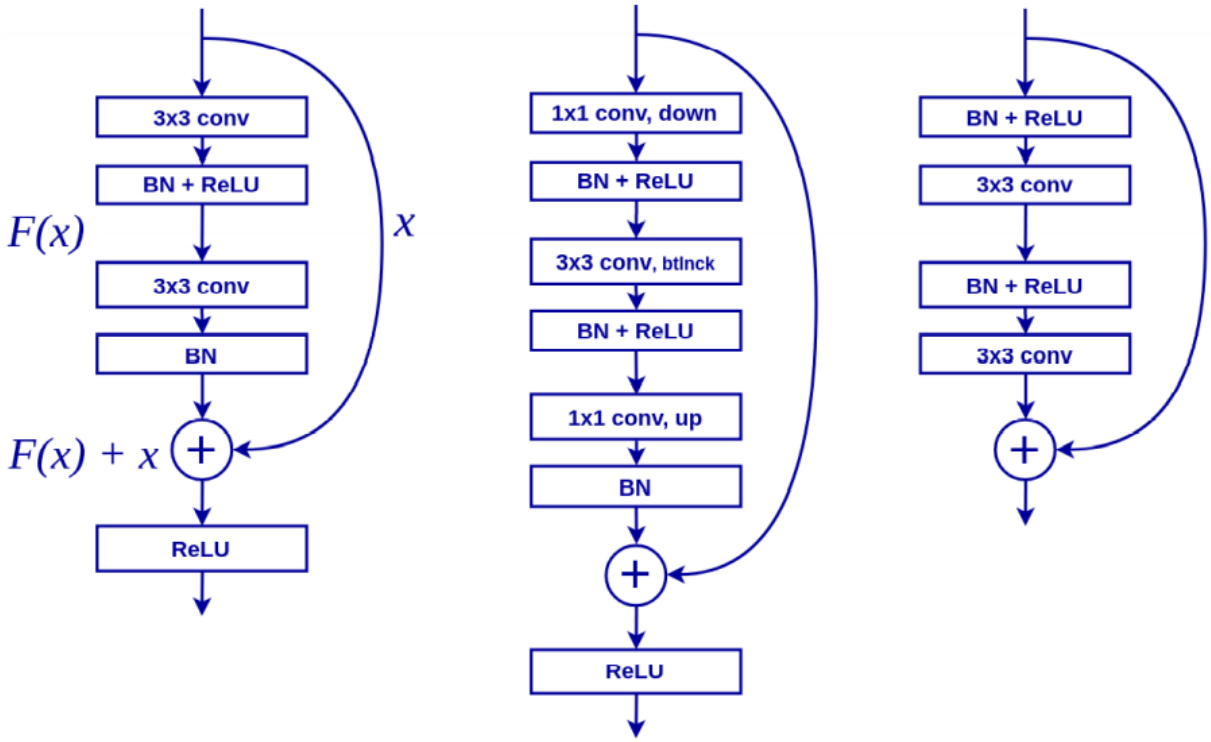
2.1.2 Redes residuais

Redes residuais (He et al., 2016) apresentam pela primeira vez um estrutura diferente, em que as camadas não são construídas apenas de maneira sequencial. Essa é a primeira estrutura em que foi possível treinar uma rede com uma profundidade de mais de 100

camadas, devido a uma melhoria de recursos, tal como novas funções de ativação, melhor inicialização de pesos e normalização em lotes (Ioffe e Szegedy, 2015).

Apesar das melhorias que possibilitaram um treino em uma maior profundidade de camadas, os autores perceberam que redes com camadas sequenciais mais profundas apresentavam um maior erro tanto no treino quanto na validação do que uma rede com menor profundidade. Dessa forma, para poder maximizar o uso dessas novas melhorias, os autores propuseram a estrutura apresentada na Figura 3. A ideia principal consiste na criação de blocos residuais, os quais são compostos de dois ou três blocos convolucionais, seguidos por uma normalização de lote e geralmente uma função de ativação ReLU (Xu et al., 2015). Cada estrutura apresenta dois caminhos, o da esquerda que seria o usual bloco convolucional e o da direita, que se pode chamar de um atalho de conexão, isto é, sem a passagem por uma camada de convolução ou agrupamento. No final de ambos os blocos, eles são somados de maneira matricial, visto que os blocos residuais apresentam um preenchimento para sempre manter as mesmas dimensões da camada de entrada. Com isso, é possível propagar o que foi aprendido pelo bloco mais a informação vinda do cenário original. Essa estrutura, por sua vez, apresenta a vantagem de que a rede pode decidir, de acordo com os pesos de cada camada, se pode pular alguns blocos convolucionais na sua estrutura e, devido a isso, é possível empilhar inúmeros blocos convolucionais na rede.

Figura 3: Estrutura básica para redes residuais. BN é abreviação para normalização em lote e ReLU é a função de ativação após a camada convolucional.



Fonte: (Okatani, 2015)

Por fim, essa estrutura apresentou uma evolução nos blocos residuais. Originalmente os blocos eram formados tal qual o primeiro é apresentando, porém se as camadas começarem a apresentar uma alta dimensionalidade é necessário mais recurso computacional. Uma solução seria utilizar a estrutura do segundo bloco, que requer menos esforço computacional. A ideia proposta seria de reduzir a profundidade (RGB para imagens, por exemplo), através de uma convolução com filtro 1x1 (atua somente em cada ponto e não em uma região). Após isso, é utilizado uma convolução 3x3, chamada de gargalo, seguida de novamente uma convolução 1x1, que retorna à dimensionalidade da camada de entrada. Além dessas duas, a ideia mais recente proposta pelo autor é de antes de passar pelas camadas de convolução ser utilizado uma normalização em lote juntamente com uma ativação ReLU. Dessa forma, a camada que passa pelo atalho não fica sujeita a nenhuma função, a não ser a soma com o bloco residual. Com isso é possível que o atalho se prolongue ao longo de toda rede e não apenas entre blocos, visto que não é apresentado nenhuma operação após a soma matricial do atalho e do bloco residual.

Com essa topologia de rede, os autores conseguiram ganhar, em 2015, todas as categorias do desafio do *ImageNet*, conseguindo um resultado em acurácia na base de 78% com a topologia mais complexa, superando em 7 pontos percentuais a estrutura *VGG* (71%).

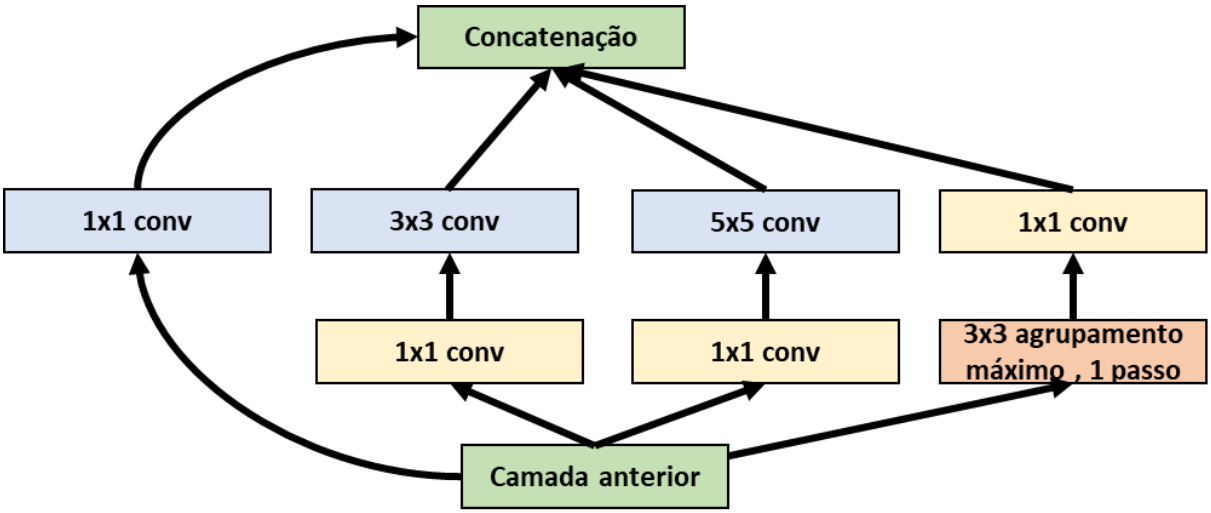
2.1.3 Redes Inception

A primeira estrutura de rede *Inception* foi proposta por Szegedy et al., em 2014 , a qual apresentou melhorias ao longo dos anos. A principal motivação da topologia da rede é baseada no fato que objetos em uma imagem podem ser apresentados com um tamanho diferente. Por exemplo, um objeto distante em uma imagem acaba ocupando uma região pequena, porém se o objeto está próximo ele acaba ocupando quase toda região da imagem.

Diante dessa motivação, o autor propôs a criação de blocos de princípio (*inception blocks)*. O bloco é composto por uma camada de entrada, a qual é separada em diferentes caminhos e esses caminhos são concatenados no fim. Cada caminho apresenta uma operação diferente, podendo ser uma camada de convolução com tamanho de filtro diferente ou camadas de agrupamento. Devido a isso, é possível aplicar diferentes filtros na imagem e, com isso, obter diferentes campos de recepção.

A primeira estrutura de *inception,* mais conhecida como GoogleLeNet (Szegedy et al., 2015), era composta por nove blocos com a composição apresentada na Figura 4. Um bloco com uma convolução 1x1, com redução de profundidade que funcionaria como representação da camada anterior, outro com uma convolução 1x1 seguida por uma 3x3, o terceiro bloco com uma convolução 1x1 seguida por uma 5x5 e, por fim, um último bloco com um agrupamento 3x3 máximo, com deslocamento de uma célula. Além disso é importante destacar que todos os blocos utilizam de preenchimento de tal maneira que a saída final seja igual a entrada da camada anterior, porém com profundidades diferentes.

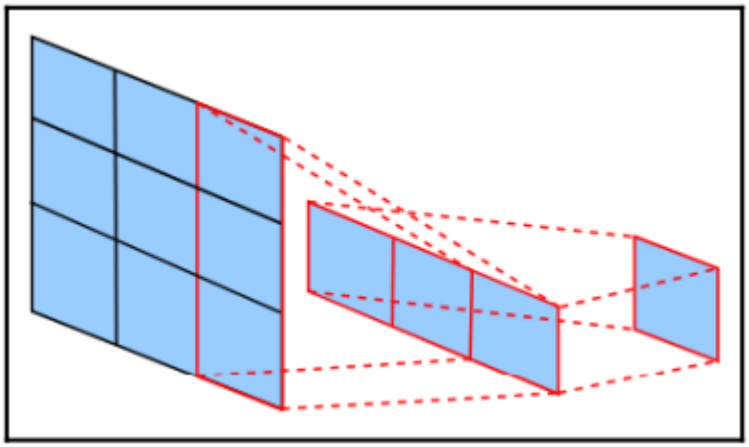
Figura 4: Estrutura dos blocos de princípio na rede GoogleLeNet.



Fonte: Adaptado de Okatani, 2015

Mais além, os autores desenvolveram outras topologias para os blocos de princípio chamadas de *Inception v2* e *Inception v3,* sendo a principal diferença entre elas o uso de normalização em lotes na última. Para ambas redes é substituída a convolução 5x5 por duas 3x3 empilhadas para melhor eficiência computacional. Outra melhoria proposta pelo autor seria de substituir as convoluções 3x3 por duas 1x3 seguidas de uma 3x1. No final, a operação obtém o mesmo resultado, tal qual a Figura 5 apresenta, ainda apresentando uma melhoria de eficiência pois o tamanho do filtro é reduzido de uma área de 9 u.m para 6 u.m.

Figura 5: Equivalência entre convolução 3x3 por duas convoluções assimétricas 1x3,3x1



Fonte: (Okatani, 2015)

2.2 Funções objetivas

2.2.1 Contrastive loss

Essa função objetiva foi desenvolvida por Hadsell et al., 2006 o qual tinha por objetivo obter uma métrica que diminuísse a distância entre as camadas de incorporação (*embeddings)* das imagens pertencentes a mesma classe e aumentasse a distância entre as não pertencentes. Para tanto a equação (2.2) foi desenvolvida, podendo ser interpretada da seguinte forma: quando duas imagens $(\overrightarrow{X_1}, \overrightarrow{X_2})$, pertencerem a mesma classe $(Y = 1)$, será calculada a distância euclidiana entre os *embeddings* $(G_w(\overrightarrow{X_1}), G_w(\overrightarrow{X_2}))$. Caso não pertençam a mesma classe, há uma margem de distância m , a qual a função objetivo só será considerado se a distância entre os *embeddings* estiver dentro do raio m.

$$D_W(\overrightarrow{X_1}, \overrightarrow{X_2}) = \left\| G_w(\overrightarrow{X_1}) - G_w(\overrightarrow{X_2}) \right\|_2 \tag{2.1}$$

$$L(W, Y, \overrightarrow{X_1}, \overrightarrow{X_2}) = \frac{(Y)}{2}(D_W)^2 + \frac{(1-Y)}{2}\{\max(0, m - D_W)\}^2 \tag{2.2}$$

Quando a equação (2.2) é minimizada, a distância entre imagens de mesma classe é minimizada e para diferentes classes, caso estejam dentro do raio m, a distância intra-classes é maximizada. Dessa forma é possível agrupar as imagens mais semelhantes dentro de um raio m, mantendo distante as que não são pertencentes a mesma classe.

No entanto, um ponto importante dessa métrica é que ao se calcular as distâncias, elas são comparadas sempre em pares, não levando em consideração os pontos vizinhos. Dessa forma, por exemplo, se três pontos (P1, P2, P3) são colineares e pertencentes a diferentes classes. Quando se maximizar a distância entre (P1, P2), a distância entre (P2, P3) será minimizada mesmo não pertencendo a mesma classe.

Com isso, tal métrica tem uma limitação de semântica de definir o que é semelhante ou não. Por exemplo, caso um conjunto de imagens pertencentes a mesma classe seja comparada a imagens aleatórias e diferentes, a semelhança é mais fácil de ser encontrada. No entanto, para um conjunto de imagens similares, mas definidas com diferentes classes, tal como identificação facial, essa métrica obtém resultados abaixo do esperado. Para suprir a falha dessa métrica, uma nova função foi desenvolvida conhecida como *triplet loss.*

2.2.2 Triplet loss

Para levar o conceito de similaridade mais afundo, não apenas comparando imagem em pares, Hoffer e Ailon (2015), criaram uma função de perda que leva em consideração trios de imagens, sendo uma chamada de âncora (P), outra imagem pertencente a mesma classe (P^+) e uma imagem de outra classe (P^-).

Para que seja possível comparar trios de imagem foi desenvolvido uma topologia específica de rede neural. Para cada imagem no trio é utilizado uma rede neural, e que compartilham o mesmo parâmetro tal qual a Figura 6 apresenta. Com essa estrutura é possível obter a função de perda levando em consideração as três imagens no espaço, e logo, minimizar a distância entre as imagens da mesma classe e ao mesmo tempo aumentar a distância da âncora frente as imagens de diferentes classes. Dada a equação (2.3) **Erro! Fonte de referência não encontrada.** , onde $G_w(X)$ é o *embedding*, é possível minimizar essa questão, visto que a $Perda(d_+, d_-) \to 0$,se e somente se $\frac{\|G_w(P) - G_w(P^+)\|}{\|G_w(P) - G_w(P^-)\|} \to 0$. Além disso essa função, juntamente com a rede de trios, permite que a rede atualize seus pesos sempre levando em consideração as três imagens simultaneamente e, portanto, corrigindo o problema apresentado pela *Constrastive Loss.*

$$Perda(d_+, d_-) = \|d_+, d_- - 1\|_2^2 = const \cdot d_+^2 \tag{2.3}$$

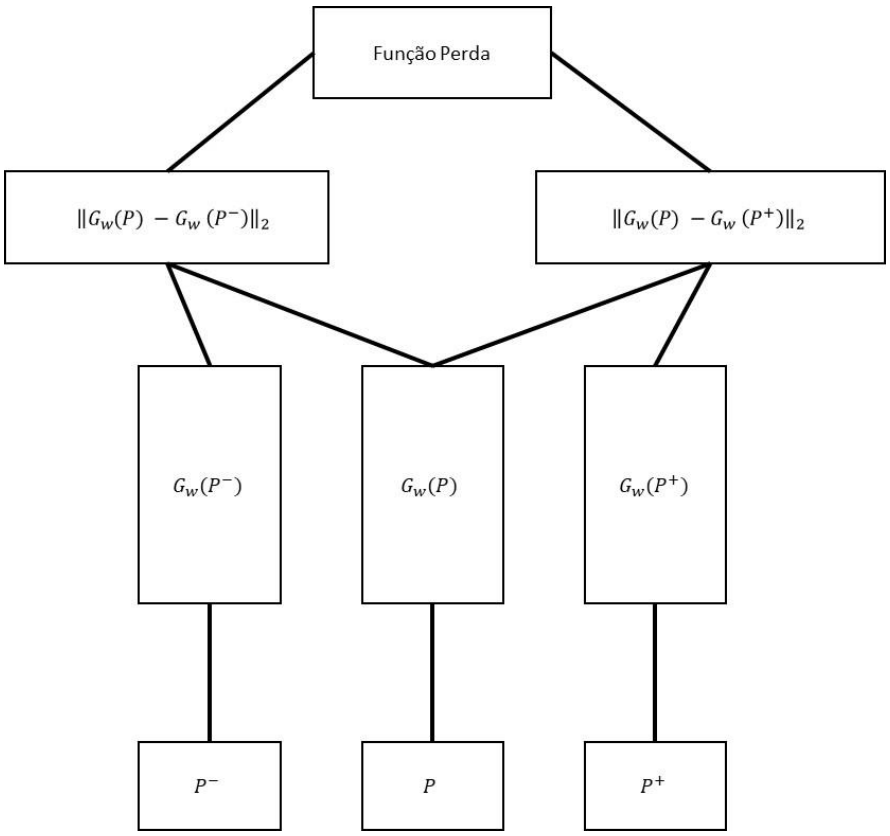
onde

$$d_+ = \frac{e^{\|G_w(P) - G_w(P^+)\|_2}}{e^{\|G_w(P) - G_w(P^+)\|_2} + e^{\|G_w(P) - G_w(P^-)\|_2}}$$

e

$$d_- = \frac{e^{\|G_w(P) - G_w(P^-)\|_2}}{e^{\|G_w(P) - G_w(P^+)\|_2} + e^{\|G_w(P) - G_w(P^-)\|_2}}$$

Figura 6: Rede de Trios



Fonte: Do Autor

Porém, tal função apresenta uma convergência lenta, visto que ela é extremamente dependente das imagens presentes no lote de treino. Se dentro do lote de treino poucas imagens desrespeitam a condição dos trios, o gradiente é pequeno para minimizar a função. Logicamente, em um cenário ideal, seria possível realizar a combinação de todos os trios dentro da base de imagens, mas é computacionalmente inviável dado a grande quantidade de conjuntos que deveriam ser comparados. A partir desse problema, técnicas de amostragem começaram a serem desenvolvidas para buscar de maneira online (dentro do lote) e off-line (na base toda), a melhor maneira de selecionar os trios a fim de acelerar a convergência do modelo.

2.2.3 Semi-Hard Triplet loss

Uma das principais técnicas utilizadas de amostragem foi abordada pela primeira vez para criar um modelo eficiente para identificação de faces das pessoas (Schroff et al., 2015).

Utilizando *Triplet Loss* como função objetivo, foi proposto que dentro do lote fossem selecionados apenas os trios que favorecessem o gradiente para reduzir a função.

A primeiro momento, a ideia principal era selecionar apenas os casos em que a distância euclidiana entre a âncora e a imagem negativa (chamada de "negativa difícil"), fossem menor que a distância entre a âncora e a imagem positiva, isto é $\|G_w(P) - G_w(P^+)\|_2^2 > \|G_w(P) - G_w(P^-)\|_2^2$, no entanto foi observado que tal condição poderia levar o modelo a mínimos locais nos estágios iniciais do treino Dessa forma para mitigar esse problema, optou-se por selecionar apenas as semi-difíceis negativas, isto é, trios em que a distância entre a âncora e a imagem positiva ainda é menor que a âncora e a positiva, porém a distância entre a imagem positiva e negativa é menor que a âncora e a imagem positiva, resumidamente que satisfaçam a equação (2.4),

$$\begin{cases} \|G_w(P) - G_w(P^+)\|_2^2 < \|G_w(P) - G_w(P^-)\|_2^2 \\ \|G_w(P) - G_w(P^+)\|_2^2 > \|G_w(P^+) - G_w(P^-)\|_2^2 \end{cases} \quad (2.4)$$

Apesar de na época ter obtido o melhor resultado na base *Label Faces in the Wild* (LFW) com acurácia de 99,63% e um aumento de 30 pontos percentuais frente ao melhor resultado apresentado até então (Sun et al., 2015). A metodologia de treino requereu um imenso esforço computacional. Para realizar o treino foram necessárias 1800 imagens por lote, sendo necessário um conjunto de CPUS trabalhando em paralelo e com um grande período de treino, de 1000 a 2000 horas. A necessidade desse grande número de imagens foi necessária para que pudesse ter uma amostragem significativa dentro de cada lote. Em virtude disso, novas funções objetivos começaram a focar não apenas no melhor resultado dado uma métrica escolhida (i.e. acurácia), mas também que apresentasse uma convergência rápida e com uma menor necessidade de recursos computacionais.

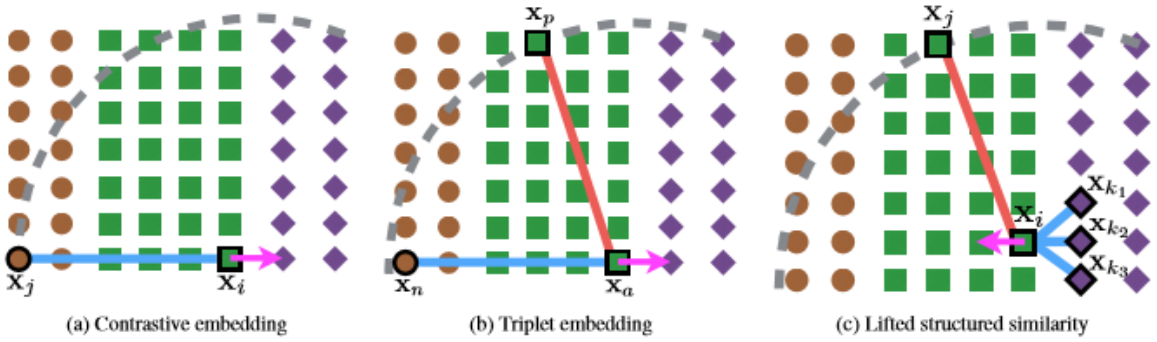
2.2.4 Lifted Struct loss

Uma nova função objetivo chamada Lifted Struct (Song et al., 2016) foi proposta para suprir falhas presentes na *Constrative e Triplet Loss*. A exemplificação das falhas é apresentada na

Figura 7. Para primeira, como mencionada anteriormente, se três imagens de diferentes classes forem colineares, a âncora não irá ser ajustada na melhor posição do espaço. Para a

segunda função, geralmente junto a ela, é utilizada uma margem de distância, para considerar se as imagens estão presentes na mesma classe ou não. Se três imagens forem comparadas e a negativa estiver dentro da margem, a âncora será afastada da negativa, sem considerar os outros vizinhos ao redor.

Figura 7: Comparativo entre funções objetivos – (a) *Constrative Loss*, (b*) Triplet Loss* e (c) *Lifted structured*. Dentro de cada exemplo é apresentando três classes diferentes e a linha cinza é referente a margem. Dentro de cada imagem é apresentado o comportamento de cada âncora ao se minimizar cada função



Fonte: Song et al., 2016

A partir dessas falhas foi proposto uma função objetivo em que todas as imagens dentro do lote sejam consideradas para a função objetivo e dessa forma cada âncora ficaria na melhor posição no espaço, isto é, mais próximo das imagens com a mesma classe e com a máxima distância frente a todas outras imagens. Dessa forma, a ideia inicial era minimizar a equação (2.6), no entanto ela não é suave para otimizar via retro propagação, devido as operações de máximo aninhados, logo foi criada uma função suavizada com o mesmo objetivo de otimização (2.7), onde $\hat{P}$ são os conjuntos de pares positivos (âncora e imagem da mesma classe) , $\hat{N}$ os conjuntos de pares negativos (âncora e imagens de diferentes classes), $\propto$ é a margem e o operador D é a distância euclidiana.

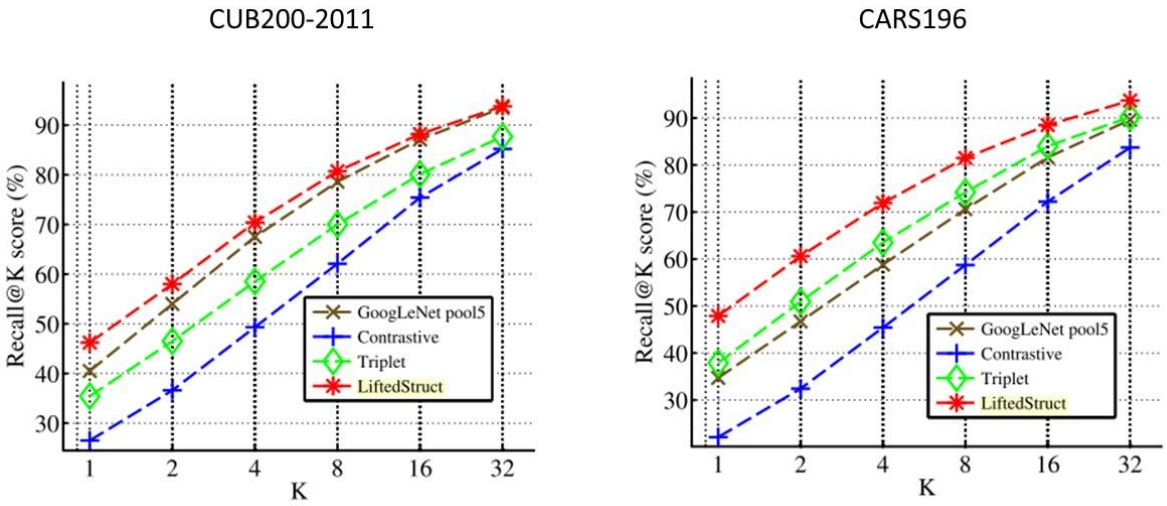
$$Perda\left(\hat{P}\right) = \frac{1}{2|\hat{P}|}\sum_{(i,j)\,\in\,\hat{P}} \max\left(0\,,\ Perda_{i,j}\right)^2, \tag{2.6}$$

$$Perda_{i,j} = \max\left(\max_{(i,k)\in\hat{N}} \propto - D_{i,k}, \max_{(j,l)\in\hat{N}} \propto - D_{j,l}\ \right) + D_{i,j}$$

$$\widetilde{Perda}_{i,j} = \ln\left(\sum_{(i,k)\,\in\,N} \exp(\propto - D_{i,k}) + \sum_{(j,l)\,\in\,N} \exp(\propto - D_{j,l})\right) + + D_{i,j} \tag{2.7}$$

A partir dela e juntamente com uma estratégia de amostragem ponderada para formação de cada lote, isto é , sempre acrescentando as imagens negativas mais próximas entre os pares positivos pela própria distância euclidiana, foi possível superar as funções objetivos abordadas anteriormente em duas bases públicas (CUB200-2011 e CARS196), tal qual a Figura 8 mostra utilizando a métrica de *Recall.* Além disso, essa função objetiva não apresentou perdas de performance significativas, mesmo com um número de lotes reduzido. A partir de 50 imagens por lote, a melhora apresentada pelo aumento se torna numericamente desprezável, tal como a **Tabela 1** apresenta, suprindo desse modo, a necessidade de elevado recurso computacional.

Figura 8: Comparativo entre *Constrative,Triple e LiftedStruct loss* utilizando a métrica de recall, em que considera o percentual de acertos frente aos K vizinhos mais próximos definido pelo modelo



Fonte: (Song et al., 2016)

Tabela 1: Recall @ 1 para bases CUB200 e CARS196 em tamanho de lotes m

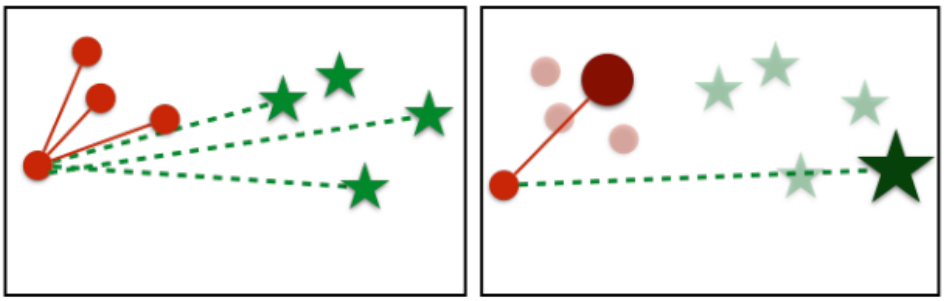
Lotes	CUB200	CARS196
m = 32	42,4	46,9
m= 48	42,1	49,4
m= 64	42,4	50,3
m = 128	42,8	49,5

Fonte: (Song et al., 2016)

2.2.5 Proxy NCA loss

Para evitar a necessidade de uma estratégia de amostragem Movshovitz-Attias et al., 2017, propuseram uma abordagem diferente. Utilizando de uma metodologia de cálculo para vizinhos próximas mais eficiente computacionalmente que KNN, chamada de NCA (Hu, 2013), foi possível manter na memória representantes de cada classe chamadas de "proxy". Ao invés de comparar a posição no espaço entre as imagens, é otimizada a posição da "proxy" frente as demais, dessa forma é preciso um menor número de conjuntos a serem comparados, tal como a Figura 9 apresenta. Devido a isso é esperado uma melhor convergência visto que não é dependente das combinações de trios.

Figura 9: Na imagem à esquerda são apresentadas 48 combinações possíveis de trios (círculos e estrelas). Na direita, os grandes círculos e estrelas são as "proxies", com elas somente 8 trios precisam ser formados.



Fonte: (Movshovitz-Attias et al., 2017)

Rippel et al., 2016 propuseram uma abordagem semelhante (*Magnet loss) ,* em que cada classe era representada por um centroide obtido pelos vizinhos mais próximos. No caso, eram amostrados uma quantidade de vizinhos definidos pertencentes à classes diferentes e a função NCA (Hu, 2013) era minimizada . Porém, para atualizar os centroides era necessário parar o treino periodicamente e atualizar a posição dos centroides, apresentando uma lentidão maior na convergência da rede

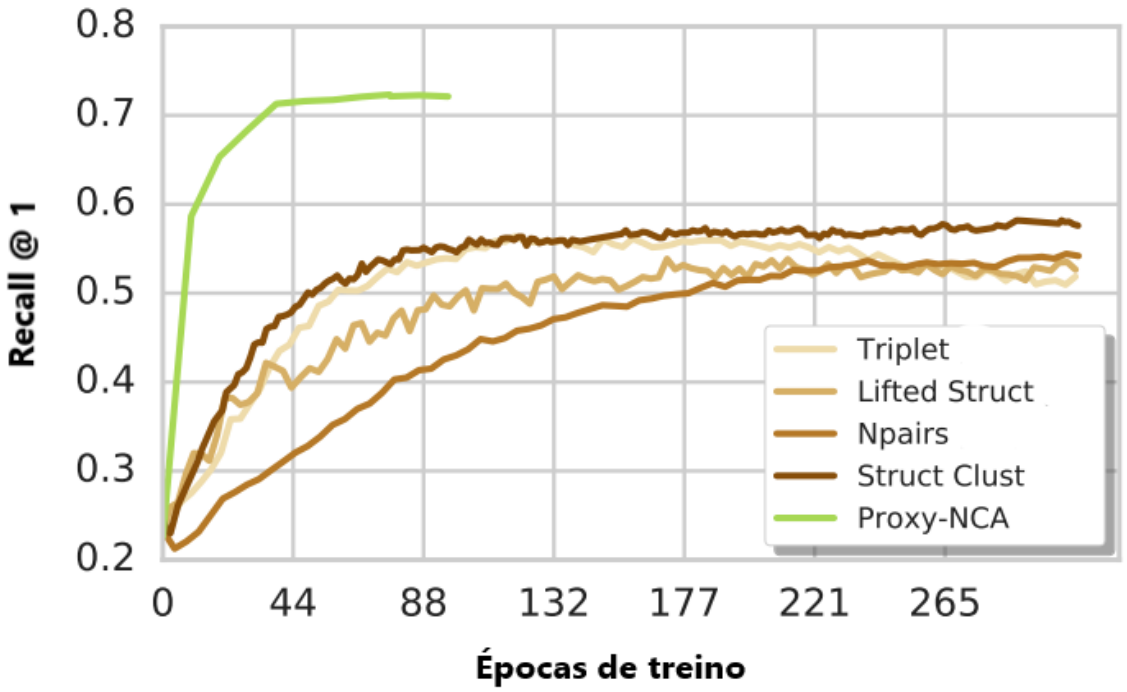
Ao utilizar "proxies" foi possível otimizar os centroides sem pausar o treino, na verdade, elas são uma camada de neurônios totalmente conectados a camada de incorporação com pesos normalizados L2 em que cada neurônio representa um centroide. Ao adotar essa

estratégia, é possível treinar toda estrutura da rede juntamente com a camada de “proxy” simultaneamente através da função objetivo NCA (2.8). Em que x seriam todas as imagens no lote após o *embedding*, y a “proxy” que x pertence (mesma classe), Z as “proxies” não pertencentes a classe x e d é o operador de distância euclidiana.

$$Perda_{NCA}(x, y, Z) = -\ln\left(\frac{\exp(-d(x,y))}{\sum_{z \in Z} \exp(-d(x,z))}\right) \quad (2.8)$$

Como as demais funções apresentadas anteriormente, quando minimizada, ela tende a diminuir a distância entre as imagens de mesma classe e aumentar frente as imagens não pertencentes a elas. Porém, diferente das outras funções a convergência é muito mais acelerada, como a Figura 10 apresenta, ela chega a ser 3 vezes mais rápida e apresenta resultados melhores em métricas usuais como *Recall.*

Figura 10: Análise de convergência na base CARS196 utilizando a métrica *Recall* para a imagem mais próxima frente a âncora. *Proxy-NCA* apresenta uma convergência 3 vezes mais rápida e um *recall* mais aumento frente as outras funções anteriormente apresentadas.



Fonte: Adaptado de Movshovitz-Attias et al., 2017

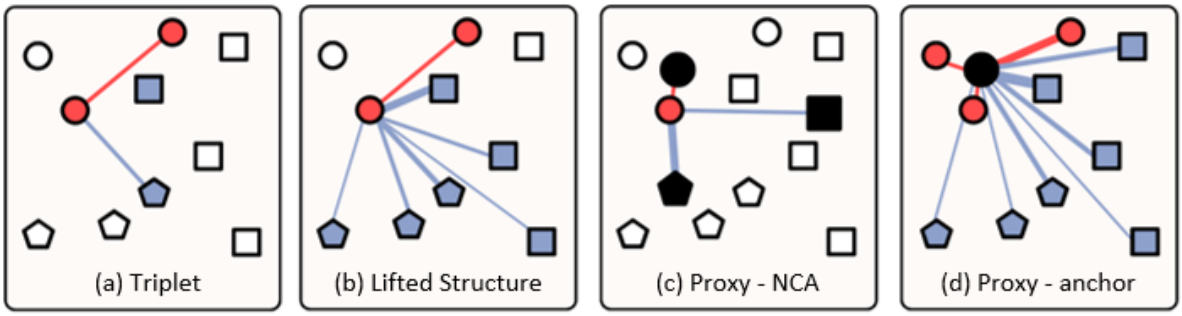
Além disso, os resultados reportados foram realizados com 32 imagens por lote e um *embedding* com 64 dimensões apenas, diferente dos abordados por (Song et al., 2016), em que apresenta uma relação melhor conforme o tamanho do *embedding* aumenta, no caso 512. Por fim, como a relação é sempre frente a "proxy" é apresentado uma maior robustez frente a ruídos na base. Apesar do melhor resultado quando comparado a outras funções objetivos, por se utilizar uma "proxy" como representante das classes, é perdido parte da riqueza da base, apresentando problemas quando a diferença semântica entre as classes é pequena.

2.2.6 Proxy anchor loss

Uma melhoria frente a Proxy-NCA foi proposta por Kim et al., 2020, em que considera a vantagem de ambas estratégias, a de comparar imagens na base a qual apresenta uma informação semântica mais rica e a comparação de imagens com a "proxy" em que apresenta uma convergência muito mais acelerada e não é dependente de estratégias de amostragem.

Diferentemente da Proxy-NCA (Movshovitz-Attias et al., 2017), que só leva em consideração a relação entre as "proxies" e cada ponto Figura 11(c) ou da Lifted Structure (Song et al., 2016) que leva em relação entre os pares de imagem, mas que a convergência é limitada as técnicas de amostragem (Figura 11(b)), o autor propõe uma relação das imagens com as "proxies", de tal forma que leve em consideração todas as imagens do lote, como a dificuldade de separação entre imagens, isto é, imagens que apresentam uma semântica mais parecida (Figura 11(d)).

Figura 11: comparativo de estratégia das funções objetivos apresentados. Os pontos são os *embedding* no lote, cada formato representa uma classe e em cores pretas são as "proxies". As espessuras das linhas representam o gradiente, quanto mais espessa maior o gradiente e cores vermelhas são relação entre mesma classe e a azul entre classes diferentes.



Fonte: Adaptado de Kim et al., 2020

Para superar a limitação da *Proxy-NCA,* mantendo a complexidade do treino baixa, isto é, sem ter que utilizar técnicas de amostragem, foi proposto pelo autor a utilização da equação (2.9) , onde X são os vetores *embeddings,* P é o conjunto de todas as "proxies", sendo que P^+ seria o conjunto das proxies que estão presentes no lote, $\propto$ é um fator de escala (temperatura) e δ é a margem.

$$Perda_{P-anchor}(X) = \frac{1}{|P^+|}\sum_{p \in P^+} \ln\left(1 + \sum_{x \in X_p^+} e^{-\propto(s(x,p)-\delta)}\right) + \frac{1}{|P|}\sum_{p \in P} \ln\left(1 + \sum_{x \in X_p^+} e^{\propto(s(x,p)+\delta)}\right) \quad (2.9)$$

Quando os gradientes da Proxy-NCA (2.40) e Proxy-anchor (2.51) são comparados é observado a principal diferença. No primeiro, se a classe pertence a mesma "proxy" é apresentando um gradiente constante e para os casos não pertencentes o gradiente é calculado apenas levando em consideração uma parcela das "proxies". Devido, principalmente, ao gradiente constante para os casos de mesma classe, a convergência não é tão rápida quanto Proxy-anchor em que o gradiente não é só são afetados pelo próprio *embedding,* mas também por todos os outros dentro do lote.

$$\frac{\partial Perda_{NCA}(X)}{\partial s(x,p)} = \begin{cases} -1, & se\ p = p^+, \\ \dfrac{e^{\,s(x,p)}}{\sum_{p^- \in P^-} e^{s(x,p^-)}}, & caso\ contrário \end{cases} \quad (2.40)$$

$$\frac{\partial Perda_{P-anchor}(X)}{\partial s(x,p)} = \begin{cases} \frac{1}{|P^+|} \frac{-\propto h_p^+(x)}{1+\sum_{z \in X_p^+} h_p^+(z)}, \ \forall\, x \in X_p^+ , \\ \frac{1}{|P|} \frac{\propto h_p^-(x)}{1+\sum_{z \in X_p^-} h_p^-(z)} , \ \forall\, x \in X_p^- \end{cases} \quad (2.51)$$

Onde

$$h_p^+ = e^{-\propto(s(x,p)-\delta)} \ e \ h_p^- = e^{\propto(s(x,p)+\delta)}$$

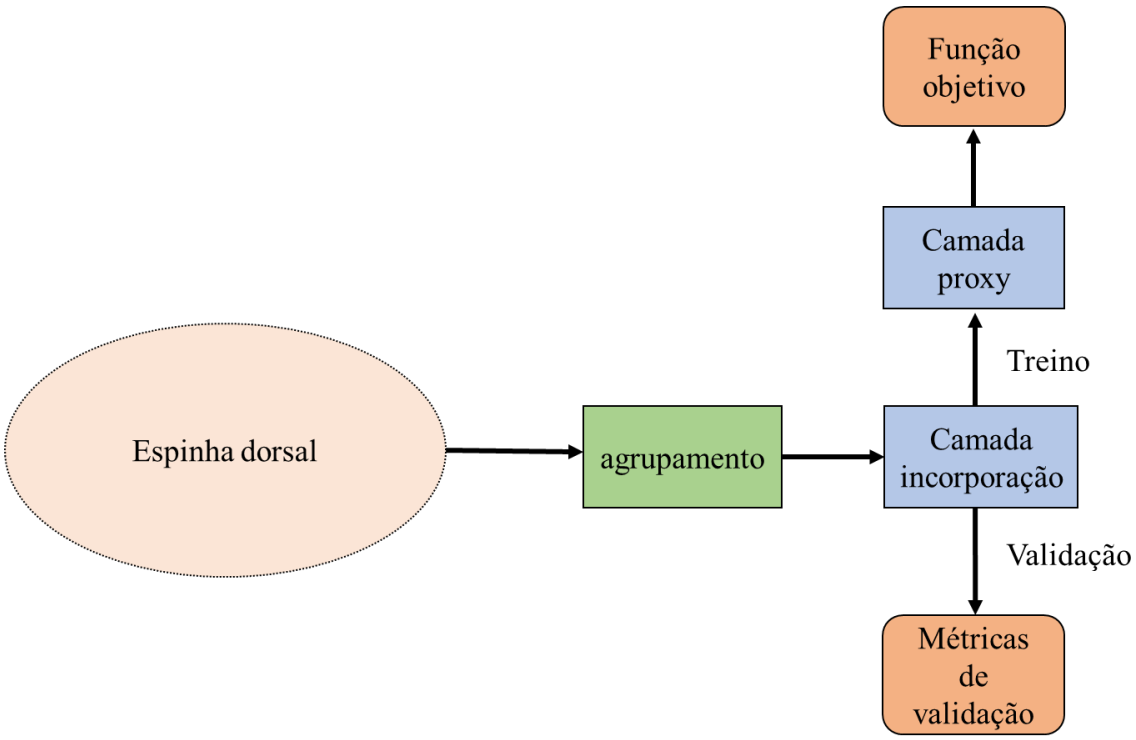
3 Materiais e Métodos

Visando a construção de um modelo adequado para a similaridade de imagens foi utilizada a base de dados *Standford Online Products* (Song et al., 2016) por se tratar de uma base extremamente heterogênea de produtos encontrados no *ebay*. A base consiste em 120 mil imagens de 23 mil objetos diferentes. Essa base foi separada em duas partes: a parte do treino, contendo 59551 imagens de 11318 objetos diferentes; e outra parte para teste com 60502 imagens diferentes das imagens utilizadas no treino. Isto é, o principal objetivo é treinar um modelo que seja flexível para identificar imagens semelhantes pertences a outras classes nunca vistas, conhecido como *zero-shooting learning* (aprendizado na tentativa zero).

Para criação do modelo foi utilizada a linguagem de programação *Python*, mais especificamente as bibliotecas *tensorflow e keras* para trabalhar com redes neurais. Dentro delas é apresentado uma ampla gama de modelos pré-treinados, otimizadores e praticidade na hora de estruturar novos elementos na rede.

A partir da escolha do software, foi necessário definir a topologia da rede. Para todos os testes, a principal topologia é apresentada da seguinte forma, como mostra a Figura 12. Primeiro é utilizado um modelo pré-treinando como espinha dorsal, tal qual os apresentados anteriormente. Nesse modelo é removida a última camada de neurônios totalmente conectados e substituída por um agrupamento global, que pode ser utilizando o máximo valor de uma região ou a média dos valores dessa região. Por fim, após o agrupamento, é conectada a camada de incorporação. Para esse estudo foi utilizado como espinha dorsal a topologia ResNet50 pré-treinanda com a base ImageNet.

Figura 12: Representação da estrutura básica da rede para funções de perda baseadas em *proxy*.



Fonte: Do Autor

Das funções objetivas destacadas foi realizado um estudo de caso para a *Proxy NCA,* por essa ter apresentado um dos melhores resultados, necessitar de um menor esforço computacional (menor número de imagens por lote e menor dimensão do *embedding*) e apresentar uma convergência mais rápida frente as outras metodologias. Para que seja possível utilizar essa função é necessário mais uma camada de neurônios totalmente conectados de dimensão igual a quantidade de classes dentro da base de treino. Tal escolha é a melhor em termos de resultado, como reportado por Movshovitz-Attias et al. (2017). Essa camada apresentam todos os pesos normalizados L2 que serão otimizados por retro propagação juntamente com a rede. É importante ressaltar, que apesar da criação da camada *proxy* na estrutura do modelo, ela não participa na validação do mesmo. A saída para validação é feita pela camada de incorporação, a qual apresenta também como parâmetro ajustado a sua dimensão.

O treinamento do modelo foi realizado com 105 imagens por lote para 20 épocas, as quais foram redimensionadas para uma proporção de 299x299, em seguida recortadas

aleatoriamente para 256x256 e horizontalmente espelhadas. Para a validação as imagens foram somente redimensionadas para 256x256. Para treinar a rede Resnet50 foi utilizado como otimizador AdamW (Loshchilov e Hutter, 2017), com uma taxa de aprendizado de 10^{-4}, queda de pesos de 10^{-4} e um multiplicador de 10^{3} na taxa de aprendizado da camada da proxy, visto que a velocidade de convergência é diferente da outra parte da rede, principalmente devido aos seus pesos serem normalizados.

Outras melhorias também foram avaliadas, como a melhor estratégia de agrupamento (média ou máximo), a utilização de uma camada de Dropout (Srivastava et al., 2014), camadas de normalização (Ba et al., 2016) e parâmetros ajustáveis das funções de perda. Para se obter a melhor combinação de parâmetros foi realizada uma busca exaustiva visando obter o máximo valor de revogação para a imagem mais próxima da âncora, para cinco épocas.

4 Resultados

Conforme abordado em materiais e métodos, com a busca exaustiva dos parâmetros foi possível avaliar o impacto separado de cada um deles. Para tanto, será discutido cada parâmetro para melhor entendimento do impacto de cada um. Além disso, com a combinação dos melhores parâmetros foi feita a avaliação da performance do modelo frente à literatura, tanto em velocidade de convergência, como também de melhor performance.

4.1 Avaliação de parâmetros

4.1.1 Temperatura (fator de escala)

A ideia da adição da temperatura (fator de escala) como parâmetro veio a partir de um pequeno ajuste frente à função de perda *Proxy NCA* (2.8) em que, ao invés de considerar no denominador a relação de distância da entrada do *embedding* frente as "proxies" não pertencentes à classe, é considerado a distância da entrada do *embedding* frente a todas as "proxies". Dessa forma a função se assemelha a uma função de *softmax* e podemos maximizar a probabilidade de uma imagem pertencer a uma "proxy" definida. Mais além, devido a essa característica, é possível introduzir o parâmetro temperatura. De acordo com Agarwala et al. (2020), o ajuste dessa parâmetro pode trazer ganhos significativos em acurácia, mais além, esse parâmetro é característico de cada espinha dorsal utilizada no modelo. A equação (2.8) ajustada é apresentada abaixo, onde A representa todo conjunto de "proxies" e T a temperatura.

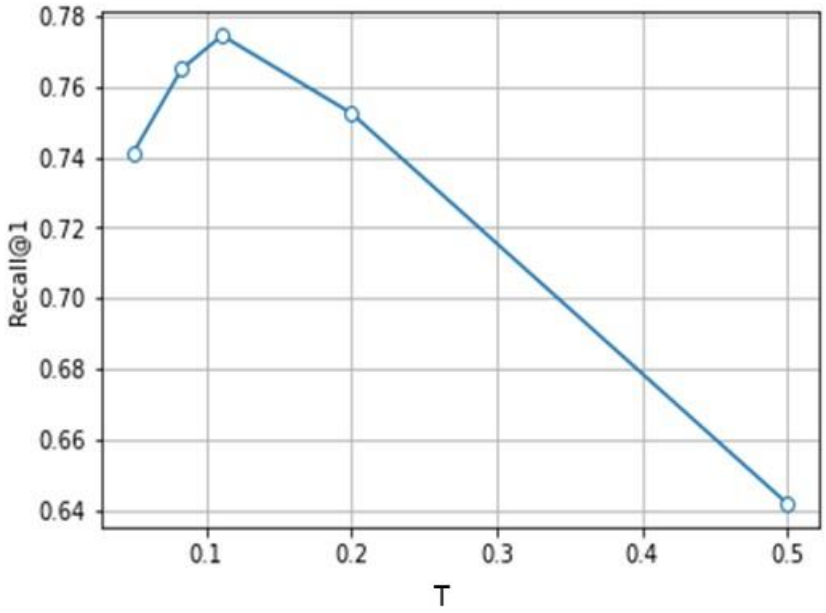
$$Perda_{NCA}(x, y, Z) = -\ln\left(\frac{\exp(-d(x,y)/T)}{\sum_{a \in A} \exp(-d(x,a)/T)}\right) \tag{4.1}$$

Quando $T \rightarrow \infty$,a distribuição se torna uniforme. Porém, para baixos valores de *T*, os limites de decisão entre uma classe e outra são menos suavizados, a um ponto que se *T* for muito baixo começa a apresentar sobre ajuste. Dessa forma, *T* é um parâmetro ajustável que determinar o quanto cada imagem influencia na probabilidade de pertencer a uma "proxy", com $T \rightarrow \infty$, cada imagem tem a mesma probabilidade de pertencer a cada "proxy", e quando *T* é baixo, cada imagem tem um viés maior de pertencer a uma "proxy" específica. Tal recurso é ajuda a controlar os ruídos presentes na base.

Para esse experimento em questão, para a rede ResNet50, o melhor *T* foi de 1/9. A

Figura 13 apresenta o impacto na revogação conforme o *T* diminui.

Figura 13: Relação de temperatura com revogação para primeira imagem mais próximo em um *embedding* de 512, sem camada de normalização e com agrupamento médio.

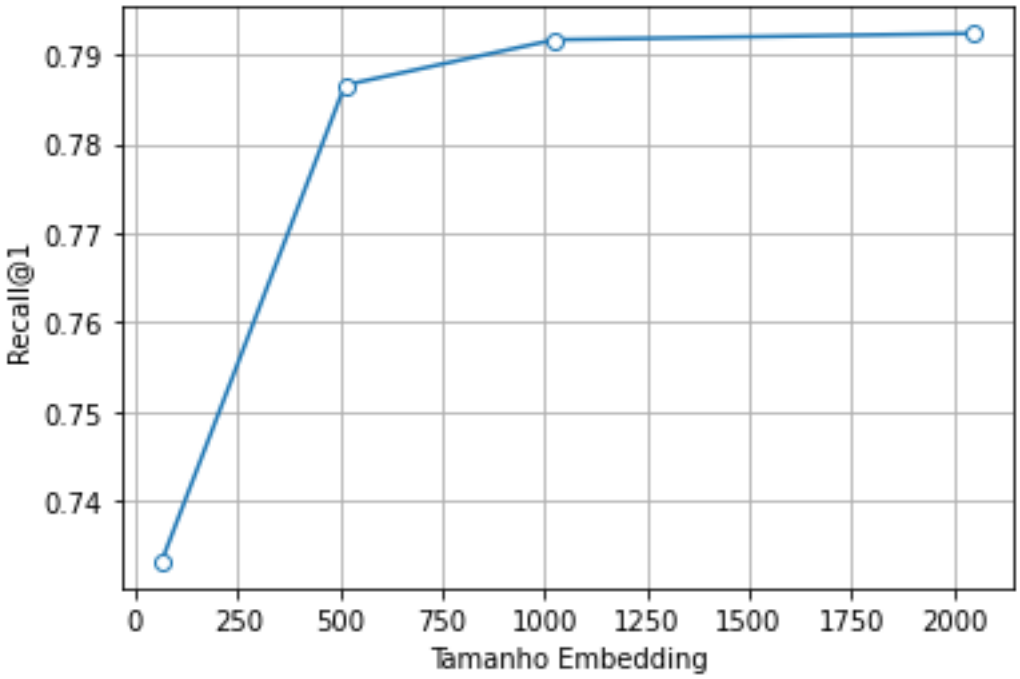


Fonte: Do autor

4.1.2 Dimensão da camada de incorporação

A dimensão da camada de incorporação foi avaliada, pois quanto maior sua dimensão, mais pesos ajustáveis o modelo teria. No entanto, nota-se pela Figura 14 que as dimensões maiores que 512 não apresentam um ganho considerável em revogação. Tal feito ocorre devido ao excesso de pesos do modelo que tendem, novamente, a um sobre ajuste. A variação do tamanho de 512 (78,6%) para 2048 (79,23%) apresenta apenas 0,6 pontos percentuais de melhora no modelo.

Figura 14: Impacto da dimensionalidade do *embedding* em revogação



Fonte: Do autor

4.1.3 Agrupamento

Em todas as avaliações da literatura o principal agrupamento utilizado era das médias, porém foi realizado um teste utilizando agrupamento máximo para avaliar se apresenta um ganho. A **Tabela 2** apresenta o impacto na revogação diante da escolha do agrupamento. Nota-se que a utilização do agrupamento médio ainda é melhor frente ao máximo, apresentando um ganho de aproximadamente 1 ponto percentual frente ao retorno da imagem mais próxima.

Tabela 2: Relação da estratégia de agrupamento

Recall	Agrupamento médio	Agrupamento máximo
@1	77,4%	76,6%
@3	84,5%	83,8%
@5	86,9%	86,3%
@100	95,5%	95,4%

4.1.4 Camada de normalização

A camada de normalização (Ba et al., 2016) foi avaliada inicialmente para redes recorrentes para a identificação de vocabulários. Porém, como no final de sua estrutura é apresentado um vetor *embedding* há uma similaridade para identificação de vocabulários e o modelo em questão. No entanto, pela busca de parâmetros exaustivos, nesse caso não foi apresentando um ganho efetivo ao adicionar essa camada, não apresentando uma variação expressiva na métrica de revogaçao, tal qua a **Tabela 3** apresenta. O uso dessa camada apresentou uma perda de cerca de um ponto percentual também.

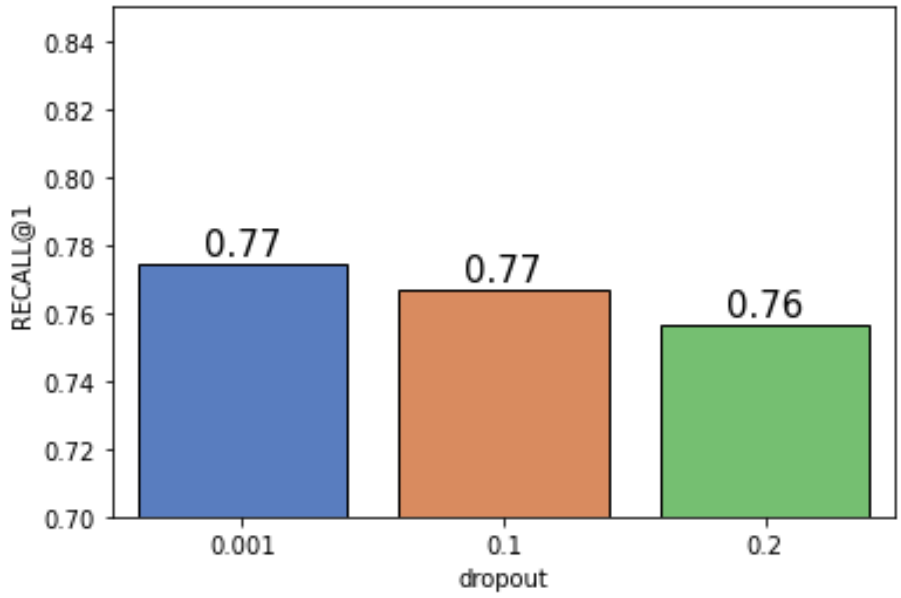
Tabela 3: Impacto do usa da camada de normalização na revogação

Recall	Sem camada	Com camada
@1	77,4%	76,9%
@3	84,5%	84,1%
@5	86,9%	86,6%
@100	95,5%	95,4%

4.1.5 Dropout

A camada de *dropout* foi criada principalmente para controlar sobre ajuste. A cada passo são escolhidos aleatoriamente pesos a serem zerados, diminuindo assim a quantidade de parâmetros a serem ajustados em cada iteração. Durante o treino é notada uma convergência acelerada do modelo e em torno de 12 épocas já é observado um sobre ajuste, em que a função de perda do treino é diminuída, mas não é visto um ganho na revogação na base de teste. Com isso, foi realizada uma tentativa de controle desse sobre ajuste com uso de *dropout*. Porém, foi observada uma correlação negativa entre o aumento da taxa de pesos zerados frente a revogação, como mostra a Figura 15. Dessa forma, foi optado por não utilizar essa camada para treinar a rede.

Figura 15: Relação revogação com taxa de *dropout* para dimensão 512 do *embedding*, agrupamento médio, sem camada de normalização e temperatura de 1/9.



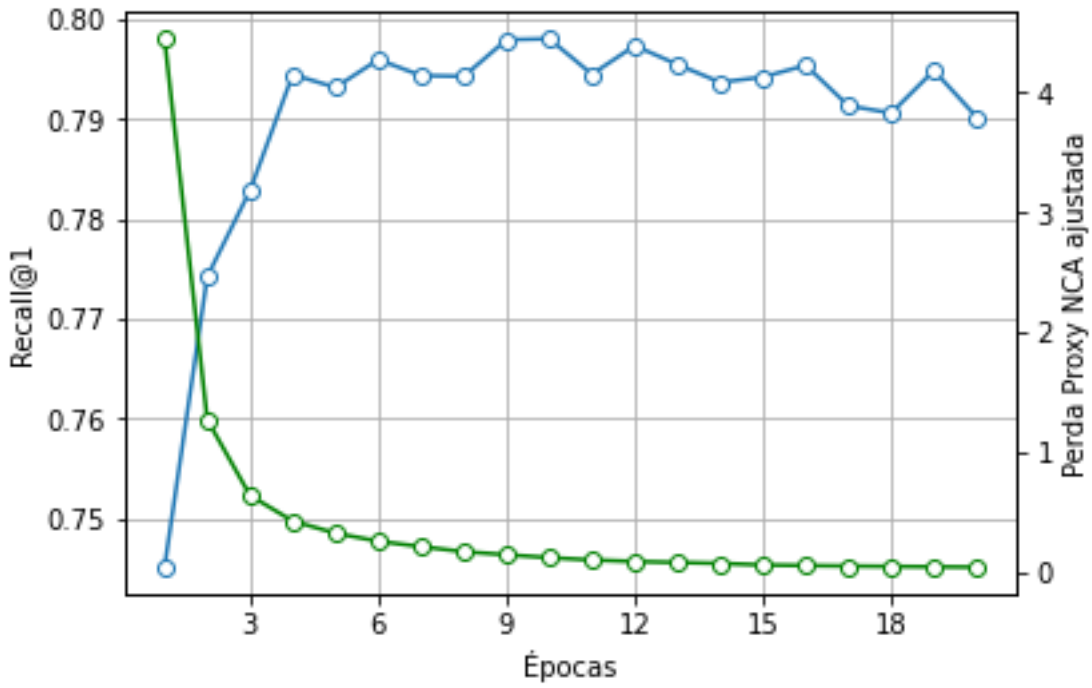
Fonte: Do autor

4.1.6 Curva de treino e convergência

Após avaliar qual a melhor estrutura e quais parâmetros apresentam o melhor resultado foi avaliada a convergência do modelo frente a metodologias anteriores. A convergência mais rápida era representada pela *Proxy NCA,* em que o modelo convergia em 44 épocas, utilizando como espinha dorsal a rede Inception V3. Ao trocar para a rede ResNet50 e

utilizar como parâmetro ajustável a temperatura foi possível acelerar a convergência do modelo e em 10 épocas já era possível obter o máximo do modelo em revogação, tal qual a Figura 16 apresenta. A partir da décima época, apesar da perda ainda diminuir, não há uma melhora efetiva em revogação, inclusive é apresentado uma queda no final (após a décima oitava época), indicando sobre ajuste. Para definir o modelo foi escolhido a época que na base de teste apresentou o maior valor de revogação.

Figura 16: Evolução da revogação (recall@1) e perda *Proxy NCA ajustada* ao longo das épocas para o modelo com dimensão de 2048 *embedding*, agrupamento médio e sem camada de normalização.



Fonte: Do autor

4.2 Avaliação final do modelo

Com os melhores resultados obtidos foi feito uma comparação frente à literatura. É importante ressaltar que parte das avaliações da literatura não foram realizadas com o mesmo tamanho de imagem (256x256), o que leva o modelo em questão apresentar uma revogação um pouco inflacionada. Porém, no geral, o resultado obtido ficou muito próximo dos melhores estados de arte desenvolvidos até então, como *proxy anchor* (Kim et al., 2020). A **Tabela 4** presenta os melhores resultados obtidos para revogação na literatura frente ao modelo obtido. O modelo atual chega a superar metodologias como *triplet semi-hard*, *Lifted Struct* e

Proxy NCA, superando o último por mais de 6 pontos percentuais. Frente ao *Horde* (Jacob et al., 2019), o resultado obtido ficou 0,3 ponto percentual abaixo, porém essa metodologia apresenta uma convergência extremamente lenta e exige muitos recursos computacional. Frente a *Proxy Anchor*, o resultado ficou 0,5 ponto percentual abaixo.

Tabela 4: Comparativo de Revogação frente a literatura

Função de Perda	Recall @1	Tamanho imagem	Tamanho embedding
Triple Semi-Hard	66,7%	(227x277)	512
Lifted Struct	62,5%	(227x277)	512
Proxy NCA	73,3%	(227x277)	64
Modelo atual	79,8%	(256x256)	2048
Horde (Jacob et al., 2019)	80,1%	(256x256)	512
Proxy Anchor	80,3%	(256x256)	512

Outra análise realizada foi extrair as 5 imagens mais próximos de uma âncora, tal qual a Figura 17 apresenta. Nota-se que a maioria das imagens mais próximas da âncora são pertencentes a mesma classe e as que não pertencem apresentam uma semântica muito parecida.

Figura 17: Imagens retornadas mais próximas da âncora. Imagens com X são de classes diferentes da âncora

Fonte: Do autor

5 Conclusões e Trabalhos Futuros

Nesse trabalho foi avaliado um estudo de parâmetros para que se pudesse obter um modelo de rede neural capaz de reconhecer imagens por similaridade e obter resultados em revogação próximo aos principais estados de arte. O principal ganho do estudo foi a modificação da função objetivo Proxy-NCA (2.8) para a equação (5.1), que possibilita a maximização da probabilidade de cada imagem pertencer a uma "*proxy*" como também a adição do parâmetro de temperatura que apresentou ganhos de até 10 pontos percentuais na revogação. Os outros parâmetros como *dropout* e camadas de normalização não apresentaram um ganho ao modelo. Levando esses pontos em consideração foi possível obter uma rede com revogação de 79,8% para a imagem mais próxima, apresentando 0,5 ponto percentual a menos frente ao estado de arte atual.

O principal ganho do modelo é a utilização dele para bases extremamente heterogêneas, que pode realizar o treino em um certo tipo de base de imagens e aplicá-lo em outras bases de imagens jamais vistas, visto que o modelo tem por objetivo aprender a semântica e ordená-las por semelhança. Outra vantagem do modelo é o fato da camada de incorporação ser invariante, isto é, o resultado do reconhecimento da imagem é pouco afetado pelo tratamento da imagem, seja por rotação, recorte ou mudanças de escala de cor, se apresentando um modelo robusto, independente da condição da imagem.

Aprendizado por métricas de distância é uma área relativamente nova estudada em redes neurais, com desenvolvimentos recentes até em 2020. É uma área promissora para reconhecimento de imagens em bases grandes, ainda mais nos dias de hoje, em que há um volume muito grande de dados com dimensionalidade elevada. Dessa forma, a avaliação se torna de difícil interpretação e com limitação computacional. Com a estratégia do trabalho apresentado é possível reduzir a dimensionalidade dessas bases e obter representações suficientemente precisas para diversas aplicações, como identificação de objetos roubado, sistemas de recomendação para usuário, reconhecimento de face, além de aplicações na medicina como segmentação de tipos de carcinoma (Zhou, 2018) ou identificação de nódulos no pulmão (Wei et al., 2020).

Certas melhorias poderiam ser implementadas em trabalhos futuros, tal como utilização de redes como espinha dorsal mais complexas como Inception V4 (Szegedy et al., 2017) e

estruturas, juntamente com essa espinha dorsal, que melhorassem a identificação de imagens mais refinadas como a estrutura proposta por Wang et al., 2014. Além disso, uma limitação dessa técnica é que é necessário rodar o modelo para todas as imagens da base para obter um comparativo. Dessa forma, bases muito grandes demorariam muito para se ter uma validação dos resultados. Uma estratégia interessante seria de criar grupos que representassem as imagens e calcular a distância da imagem frente a esses grupos para se filtrar a base antes e comparar frente a um grupo menor.

6 Referências

AGARWALA, A.; PENNINGTON, J.; DAUPHIN, Y.; et al. Temperature check: theory and practice for training models with softmax-cross-entropy losses. 2020.

BA, J. L.; KIROS, J. R.; HINTON, G. E. Layer Normalization. 2016.

HADSELL, R.; CHOPRA, S.; LECUN, Y. Dimensionality reduction by learning an invariant mapping. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, v. 2, p. 1735–1742, 2006.

HAMILTON, M.; SHELHAMER, E.; FREEMAN, W. It Is Likely That Your Loss Should be a Likelihood. **arXiv**, n. 3, 2020.

HE, K.; ZHANG, X.; REN, S.; et al. Identity mappings in deep residual networks. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, v. 9908 LNCS, p. 630–645, 2016.

HOFFER, E.; AILON, N. Deep metric learning using triplet network. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, v. 9370, n. 1271, p. 84–92, 2015.

HOU, Y.; ZHANG, P.; XU, X.; et al. Nonlinear dimensionality reduction by locally linear inlaying. **IEEE Transactions on Neural Networks**, v. 20, n. 2, p. 300–315, 2009.

HU, H. Enhanced gabor feature based classification using a regularized locally tensor discriminant model for multiview gait recognition. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 23, n. 7, p. 1274–1286, 2013.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. **32nd International Conference on Machine Learning, ICML 2015**, v. 1, p. 448–456, 2015.

JACOB, P.; PICARD, D.; HISTACE, A.; et al. Metric learning with HORDE: High-order regularizer for deep embeddings. **arXiv**, 2019.

JIA DENG; WEI DONG; SOCHER, R.; et al. ImageNet: A large-scale hierarchical image database. p. 248–255, 2009.

KIM, S.; KIM, D.; CHO, M.; et al. Proxy anchor loss for deep metric learning. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, p. 3235–3244, 2020.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet classification with deep convolutional neural networks. **Communications of the ACM**, v. 60, n. 6, p. 84–90, 2017.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; et al. A B7CEDGF HIB7PRQTSUDGQICWVYX HIB edCdSISIXvg5r ` CdQTw XvefCdS. **proc. OF THE IEEE**, 1998.

LOSHCHILOV, I.; HUTTER, F. Decoupled weight decay regularization. **arXiv**, 2017.

MOVSHOVITZ-ATTIAS, Y.; TOSHEV, A.; LEUNG, T. K.; et al. No Fuss Distance Metric Learning Using Proxies. **Proceedings of the IEEE International Conference on Computer Vision**, v. 2017- Octob, p. 360–368, 2017.

OKATANI, T. **Python Deep Learning 2nd**. [s.l: s.n.]. v. 33

RIPPEL, O.; PALURI, M.; DOLLAR, P.; et al. Metric learning with adaptive density discrimination. **4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings**, n. Dml, p. 1–15, 2016.

SCHROFF, F.; KALENICHENKO, D.; PHILBIN, J. FaceNet: A unified embedding for face recognition and clustering. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, v. 07-12- June, p. 815–823, 2015.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. **3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings**, p. 1–14, 2015.

SONG, H. O.; XIANG, Y.; JEGELKA, S.; et al. Deep Metric Learning via Lifted Structured Feature Embedding. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, v. 2016- Decem, p. 4004–4012, 2016.

SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; et al. Dropout: A simple way to prevent neural networks from overfitting. **Journal of Machine Learning Research**, v. 15, p. 1929–1958, 2014.

SUN, Y.; WANG, X.; TANG, X. Deeply learned face representations are sparse, selective, and robust. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, v. 07-12- June, p. 2892–2900, 2015.

SZEGEDY, C.; LIU, W.; JIA, Y.; et al. Going deeper with convolutions. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, v. 07-12- June, p. 1–9, 2015.

SZEGEDY, C.; VANHOUCKE, V.; IOFFE, S.; et al. Rethinking the Inception Architecture for Computer Vision. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, v. 2016- Decem, p. 2818–2826, 2016.

SZEGEDY, C.; IOFFE, S.; VANHOUCKE, V.; et al. Inception-v4, inception-ResNet and the impact of residual connections on learning. **31st AAAI Conference on Artificial Intelligence, AAAI 2017**, p. 4278–4284, 2017.

WANG, J.; SONG, Y.; LEUNG, T.; et al. Learning fine-grained image similarity with deep ranking. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, p. 1386–1393, 2014.

WEI, G.; QIU, M.; ZHANG, K.; et al. A multi-feature image retrieval scheme for pulmonary nodule diagnosis. **Medicine (United States)**, v. 99, n. 4, p. 1–8, 2020.

XU, B.; WANG, N.; CHEN, T.; et al. Empirical Evaluation of Rectified Activations in

Convolutional Network. 2015.

ZHOU, Z. M. S. Z. X. W. H. Z. W. S. S. AND J. Nasopharyngeal Carcinoma Segmentation based onEnhanced Convolutional Neural Networks usingMulti-modal Metric Learning. 2018.