Comparação de Técnicas para Representação Vetorial de Imagens com Redes Neurais para Aplicações de Recuperação de Produtos do Varejo

Gustavo Ramos Lima
gustavo.r.lima91@gmail.com
Programa de Pós-graduação em Computação Aplicada (PPComp), Instituto Federal do Espírito Santo
Serra, ES, Brasil

Patrick Marques Ciarelli
patrick.ciarelli@ufes.br
Departamento de Engenharia Elétrica
Universidade Federal do Espírito Santo
Vitória, ES, Brasil

Thiago Oliveira-Santos
todsantos@inf.ufes.br
Departamento de Informática
Universidade Federal do Espírito Santo
Vitória, ES, Brasil

Filipe Mutz
filipe.mutz@ifes.edu.br
Programa de Pós-graduação em Computação Aplicada (PPComp), Instituto Federal do Espírito Santo
Universidade Federal do Espírito Santo
Vitória, ES, Brasil

ABSTRACT

Product retrieval from images has multiple applications ranging from providing information and recommentations for customers in supermarkets to automatic invoice generation in smart stores. However, this task present important challenges such as the large number of products, the scarcity of images of items, differences between real and iconic images of the products, and the constant changes in the portfolio due to the addition or removal of products. Hence, this work investigates ways of generating vector representations of images using deep neural networks such that these representations can be used for product retrieval even in face of these challenges. Experimental analysis evaluated the effect that network architecture, data augmentation techniques and objective functions used during training have on representation quality. The best configuration was achieved by fine-tuning a VGG-16 model in the task of classifying products using a mix of Randaugment and Augmix data augmentations and a hierarchical triplet loss as a regularization function. The representations built using this model led to a top-1 accuracy of 80,38% and top-5 accuracy of 92.62% in the Grocery Products dataset.

KEYWORDS

Artificial Neural Networks, Convolutional Neural Network, Image Recovery, Products Recognition.

1 INTRODUÇÃO

Segundo relatório da empresa *Facts and Factors*, responsável por pesquisas de mercado, em 2019 as aplicações de inteligência artificial no mercado global de varejo foram estimadas em USD 2,7 bilhões e a previsão é que atinjam o valor de USD 20,05 bilhões em 2026, com uma taxa de crescimento anual de 39% de 2020 a 2026 [1]. A disseminação da cultura *data driven*, que busca utilizar dados para orientar a tomada de decisão, tem feito com que grandes empresas invistam cada vez mais em soluções de inteligência artificial. Já que por meio dessas é possível automatizar operações reduzindo os custos operacionais. Um exemplo é o desenvolvimento de robôs (*bots*) para auxiliar os clientes, ajustar automaticamente de preços, melhorar o gerenciamento da cadeia de suprimentos e logística e oferecer uma experiência de compra personalizada [2].

A evolução do poder computacional das máquinas tem dado suporte ao desenvolvimento de novas soluções usando inteligência artificial [3]. Estes desenvolvimentos têm chegado ao setor supermercadista [4] e levado à criação de lojas cada vez mais automatizadas. A identificação de produtos através de imagens é uma ferramenta que pode oferecer redução de gastos, pois não é necessário realizar a compra de uma grande quantidade de equipamentos e há uma melhor experiência para os clientes durante as compras, pois é possível utilizar o recurso em *smartphones*. Na loja Amazon Go [5], por exemplo, clientes precisam apenas se cadastrar no aplicativo, pegar os produtos e sair do estabelecimento. No exemplo citado, todos os produtos que o cliente selecionou são computados por um sistema com câmeras que utiliza algoritmos avançados para a identificação de imagens. A loja dispensa, assim, caixas e filas.

Os algoritmos utilizados em tais soluções são, em geral, técnicas de aprendizagem profunda (em inglês *Deep Learning*) [6], que é uma sub-área de aprendizagem de máquina, que por sua vez é um subconjunto da área de inteligência artificial. Modelos baseados em aprendizagem profunda tem se destacado nos últimos anos por apresentarem resultados satisfatórios e por serem capazes de aprender a partir de grandes volumes de dados [7].

Em um supermercado, o processo de compra de um cliente envolve, por exemplo, as etapas de ir até uma sessão do supermercado, verificar nas prateleiras os produtos de interesse, identificar e comparar os preços e colocar o produto desejado no carrinho de compras. Durante esse processo, o consumidor pode não conseguir identificar o preço do produto na prateleira, ou então pode não ter informações mais detalhadas sobre o produto que está adquirindo, impactando negativamente a compra. No primeiro caso, a utilização de uma máquina de leitura de código de barras, por exemplo, poderia ser utilizada para a aferição do preço do produto, sendo necessário o deslocamento até o equipamento, o que pode desestimular a compra. Já para o caso do cliente não conseguir ter acesso às informações mais detalhadas do produto, ele precisaria

da assistência de outra pessoa ou precisaria buscar na internet mais detalhes sobre o produto, por exemplo.

Realizar a identificação de produtos através de imagens torna-se uma alternativa interessante e que pode auxiliar no processo de compra. Imagine poder apontar a câmera do celular para um produto e conseguir obter informações detalhadas como preço, descrição do produto e avaliações de outros clientes. Isso poderia oferecer uma melhor experiência, além de auxiliar pessoas com deficiência visual a obter mais informações sobre os produtos, que muitas vezes vêm impressas na caixa e são pouco legíveis [8]. Tal sistema também poderia levar ainda ao aumento do número de vendas dado que produtos relacionados poderiam ser recomendados aos clientes.

Apesar das vantagens mencionadas anteriormente, a identificação de produtos através de imagens é uma tarefa complexa, já que na maioria das vezes existem poucas imagens por produto para treinamento dos algoritmos [9]. Outro problema está relacionado à qualidade das imagens capturadas no tempo de uso do sistema que podem não estar em condições ideais devido à baixa resolução, problemas de iluminação da cena e oclusão. Por fim, os catálogos de supermercados são altamente dinâmicos e apresentam constante inserção e remoção de produtos, além de mudanças em aspectos visuais de itens preexistentes devido à campanhas de marketing e de reposicionamento de marca.

Portanto, este trabalho se propõe a investigar maneiras de gerar representações vetoriais de imagens usando redes neurais profundas de forma que essas representações possam ser usadas para identificação de produtos mesmo diante dos desafios mencionados. São realizados experimentos que avaliam o efeito que a arquitetura de rede, técnicas de aumento de dados e funções objetivo usadas no treinamento têm na qualidade das representações. A base de dados *Grocery Products* [10] foi utilizada nos experimentos. Ela contém imagens com qualidade de estúdio de milhares de produtos e, associadas a estas, imagens dos produtos em prateleiras de supermercados. Os métodos estudados são utilizados para codificar as imagens em *embeddings* e estes são utilizados para buscar as imagens de estúdio mais próximas das imagens reais. Por fim, são calculadas métricas que medem o percentual de vezes em que as imagens são do mesmo produto (acurácia top-1) e quantas vezes o produto buscado está no conjunto dos cinco imagens mais próximas (acurácia top-5).

2 TRABALHOS RELACIONADOS

Tonioni and Di Stefano [11] propuseram uma arquitetura que utiliza GAN (*Generative Adversarial Network*) para lidar com a mudança de domínios entre as imagens de treinamento, que possuem qualidade de estúdio e imagens de teste, que são obtidas por meio de câmeras de baixa qualidade. As imagens de produtos foram ajustadas para o domínio mais próximo da realidade ao passarem por uma GAN e posteriormente foram geradas representações vetoriais que impõem uma hierarquia entre categorias de produtos. Durante a realização da etapa de teste, foi utilizada busca através do algoritmo K-NN (*K-Nearest Neighbor*) para encontrar, a partir de sua representação vetorial (*embedding*), a qual classe um produto pertence. Foram utilizadas as bases de dados *Product8600* e *Grocery Products* [10], que possuem imagens de produtos de mercado.

As métricas de avaliação utilizadas foram a acurácia, para a busca de similaridade com 1-NN e 5-NN, em três cenários distintos, utilizando diferentes funções de perda. Os resultados foram que uma modificação hierárquica do *triplet ranking loss* é efetiva para aprender funções de *embedding*. A utilização de GANs foi uma abordagem efetiva considerando o cenário em que existem poucas imagens de treinamento. Foram obtidas acurácias de 84,2% (K=1) e 94,2% (K=5) na base de dados *Grocery Products* e acurácias de 82,7% (K=1) e 94,2% (K=5) na base de dados Product8600.

Filax et al. [12] propuseram um método para realizar a identificação de produtos de granulação fina em ambiente real, utilizando *metric learning*. Produtos de granulação fina são específicos e únicos. Um exemplo de identificação de produtos de granulação fina pode ser a identificação de uma caixa de leite desnatado da marca x. A identificação de produtos com maior granularidade, por outro lado, associa os produtos a macro classes.

No trabalho foi utilizado *deep metric learning* para realizar a identificação de produtos de mercado, onde as classes a serem preditas são desconhecidas durante a etapa de treino. Também foi proposta uma nova técnica chamada de Triple Mining, que utiliza todas as classes conhecidas durante o treinamento, preservando a capacidade de uso da validação cruzada K-Fold. Os experimentos foram realizados utilizando três bases de dados públicas: *Stanford Online Products*, *Magdeburg Groceries* e *Aliproducts*. Foi utilizada a métrica *mean Recall@k* com k =[1,2,4,8], utilizando a distância euclidiana para realizar a busca dos produtos. A abordagem proposta preservou a capacidade de distinguir produtos nunca antes vistos, enquanto que aumentou a precisão em até 5% nas bases de dados utilizadas.

Tonioni et al. [13] propuseram um *pipeline* com *deep learning* para realizar a identificação de produtos em prateleiras de lojas. Foi utilizado um detector de objetos em imagens para inicialmente extrair os produtos das imagens. As imagens extraídas alimentaram uma CNN (*Convolutional Neural Network*) com arquitetura VGG-16, que foi treinada como um descritor de características, transformando a imagem em um vetor. Para realizar a identificação por similaridade dos produtos, foi utilizado a técnica K-NN. Para os experimentos foi utilizada a base de dados *Grocery Products*, contendo para cada produto uma única imagem de referência. Para avaliar os experimentos foram utilizadas as métricas mAP (*mean Average Precision*), *mean recall* e mAMCA (*mean average multi-label classification accuracy*). Em um dos experimentos foi obtido mAP de 76,93% e resultados comparáveis ao estado da arte.

3 DESENVOLVIMENTO

3.1 Base de Dados

Neste trabalho, foi utilizada a base de dados *Grocery Products* [10] que contém 8.403 imagens de produtos de mercado como alimentos, produtos de beleza e limpeza. As imagens pertencem a dois domínios distintos: de imagens *in vitro* e *in situ*. As imagens *in vitro* são as imagens de qualidade de estúdio, ou seja, são imagens capturadas em condições próximas ao ideal. Já as imagens *in situ* são imagens capturadas em ambiente real, com equipamentos e condições que podem ser de baixa qualidade. Dentre os tipos de ruídos que podem estar presentes nas imagens *in situ* podemos citar má iluminação, reflexo, sombras, desfoque, distorções e oclusão. Nesse projeto foi

utilizado um subconjunto da base de dados, que contém imagens de 3.287 classes de produtos alimentícios (*in vitro*) armazenados em embalagens, como enlatados, caixas e garrafas. Esta subamostragem foi realizada porque o conjunto de imagens reais contém apenas produtos destas classes.

As imagens *in vitro* estão organizados de forma hierárquica (e.g., food > candy > chocolate > migros budget cciocolato al latte; ou food > coffee > delizio lungo fortissimo) e existe apenas uma única imagem para cada produto. As imagens *in situ* são obtidas a partir de imagens de prateleiras contendo vários produtos e, portanto, foi necessário realizar um pré-processamento para recortar as imagens dos produtos individuais das imagens das prateleiras. Para realizar essa atividade foram utilizadas "anotações" que indicavam o posicionamento de cada produto nas imagens das prateleiras. No total, foram extraídas 948 imagens de produtos *in situ*. A primeira coluna da Figura 3 ilustra imagens *in situ*, enquanto as demais colunas ilustram imagens *in vitro*.

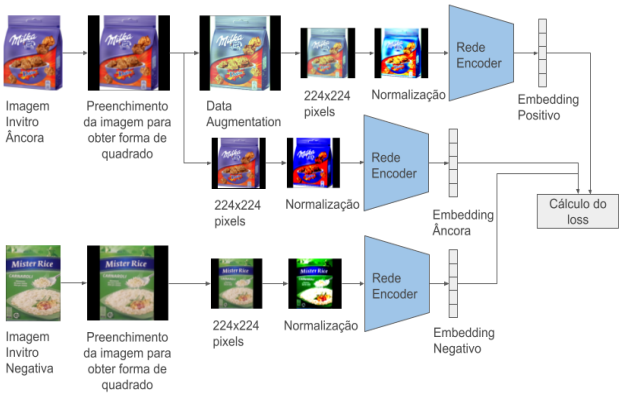

Figura 1: Diagrama de blocos descrevendo o sistema de reconhecimento de produtos proposto na etapa de treinamento.

3.2 Metodologia

A abordagem comumente utilizada nos últimos anos para a tarefa de classificação de imagens com redes neurais profundas é a utilização de uma rede neural convolucional para extração de características e camadas totalmente conectadas para transformação dos vetores de características e classificação. A última destas camadas gera como saída um vetor de probabilidades para cada classe.

Nestas arquiteturas, caso o número de classes aumente é preciso realizar um novo treinamento da rede. No contexto de identificação de produtos, cada produto corresponde a uma classe. Como frequentemente novos produtos são adicionados e removidos, a necessidade de retreino do modelo se torna indesejável. Outro ponto é que a medida que o número de classes aumenta o vetor de saída da rede se torna mais esparso e acaba ocupando mais memória sem necessidade e pode levar à instabilidade do processo de treinamento [14][15].

Neste projeto, busca-se construir uma solução que gere representações vetoriais (*embeddings*) de tamanho fixo ($N = 128$) para cada imagem que permitam identificar produtos mesmo que as condições de captura sejam diferentes nos conjuntos de treino e teste. Devido às limitações da base de dados, existem poucas imagens para cada um dos produtos, o que torna necessária a adoção de técnicas de regularização para prevenção de sobreajuste (*overfitting*) do modelo quando existe um conjunto de treinamento de tamanho limitado.

Para realizar a tarefa são realizados os seguintes passos: Utilização de modelo pré-treinado com as imagens da base de dados Imagenet [16]; Desbloqueio dos pesos de todas as camadas da rede, permitindo a realização de *fine-tunning* em uma nova base de dados; Remoção das camadas de classificação (presente nas últimas camadas da rede) e adição de uma camada de *max pooling* e uma camada linear com 128 neurônios; Normalização L2 da saída.

O *max pooling* foi aplicado para converter os *kernels* das camadas anteriores em representações unidimensionais, enquanto que uma camada linear foi adicionada para garantir que a saída tivesse tamanho igual a 128.

As imagens antes de alimentar o modelo, passam por pré-processamento tanto na etapa de treino quanto na etapa de teste. O primeiro pré-processamento torna a imagem quadrada, preenchendo os espaços com a cor preta. Algumas imagens podem ser alteradas utilizando técnicas de *data augmentation* como rotação, *blur*, *crops* e distorções. Esse processo é utilizado para gerar variações das imagens *in vitro*, aumentando a quantidade de imagens utilizadas durante o treinamento. Como as redes pré-treinadas foram treinadas utilizando a base de dados do Imagenet é preciso que as imagens tenham dimensão de $224x224$ pixels e sejam normalizadas utilizando valores específicos. As Figuras 1 e 2 mostram o momento que os pré-processamentos são utilizados.

As CNNs pré-treinadas correspondem a modelos que foram, na maioria dos casos, treinados com grandes bases de dados, em máquinas potentes, por horas ou dias. Portanto, muitos dos pesos das camadas da rede podem ser reaproveitados para outras tarefas, permitindo uma maior robustez e redução do tempo, pois não é preciso realizar o treinamento da rede neural profunda desde o início. O processo de aproveitamento de uma rede treinada para outras aplicações é chamado de *transfer learning*.

Como detalhado na seção 4.2, em alguns casos durante o treinamento da rede foi criado um *triplet* composto por uma imagem âncora, uma imagem positiva e uma imagem negativa. A imagem âncora é representada por uma imagem *in vitro*, a imagem positiva corresponde a imagem âncora após o processo de *data augmentation*. A imagem negativa corresponde a uma imagem *in vitro* de uma classe diferente da imagem âncora. Cada uma das três imagens alimentam o modelo que gera os respectivos *embeddings*, que são utilizados para o cálculo do erro, ou *loss*, do modelo (Figura 1).

Para realizar a classificação, todas as 3287 imagens *in vitro* alimentam o modelo e são convertidas em *embeddings* e cada uma delas está associada ao seu respectivo rótulo. As imagens de busca, ou seja, as imagens que precisam ser classificadas não possuem rótulo. Elas são representadas pelas 948 imagens *in situ*. Foi utilizado o algoritmo K-NN (*K-Nearest Neighbor*) para identificar os K *embeddings* mais próximos do *embedding* de busca, retornado K rótulos associados (Figura 2).

As métricas Top-1 e Top-5 foram utilizadas para a avaliação de desempenho dos modelos. A primeira mede o percentual de vezes que o *embedding* de treino mais próximo à um de teste corresponde

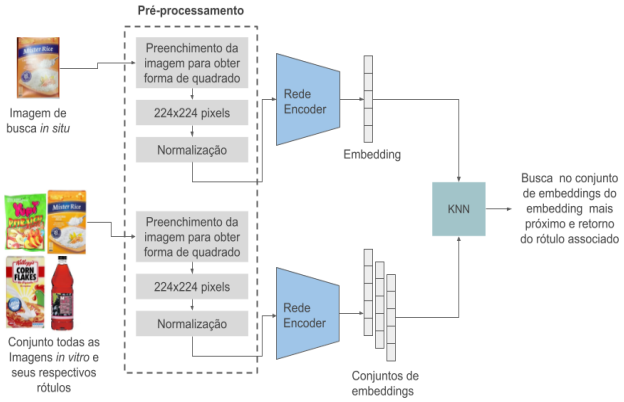

Figura 2: Diagrama de blocos descrevendo o sistema de reconhecimento de produtos proposto na etapa de teste.

ao mesmo produto. A segunda mede o percentual de vezes que o *embedding* relativo ao produto sendo buscado está entre os 5 mais próximos. Para exemplificar, suponha uma imagem de baixa qualidade de um pacote de café como imagem de busca (cujo rótulo é "café torrado"). Ao utilizar a métrica Top-5 são retornados os cinco rótulos que mais se aproximam do produto buscado, por exemplo, ["café torrado", "cappuccino", "café em grãos", "amendoim", "chá preto"]. Se o rótulo real ("café torrado") pertence ao conjunto, considera-se que houve um acerto, caso contrário, se ele não estiver no conjunto dos cinco rótulos computa-se um erro.

4 EXPERIMENTOS

Esta seção descreve a sequência de propostas avaliadas que levaram à arquitetura final do sistema. Ela discute ainda os resultados alcançados nas avaliações e ao final traz análises qualitativas acerca do desempenho dos modelos e das regiões nas imagens que mais influenciam os *embeddings* construídos. [1]

4.1 Seleção de uma CNN Pré-Treinada

O primeiro experimento teve objetivo selecionar uma rede neural convolucional (CNN) pré-treinada para servir como ponto de partida do processo. Para isto, foi selecionado um conjunto de modelos que possuem um bom desempenho preditivo na base de dados Imagenet [16], a saber, Resnet-50 [17], VGG-16 [18] e Densenet-201 [19]. Trabalhos anteriores indicam que existe correlação entre este valor e a performance em tarefas nas quais a rede será treinada posteriormente [20]. Além disso, foi avaliado o modelo Squeezenet que foi desenhado para utilizar poucos recursos computacionais mantendo um desempenho razoável [21].

As primeiras camadas de CNNs são tipicamente sensíveis a características mais simples das entradas como arestas e *blobs* de cor, enquanto neurônios das últimas camadas codificam estruturas de mais alto nível. Uma vez que produtos de supermercado são objetos diferentes daqueles encontrados na base Imagenet (e.g., navios, tigres e sinos), é possível que os códigos produzidos pelas últimas camadas dos modelos não sejam sensíveis à informações úteis para o reconhecimento de produtos. Para avaliar esta possibilidade, foi medido o desempenho alcançado usando diferentes camadas da rede VGG-16. A camada que levou aos melhores resultados foi a 4-3 e os valores das métricas para esta camada também são reportados.

[1]Todo o código desenvolvido está disponível em: https://github.com/glima91/cotb-2023-grocery-products-recognition

Tabela 1: Desempenho dos modelos pré-treinados no conjunto de teste.

Modelo	Top-1 (%)	Top-5 (%)
VGG-16	70,46	87,23
VGG-16 (4-3)	77,21	91,66
Densenet-201	77,21	93,56
SqueezeNet1.1	75,31	91,24
Resnet-50	67,93	89,66

Para este primeiro experimento, foi medida a performance dos modelos pré-treinados sem ajuste fino para a tarefa de interesse. Os modelos foram usados para criar *embeddings* para as imagens de treino e teste e foram calculadas as acurácias top-1 e top-5. A Tabela 1 traz os resultados do experimento.

As maiores acurácias top-1 foram alcançadas usando a última camada da Densenet-201 e a camada 4-3 da VGG-16, ambas com acurácia de 77,21%. A Squeezenet também alcançou bons resultados nesta métrica com acurácia top-1 de 75,31%. Analisando a acurácia top-5, vemos que embora a VGG-16 e a Densenet-201 tenham alcançado o mesmo valor de acurácia top-1, a Densenet-201 foi superior na acurácia top-5. O desempenho da VGG-16 foi equivalente à da Squeezenet nesta métrica.

Podemos concluir com este experimento que alguns modelos, a Densenet-201 e a VGG-16 (4-3), são capazes de produzir *embeddings* que levam a um bom desempenho mesmo sem treinamento específico na tarefa. É provável que o pré-treino utilizando a grande quantidade e variedade de imagens da base de dados Imagenet tenha feito os modelos sensíveis à características gerais o suficiente para permitir que eles sejam utilizados em uma tarefa tão distinta quando o reconhecimento de produtos. Uma surpresa foi o fato da Squeezenet alcançar desempenho similar mesmo possuindo uma quantidade substancialmente menor de parâmetros. Este modelo pode ser uma alternativa adequada em contextos com recursos computacionais limitados como a computação de borda (*edge computing*) em aplicações de Internet das Coisas (*Internet of Things* - IoT). O experimento indica ainda que, de fato, utilizar camadas superiores dos modelos pode levar à ganhos de desempenho em relação ao uso das últimas camadas.

Considerando os resultados dos experimentos iniciais, selecionamos a rede VGG-16 cortada na camada (4-3) para os próximos experimentos dado que ela alcançou desempenho tão bom quanto a Densenet-201 na acurácia top-1 e por possuir menos parâmetros. Nos próximos experimentos será realizado o ajuste fino do modelo na tarefa de interesse e como o conjunto de treinamento é composto por apenas uma imagem por produto, redes com menos parâmetros tendem a sofrer menos com superajuste.

4.2 Comparação de Métodos de Codificação

Neste experimento, o modelo VGG-16 (4-3) pré-treinado na base de dados Imagenet passou por um processo de ajuste fino para a tarefa de interesse utilizando diferentes métodos de aumento

de dados e funções objetivo. Este treinamento adicional almejava aumentar o desempenho do modelo utilizando dados da tarefa em que ele será aplicado. Foram avaliadas diversas combinações de aumentos de dados e funções objetivo. Seguindo a metodologia descrita na seção anterior, durante o treinamento as imagens âncora, positiva e negativa alimentam o modelo. Esse teve a sua camada de classificação substituída pelas camadas de *max pooling* e a camada linear. A saída é composta por *embeddings* de tamanho 128, que são utilizados para o cálculo da função objetivo e posterior atualização dos pesos da rede por *backpropagation*. Após o treinamento, os modelos são utilizados para produzir *embeddings* para os conjuntos de treino e teste e são calculadas as métricas top-1 e top-5.

4.2.1 Métodos de Aumento de Dados. Foram comparados quatro métodos de aumentos de dados: o RandAugment [22], o Augmix [23], duas combinações dos dois e um conjunto de operações escolhidas considerando as características da tarefa de reconhecimento de produtos. Iremos nos referir à esta última como *Custom Transform*. O RandAugment utiliza uma política de busca otimizada, que facilita o processo de seleção dos tipos de transformação aplicáveis nas imagens, já que existem várias combinações possíveis de transformações, com diversos hiperparâmetros. Para utilizar essa técnica, são considerados dois elementos: a quantidade de transformações aplicáveis sequencialmente e a sua magnitude. O Augmix é uma técnica que realiza diversas transformações de imagens em paralelo e no final combina todas as imagens transformadas em uma única imagem, sem apresentar grandes mudanças em relação à imagem original. Avaliamos a combinação entre essas duas técnicas pois enquanto uma aplica transformações relevantes sequencialmente a outra está focada em aumentar a robustez das imagens em relação aos ruídos provenientes da variação de domínio. As duas combinações dos métodos foram (i) a utilização do Augmix seguida do RandAugment (chamaremos esta estratégia de Augmix E RandAugment) e (ii) a seleção de um deles aleatoriamente, com igual probabilidade para cada amostra (chamaremos esta estratégia de Augmix OU RandAugment).

A Augmix e a RandAugment podem produzir imagens que não são verossímeis e, pior, podem tornar os modelos invariantes à propriedades indesejáveis. Considere, por exemplo, um modelo treinado para gerar a mesma resposta ao ver a embalagem do produto com diferentes cores. Na realidade, cores diferentes tipicamente indicam que, embora os itens sejam do mesmo tipo, eles possuem características diferentes como sabor em produtos alimentícios ou odor em produtos de limpeza. Na tarefa de reconhecimento de produtos, itens com sabores ou odores diferentes não são considerados o mesmo produto. Portanto, ao utilizar a Augmix e a RandAugment, os modelos poderiam perder a capacidade de diferenciar produtos pela cor. Vale reforçar que mudanças na cor são apenas algumas dentre várias possíveis transformações que não são compatíveis com o que é observado na realidade. Este fato justificou a construção do *Custom Transform*, um conjunto de transformações que eram comumente observadas ao avaliar visualmente imagens da base de dados.

As transformações utilizadas no *Custom Transform* foram:

- Cortar a imagem na metade horizontalmente ou verticalmente em 10% dos casos para simular oclusão ou situações em que o produto não está completamente contido na imagem.
- Também com 10% de chance, foi adicionado aos pixels ruído amostrado de uma distribuição Gaussiana com média zero e desvio padrão aleatório escolhido uniformemente entre 0 e 1. O valor do desvio padrão era escolhido uma vez por imagem e utilizado para amostrar os valores que seriam somados aos pixels. O objetivo desta operação foi simular o ruído de captura que acontece em ambientes com baixa iluminação como as imagens de produtos localizados no fundo das prateleiras.
- Em 10% dos casos, foi aplicado desfoque Gaussiano com intensidades aleatórias (sigma entre 0.1 e 1.2[2]). Esta transformação em conjunto com a próxima buscam simular diferentes níveis de nitidez nas imagens.
- Aumento da nitidez com fator de 10 e probabilidade de utilização de 50%. A transformação poderia tornar textos de mais informações de alta frequência mais evidentes nas imagens e incentivar o modelo a se tornar sensível à estas informações.
- Distorção de perspectiva com fator de 0.2 e probabilidade de aplicação de 10%. O objetivo desta transformação foi simular os casos em que o produto estava inclinado em relação à câmera ao capturar a imagem.
- Além das anteriores, foram aplicadas as transformações mais tradicionais de espelhamento vertical e rotação aleatória com ângulos escolhidos entre -20 e 20 graus, cada transformação com 50% de chance de aplicação.

4.2.2 Tarefas e Funções de Perda. A primeira tarefa utilizada para treinar os modelos foi o reconhecimento dos produtos do conjunto de treino usando técnicas de classificação de imagens. Cada produto foi representado por uma classe e a rede foi treinada utilizando a entropia cruzada categórica (*categorical cross-entropy* - CCE) [24]. Vale ressaltar que após o treinamento, a camada de classificação foi removida e rede foi utilizada para produzir *embeddings*.

A segunda tarefa foi a utilização de uma técnica de aprendizagem contrastiva (*contrastive learning*) [24] também utilizada para aprendizagem de métricas (*metric learning*) [24] e similar ao método SimCLRv2 [25]. Ela consiste em utilizar a CNN pré-treinada para produzir *embeddings* de três imagens, sendo duas do mesmo produto, uma original e outra produzida usando aumento de dados, e a terceira sendo uma imagem de outro produto. A rede é treinada usando o *triplet loss* [26, 27] para aproximar os *embeddings* que correspondem a imagens do mesmo produto e afastar o *embedding* relativo à imagem original do primeiro produto do *embedding* relativo à imagem do segundo produto. O *triplet loss* $L(A, P, N)$ é dado por:

$$L(A, P, N) = max(||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + \alpha, 0) \quad (1)$$

onde A, P e N são as imagens âncora (original do produto de referência), positiva (variação do produto de referência) e negativa (outro produto), respectivamente, f é a função de codificação que produz os *embeddings* (a CNN no nosso caso) e α é uma margem entre pares positivos e negativos. A operação $||.||^2$ é a distância euclidiana entre os *embeddings*.

[2] https://pytorch.org/vision/stable/generated/torchvision.transforms.GaussianBlur.html

O *triplet loss hard* $L_{hard}(A, P, N)$ é uma variação do cálculo do *triplet loss*, onde dados A, P e N, para um *batch* de imagens, considera-se a maior distância entre os *embeddings* de A em relação a P e a menor distância entre os *embeddings* de A em relação a N (Equação 2).

$$L_{hard}(A, P, N) = max(max(||f(A) - f(P)||^2) - min(||f(A) - f(N)||^2) + \alpha, 0) \quad (2)$$

As imagens da base de dados estão organizadas de forma hierárquica, isto é, produtos são organizados em categorias que por sua vez são agrupadas em categorias de níveis mais altos. A organização dos produtos pode ser utilizada em conjunto com o *triplet loss* para impor a restrição que *embeddings* de produtos da mesma categoria devem estar mais próximos no espaço que produtos de categorias diferentes, embora também tenham que manter uma certa distância por serem produtos diferentes. E mais, o afastamento entre as categorias deve ser maior quanto mais alta na hierarquia é a distinção entre elas. Para ilustrar a ideia, considere que os *embeddings* de dois tipos de arroz devem estar mais próximos que os *embeddings* de um arroz e um pacote de café. Por outro lado, os *embeddings* de um arroz e um café devem estar mais próximos entre si do que de um detergente ou perfume.

Para acomodar esta proposta, foi utilizada uma função de perda chamada *hierarchical triplet loss* (por concisão, nos resultados iremos nos referir ao método apenas como *hierarchical triplet*) [11] que é similar ao *triplet loss* tradicional, mas utiliza valores de margem diferentes dependendo da diferença na quantidade de hierarquia compartilhada entre os itens. Seja H uma função que retorna uma sequência de classes representando a hierarquia à qual um produto pertence (e.g., Alimentícios > Grãos > Arroz). Assumindo que os valores máximo e mínimo da margem são α_{max} e α_{min}, o valor de margem considerando a hierarquia $\alpha \in [\alpha_{min}, \alpha_{max}]$ é dada por:

$$\alpha = \alpha_{min} + \left(1 - \frac{|H(A) \cap H(N)|}{|H(A)|}\right).(\alpha_{max} - \alpha_{min}) \quad (3)$$

onde A e N são as imagens âncora e negativa e $|.|$ é o operador de cardinalidade para conjuntos. Se $H(A)$ e $H(N)$ compartilham todos as classes da hierarquia, $\alpha = \alpha_{min}$, enquanto que se as hierarquias forem completamente diferentes, teremos $\alpha = \alpha_{max}$ [11].

A última tarefa avaliada foi uma combinação dos anteriores. O modelo foi treinado na tarefa de classificação de imagens, mas o *hierarchical triplet loss* foi utilizado sobre a penúltima camada (a última antes da de classificação) como forma de regularizar o modelo. A função de perda utilizada no treinamento foi a soma da entropia cruzada categórica e o triplet loss (Equação 1) com α calculado de acordo com a Equação 3.

4.2.3 Resultados. Neste experimento, a rede VGG-16 (4-3) foi treinada utilizando combinações das técnicas de aumento de dados e das tarefas descritas acima. Após o treinamento, ela foi utilizada para codificar as imagens de treino e teste e os *embeddings* foram utilizados para calcular as métricas de avaliação.

Os valores das métricas obtidos são descritos na Tabela 2. A tabela está organizada em grupos de acordo com o tipo de método de aumento de dados utilizado. Cada linha da tabela informa os valores de top-1 e top-5 alcançados treinando o modelo com a função objetivo descrita na primeira coluna e usando a técnica de aumento de dados no título do grupo. Os resultados marcados em negrito são os melhores valores em cada métrica.

Tabela 2: Desempenho do modelo VGG-16 (camada 4-3) considerando treino em 20 épocas e *batches* de tamanho 32.

Função de Perda	Top-1 (%)	Top-5 (%)
RandAugment		
CCE	80,27	91,67
Triplet-Loss	78,16	89,98
Triplet-Loss-Hard	68,35	85,44
Hierarchical Triplet	76,48	91,03
CCE + Hierarchical Triplet	78,27	91,98
Augmix OU RandAugment		
CCE	80,17	91,14
Triplet-Loss	74,58	90,93
Triplet-Loss-Hard	67,93	85,65
Hierarchical Triplet	79,01	**92,83**
CCE + Hierarchical Triplet	**80,38**	92,62
Augmix E RandAugment		
CCE	77,95	91,14
Triplet-Loss	71,84	87,55
Triplet-Loss-Hard	58,44	75,74
Hierarchical Triplet	69,73	86,92
CCE + Hierarchical Triplet	72,99	88,50
Custom Transformation		
CCE	70,15	82,91
Triplet-Loss	59,28	79,43
Triplet-Loss-Hard	35,86	51,90
Hierarchical Triplet	67,30	84,92
CCE + Hierarchical Triplet	62,97	84,70

O treinamento usando a CCE em geral levou aos melhores resultados, contudo os melhor valor na métrica top-1 foi alcançado utilizando esta função de perda em conjunto com o regularizador baseado no *hierarchical triplet*, e com a estratégia Augmix OU RandAugment para realização de aumento de dados. Com esta configuração, o método foi capaz de reconhecer o produto correto em 80,38% dos casos e em 92,62% o produto estava entre os 5 mais próximos. O melhor desempenho na métrica top-5 foi alcançado usando a mesma estratégia de aumento de dados, mas treinando o modelo utilizando apenas o *hirarchical triplet*. Vale notar, contudo, que a diferença no valor de top-5 foi pequena em relação à configuração anterior que alcançou 92,62%. Vale notar ainda que usando apenas CCE e as estratégias de aumento RandAugment e Augmix OU RandAugment foi possível obter modelos com bom desempenho nas métricas.

Os resultados indicam que embora a estatégia de aumento *Custom Transformation* tenha sido desenhada considerando a tarefa de reconhecimento de produtos, ela leva à resultados inferiores em comparação com as técnicas RandAugment e Augmix. Outro resultado notável é o fato da utilização de técnicas para seleção de triplets difíceis em geral ter prejudicado o desempenho do método.

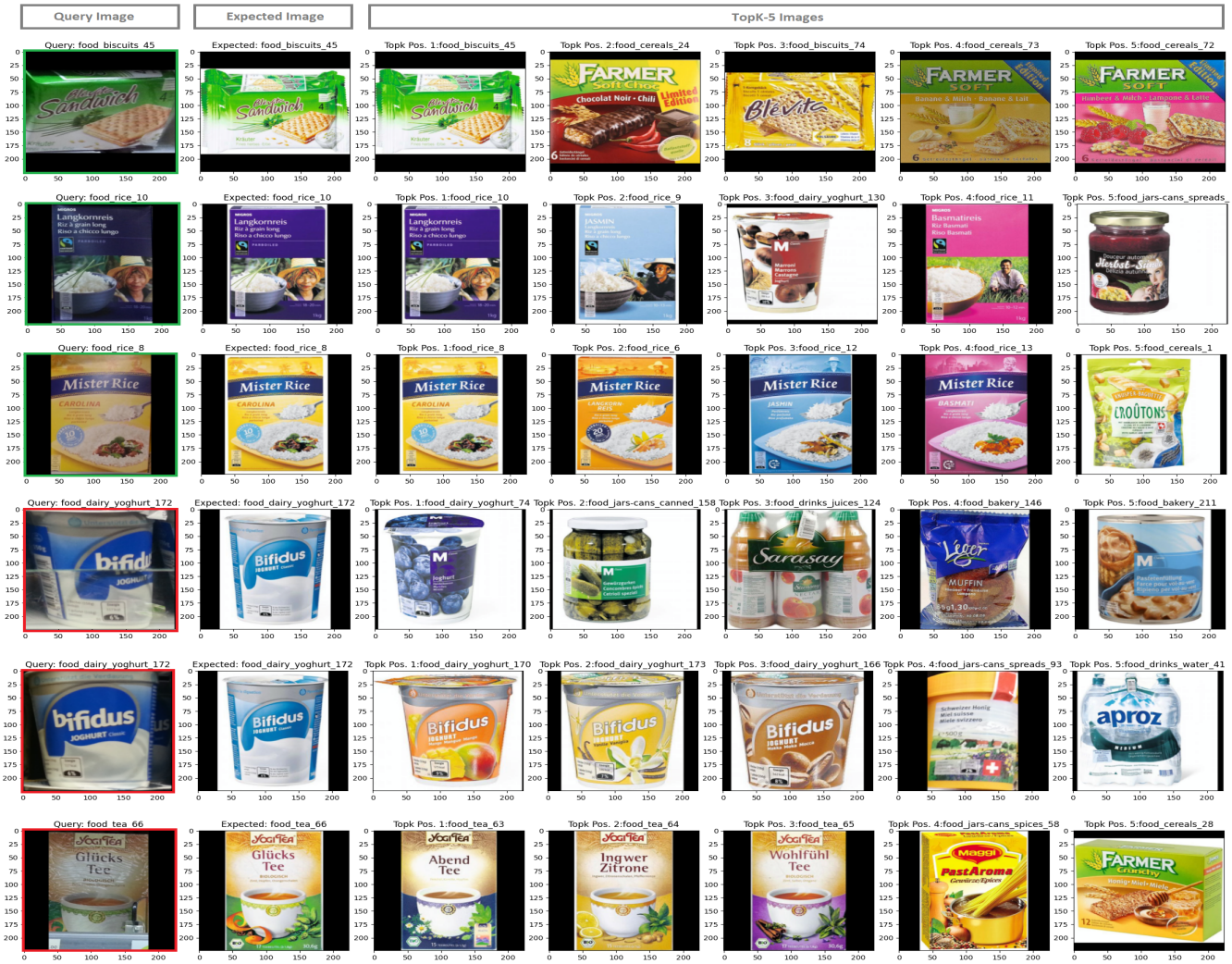

Figura 3: Casos de sucesso e falha do sistema proposto. Considerando o sentido da esquerda para a direita, as imagens na primeira coluna representam as consultas, a segunda coluna contém as imagens esperadas (*ground-truth*) e as imagens nas demais colunas representam as K imagens mais próximas. Nas consultas com borda verde, o produto pertence ao conjunto das 5 imagens mais próximas e nas marcadas em vermelho, não.

4.3 Análise Qualitativa do Modelo

Este experimento teve como objetivo analisar de forma qualitativa as imagens retornadas pelo sistema dado um conjunto de consultas, para avaliar se o modelo retorna imagens sensíveis e para identificar oportunidades de melhorias nos casos em que o sistema comete erros.

A Figura 3 ilustra casos de utilização do sistema. Cada linha representa uma consulta. A primeira coluna mostra as imagens que foram utilizadas como entrada, a segunda coluna mostra a imagem que o sistema deveria responder como a mais próxima (*ground truth*) e as colunas seguintes representam as 5 imagens mais próximas identificadas ordenadas de forma crescente pela distância. Nas linhas em que a imagem de entrada está marcada com borda verde são casos de acerto, i.e., casos em que a imagem mais próxima é a correta. Já as linhas marcadas com borda vermelha são casos de erro. Nestes casos, a imagem desejada não está entre as 5 mais próximas.

Como pode ser observado nas duas últimas linhas, mesmo quando o sistema erra as imagens retornadas são similares à entrada. Em geral, as imagens retornadas são do produto correto, mas com sabores diferentes. Em situações raras, como aquela ilustrada na quarta linha, o sistema comete erros catastróficos, no sentido de que os produtos retornados não têm similaridade alguma com o produto buscado. A comparação dos produtos retornados nas linhas 4 e 5 parece indicar que a rotação do produto na linha 4 pode ter influenciado negativamente o funcionamento do sistema. Este é um caso

difícil de ser tratado sem aumentar a base de dados com imagens dos produtos capturadas de diferentes ângulos.

Analisando as duas imagens mais próximas retornadas na segunda linha, é possível notar que elas contêm vasilhas de arroz e pessoas utilizando chapéus culturais. Estes resultados, e outros omitidos por restrições de espaço, mostram que a rede aprendeu a identificar características de alto nível nas imagens como ornamentos e representações visuais dos produtos. Existem indícios também de que a rede se tornou sensível aos textos existentes nas embalagens dos produtos.

5 CONCLUSÃO

Este trabalhou propôs e avaliou um sistema baseado em redes neurais artificiais para identificação de produtos do varejo à partir de fotos dos itens capturadas em ambiente real. O conjunto de treinamento eram imagens com qualidade de estúdio capturadas em condições ideais, i.e., com os produtos bem iluminados e em foco.

Foram comparadas diversas estratégias para a codificação das imagens usando redes neurais convolucionais de forma que os *embeddings* produzidos levassem à uma boa taxa de recuperação mesmo com as imagens apresentando características (iluminação, contraste, foco, orientação, etc.) diferentes daquelas encontradas no conjunto de treinamento. Resultados experimentais mostraram que a melhor arquitetura utilizou uma combinação das funções de perda *cross entropy loss* (CCE) e *hierarchical triplet loss*. A mistura de duas técnicas de aumento de dados (Randaugment ou Augmix), fez com que o modelo alcançasse acurácia de 80,38%, considerando a métrica top-1. Isto permitiu concluir ainda que a combinação de diferentes funções de perda e aumentos de dados pode proporcionar melhores resultados.

Devido às características da base de dados utilizada neste trabalho, apenas produtos alimentícios armazenados em embalagens foram considerados na avaliação. Estes produtos representam um subconjunto do total de produtos existentes em lojas do varejo. Em trabalhos futuros, o sistema proposto será avaliado em bases de dados mais variadas, contendo frutas e produtos de outras classes. Exemplos são as bases de dados Grocery Store Dataset [28] e Magdeburg Groceries Dataset [29].

AGRADECIMENTOS

Este estudo foi financiado em parte pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq, Brazil); e pela Fundação de Amparo à Pesquisa do Espírito Santo (FAPES, Brazil) - processo 2021-07KJ2; Os autores agradecem ainda ao apoio da FAPES e da CAPES (processo 2021-2S6CD, nºFAPES 132/2021) por meio do PDPG (Programa de Desenvolvimento da Pós-Graduação, Parcerias Estratégicas nos Estados).

REFERÊNCIAS

[1] Facts & Factors. At 39 % cagr, growing demand and trends in artificial intelligence (ai) in retail market share will hit usd 20.05 billion revenues by 2026, according to facts & factors, May 2021. URL https://www.globenewswire.com/news-release/2021/05/26/2236383/0/en/At-39-CAGR-Growing-Demand-and-Trends-in-Artificial-Intelligence-AI-in-Retail-Market\-Share-Will-Hit-USD-20-05-Billion-Revenues-by-2026-According-to-Facts-Factors.html.

[2] Chithrai Mani. Council post: Seven ways artificial intelligence is disrupting the retail industry, Aug 2020. URL https://www.forbes.com/sites/forbestechcouncil/2020/08/21/seven-ways-artificial-intelligence-is-disrupting-the-retail-industry/?sh=2601001256ae.

[3] Erik Brynjolfsson and Andrew McAfee. What's driving the machine learning explosion. *Harvard Business Review*, 18(3), 2017.

[4] Georges Mirza. The future of store automation post-covid-19, Sep 2020. URL https://risnews.com/future-store-automation-post-covid-19.

[5] Blake Ives, Kathy Cossick, and Dennis Adams. Amazon go: Disrupting retail? *Journal of Information Technology Teaching Cases*, 9(1):2–12, 2019.

[6] Yuchen Wei, Son Tran, Shuxiang Xu, Byeong Kang, and Matthew Springer. Deep learning for retail product recognition: Challenges and techniques. *Computational intelligence and neuroscience*, 2020, 2020.

[7] Xing Hao, Guigang Zhang, and Shang Ma. Deep learning. *International Journal of Semantic Computing*, 10(03):417–439, 2016.

[8] Chen Chao. On-device supermarket product recognition, Aug 2020. URL https://ai.googleblog.com/2020/07/on-device-supermarket-product.html.

[9] Xiaoxu Li, Xiaochen Yang, Zhanyu Ma, and Jing-Hao Xue. Deep metric learning for few-shot image classification: A selective review. *arXiv preprint arXiv:2105.08149*, 2021. URL https://arxiv.org/abs/2105.08149.

[10] Marian George and Christian Floerkemeier. Recognizing products: A per-exemplar multi-label image classification approach. In *European Conference on Computer Vision*, pages 440–455. Springer, 2014.

[11] Alessio Tonioni and Luigi Di Stefano. Domain invariant hierarchical embedding for grocery products recognition. *Computer Vision and Image Understanding*, 182:81–92, 2019.

[12] Marco Filax, Tim Gonschorek, and Frank Ortmeier. Grocery recognition in the wild: A new mining strategy for metric learning. In *VISIGRAPP (4: VISAPP)*, pages 498–505, 2021.

[13] Alessio Tonioni, Eugenio Serra, and Luigi Di Stefano. A deep learning pipeline for product recognition on store shelves. In *2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS)*, pages 25–31. IEEE, 2018.

[14] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey. *arXiv preprint arXiv:1901.09069*, 2019.

[15] Shirui Wang, Wenan Zhou, and Chao Jiang. A survey of word embeddings based on deep learning. *Computing*, 102(3):717–740, 2020.

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[18] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[20] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[21] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[22] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[23] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

[24] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[25] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

[26] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11(3), 2010.

[27] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[28] Marcus Klasson, Cheng Zhang, and Hedvig Kjellström. A hierarchical grocery store image dataset with visual and semantic labels. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2019.

[29] Marco Filax, Tim Gonschorek, and Frank Ortmeier. Data for image recognition tasks: An efficient tool for fine-grained annotations. In *ICPRAM*, pages 900–907, 2019.