Análise Comparativa de Redes Neurais Convolucionais no Reconhecimento de Cenas

Victor Souza
Faculdade de Computação e Engenharia Elétrica
Universidade Federal do Sul e Sudeste do Pará
Marabá, Pará
victoor.soouza@unifesspa.edu.com

Luan Silva
Faculdade de Computação e Engenharia Elétrica
Universidade Federal do Sul e Sudeste do Pará
Marabá, Pará
luansilvatec@unifesspa.edu.br

Leandro Araújo
Faculdade de Computação e Engenharia Elétrica
Universidade Federal do Sul e Sudeste do Pará
Marabá, Pará
leandronetlink@unifesspa.edu.br

Adam Santos
Faculdade de Computação e Engenharia Elétrica
Universidade Federal do Sul e Sudeste do Pará
Marabá, Pará
adamdreyton@unifesspa.com

ABSTRACT

This paper aims to compare the convolutional neural networks (CNNs): ResNet50, InceptionV3, and InceptionResNetV2 tested with and without pre-trained weights on the *ImageNet* database in order to solve the scene recognition problem. The results showed that the pre-trained ResNet50 achieved the best performance with an average accuracy of 99.82% in training and 85.53% in the test, while the worst result was attributed to the ResNet50 without pre-training, with 88.76% and 71.66% of average accuracy in training and testing, respectively. The main contribution of this work is the direct comparison between the CNNs widely applied in the literature, that is, to enable a better selection of the algorithms in the various scene recognition applications.

KEYWORDS

Redes neurais convolucionais, Reconhecimento de cenas, Aprendizado de máquina

1 INTRODUÇÃO

O reconhecimento de cena, o processo de categorizar imagens em diferentes classes (por exemplo, costa, rodovia, rua, quarto e loja), é um problema desafiador, importante no campo de visão computacional [1]. No âmbito do reconhecimento de cenas existem diversas classificações como as macro-classes: interior e exterior de ambientes, cenas urbanas e naturais. Elas podem ser subdivididas em classes como montanha, quarto, loja, mar, prédios, entre outras.

Reconhecimento de cena é uma tarefa complexa na visão computacional, pois aborda o problema de onde determinado objeto se encontra, como num quarto, numa cozinha ou numa rua. Tal tarefa é amplamente utilizada em muitos aspectos, por exemplo, no planejamento de caminhos robóticos, análise de conteúdo de vídeo, recuperação de imagens baseada em conteúdo e vigilância por vídeo [2].

A rede neural artificial (RNA) é uma técnica bio-inspirada que após a apresentação de um conjunto de dados e um processo de treinamento é capaz de aprender e se adaptar para a realização de diversas tarefas. Com a possibilidade de aprendizagem e adaptação os modelos de redes neurais podem lidar com dados imprecisos e situações que não foram totalmente definidas no início do treinamento. Tais características tornam esses modelos cada vez mais atrativos na utilização em técnicas de processamento de imagem, reconhecimento de padrão, problemas de classificação, controle de processos, etc. [3].

As redes neurais convolucionais (RNCs) são utilizadas no processamento de imagens, vídeos, voz e áudio [3]. As RNCs vêm sendo cada vez mais utilizadas na classificação de imagens: imagens médicas, placas de trânsito, objetos e cenas. O Desafio de Reconhecimento Visual de Grande Escala do *ImageNet* (*ILSVRC*) tem auxiliado a constante evolução das RNCs no que tange a avaliação de diversas arquiteturas utilizando-se de uma extensa base de dados com diversas categorias de objetos.

Neste trabalho são exploradas as RNCs ResNet50, InceptionV3 e InceptionResNetV2. A utilização desses modelos tem como objetivo identificar a melhor alternativa para a classificação de cenas, verificar o comportamento da combinação da arquitetura Inception com a ResNet através da InceptionResNetV2 e facilitar a seleção de arquiteturas para possíveis aplicações. Os modelos foram avaliados a partir de várias simulações com e sem pesos pré-treinados. Para a validação dos resultados obtidos foram utilizados os valores médios das métricas: acurácia, precisão, revocação e pontuação - F1. Nas simulações foram utilizadas 10 classes da base de dados *Places* [4].

Este trabalho está organizado da seguinte maneira. Na seção 2 serão abordadas mais informações sobre o problema de classificação de cenas. Na seção 3, uma breve explicação das RNCs deste estudo é apresentada. Na seção 4, a metodologia utilizada, ferramentas, base de dados e arquiteturas são discutidos. Na seção 5, os resultados obtidos com os modelos testados são apresentados. Por fim, na seção 6, as considerações finais deste trabalho são sumarizadas.

2 PROBLEMA

Cena pode ser definida como o local em que uma pessoa pode interagir ou se locomover [5]. O reconhecimento de cenas traz mais desafios se comparado a tarefa de reconhecimento de objetos, por exemplo, a variabilidade no conteúdo das fotografias de cenas e variações de intensidade e escala [2].

As cenas, e suas funções associadas, estão intimamente relacionadas com as características visuais que estruturam o espaço. A função dos ambientes pode ser definida pela sua forma e tamanho

(um corredor estreito é para caminhar, uma arena expansiva é para eventos públicos), pelos seus materiais constituintes (neve, grama, água, madeira), ou por objetos embutidos (mesa e cadeiras, jóias, equipamentos de laboratório) [5].

Em Zhou [4] é descrito o banco de dados *Places*, sendo avaliadas várias RNCs tais como: ResNet, AlexNet, GoogLeNet e VGG. Essas redes foram testadas utilizando duas versões da base de dados *Places*: *Places*205 e *Places*365. Além dos testes somente com o *Places*, também foram verificadas outras bases de dados, tais como SUN397, Indoor67, Scene15, Caltech101, Caltech256, Action40 e Event8, mas com pesos pré-treinados da *ImageNet*, *Places*205 e *Places*365. Com os testes realizados, foi possível identificar um bom desempenho da VGG com a acurácia *top*-1 no teste de 55,24% e na validação 55,19% utilizando 365 classes da base *Places*.

Em [6] é abordado o problema de classificação de cenas internas, no qual são comparadas diferentes formas de extração de características para posterior classificação. Para a extração de características das imagens, foram utilizados as seguintes técnicas: *Scale Invariant Feature Transform* (SIFT),*Speeded-Up Robust Feature* (SURF) e *Tamura features*. Além das técnicas citadas, o autor propõe uma técnica para a extração de características, a qual consiste na junção do SIFT, SURF e *Tamura features*.

Para a classificação das cenas foi utilizado *Support Vector Machine* (SVM), técnica de aprendizado supervisionado utilizada para classificação e regressão. A base de dados utilizada foi a MIT-Indoor [7]. Para a avaliação dos resultados foram utilizadas as métricas: acurácia, precisão, revocação e pontuação-f1. Como resultado, o autor obteve uma acurácia de 66%, 62%, 54% e 70% com as técnicas SIFT, SURF, TAMURA e a proposta pelo autor , respectivamente.

Em Xiao [5] foi desenvolvida a primeira grande base de dados para o reconhecimento de cenas com 899 classes (SUN *database*), também foi verificada com qual acurácia que os humanos podem classificar c enas e m c entenas d e c lasses, o btendo-se e m média 58,60%. Verificou-se também a acurácia de diversos algoritmos na classificação de cenas e a possibilidade de detectar cenas inseridas em cenas maiores.

3 REDES NEURAIS CONVOLUCIONAIS

As RNCs são arquiteturas biologicamente inspiradas capazes de serem treinadas e aprenderem representações invariantes à escala, translação, rotação e transformações afins [8, 9].

Uma RNC é composta pelas seguintes camadas: camada convolucional, camada de agrupamento e camada totalmente conectada. As camadas convolucionais usam filtros sobre a imagem para obter uma série de mapas de características, sendo um para cada filtro. As camadas de agrupamento são responsáveis por reduzir a amostra para esses mapas de características e, por fim, a tarefa de classificação das imagens é realizada pela camada totalmente conectada [10]. A principal camada dessas redes é a convolucional. A sua função é aplicar máscaras nas imagens de entrada, com base em uma vizinhança de *pixels*. Com isso é possível obter filtros de convolução (matrizes) que armazenam os pesos das conexões entre os neurônios [11, 12]. O compartilhamento de pesos na camada de convolução garante que os filtros sejam aplicados em diferentes posições na imagem, diminuindo significativamente o número de dados a serem aprendidos [13].

Outra camada que também é importante nas RNCs é a camada de agrupamento. Ela é responsável por reduzir a dimensionalidade dos mapas de características, diminuindo largura e altura. A operação de agrupamento possibilita uma invariância espacial. O agrupamento de características, na maioria das arquiteturas de convolução, utilizam as funções de *Max-pooling* e *Avarage-Pooling*, as quais são capazes de determinar respectivamente o valor máximo e médio de agrupamento em uma vizinhança [11, 14].

As próximas camadas das RNCs desempenham o papel de regressão das ativações. Em qualquer rede desse tipo, após a camada de agrupamento, é necessário ao menos uma camada totalmente conectada. Elas servem para criar caminhos de decisões a partir dos filtros obtidos na camada anterior [15]. A última camada das redes convolucionais também é totalmente conectada. Essa, por sua vez, é responsável por realizar a classificação dos dados. Nessa situação, uma função determina a identificação das saídas em classes. A função mais utilizada é a *Softmax* (para problemas de múltiplas classes) ou *Sigmoid* (problemas binários) [12]. O treinamento das RNCs é realizado na maioria dos casos com o *backpropagation*, que ajusta os pesos w dos neurônios pelo erro mensurado entre a verdade e a predição da rede, utilizando os componentes do vetor gradiente.

3.1 ResNet50

A ResNet (Rede Residual) é uma rede convolucional clássica usada como *backbone* para muitas tarefas de visão computacional. Esse modelo foi o vencedor do desafio *ImageNet* em 2015. O avanço fundamental com a ResNet é permite treinar redes neurais extremamente profundas com mais de 150 camadas. Antes do treinamento da ResNet, redes neurais muito profundas eram difíceis devido ao problema dos gradientes de desaparecimento, após uma dada quantidade de camadas o incremento de mais camadas não conseguia aumentar a qualidade do modelo [16].

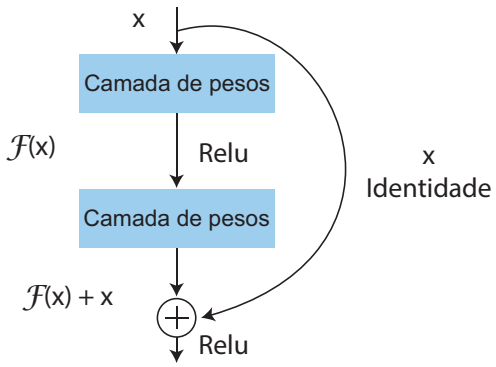

Figura 1: Bloco Residual da ResNet (Adaptada [16]).

Essa arquitetura trabalha na proposta em que as camadas continuem a receber os valores resultantes das função de ativação *Rectified Linear Unit* (ReLu - Uma função simples não linear em que se o valor de entrada é negativo retorna zero e caso seja positivo ou zero mantém o valor de entrada) - *F(x)*, da camada anterior, mas também recebam os valores de entrada x dessas funções. Como pode ser observado na Figura 1.

Essa arquitetura possui duas características principais:

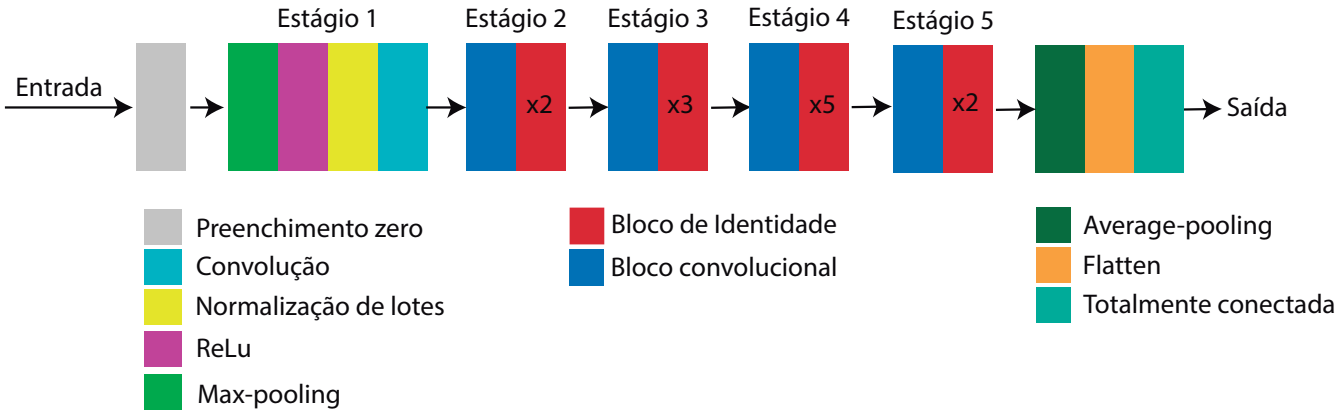

Figura 2: Arquitetura da ResNet (Adaptada [17]).

- "Conexões de atalho de identidade": uma estratégia de "atalhos" ou "conexões de salto", que pulam pares de grupos de camadas convolucionais. São também chamadas unidades bloqueadas ou unidades recorrentes fechadas, apesar de não apresentarem uma recorrência no sentido tradicional dos modelos de redes neurais recorrentes;
- Grande foco em normalização de lotes, o que possibilita tornar diferentes entradas de um modelo de aprendizado de máquina mais semelhantes entre si, para que o modelo possa aprender e generalizar bem novos dados [18].

De forma simplificada, a ideia dos atalhos em ResNets é evitar que a rede, muito profunda, "morra" por esvanecimento de gradientes através do empilhamento de mapeamentos de identidades que, do ponto de vista matemático, segundo os autores [16], estão simplesmente empilhando camadas que não fazem nada. Com isso, como mostra a Figura 1, a ResNet utiliza, num determinado ponto, um sinal que é a soma do sinal produzido pelas duas camadas convolucionais anteriores somado ao sinal transmitido diretamente do ponto anterior a estas camadas, juntando um sinal processado com um sinal de uma etapa anterior no processamento [16].

A arquitetura básica de uma ResNet é descrita na Figura 2. É aplicada na entrada um bloco de preenchimento zero (adição de linhas e colunas com o valor zero em cada lado do filtro de convolução [18]) de (3,3). No Estágio 1, a Convolução 2D tem 64 filtros da forma (7,7) e usa uma *stride* de (2,2). A normalização de lotes é aplicado ao eixo dos canais da entrada. O *max-pooling* usa uma matriz (3,3) e uma *stride* (2,2). No Estágio 2, 3, 4 e 5 o bloco convolucional usa três conjuntos de filtros. No estágio 2 usa 2 blocos de identidade que utilizam três conjuntos de filtros. No estágio 3 os 3 blocos de identidade usam três conjuntos de filtros. No estágio 4, os 5 blocos de identidade usam três conjuntos de filtros. Estágio 5, os 2 blocos de identidade usam três conjuntos de filtros. O *average-pooling* usa uma matriz de (2,2). O *flatten* (transforma o mapa de características para que os dados possam ser utilizados na camada totalmente conectada) não possui nenhum hiperparâmetro. Por fim, a camada totalmente conectada (densa) reduz sua entrada para o número de classes usando uma ativação *Softmax*[16, 17] .

3.2 InceptionV3

A rede GoogLeNet foi vencedora do ILSVRC no ano de 2014 [19]. Sua principal contribuição foi o desenvolvimento de um módulo Inception que reduziu drasticamente o número de parâmetros na rede para 4 milhões, comparada a rede AlexNet com 60 milhões. O objetivo principal do módulo Inception é atuar como um extrator de características em vários níveis computando convoluções 1x1, 3x3 e 5x5 dentro do mesmo módulo da rede, conforme evidencia a Figura 3.

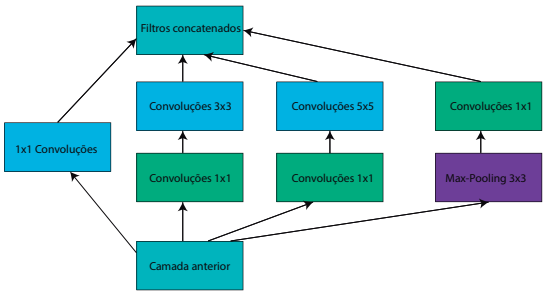

Figura 3: Módulo Inception (Adaptada [20]).

O InceptionV3 é uma arquitetura voltada para a resolução de classificação de imagens que foi treinada no conjunto de dados *ImageNet*. Essa rede apresentou bom desempenho com relativamente baixo custo computacional no ano de 2015, pois ela reduziu os parâmetros que são estimados pela rede, fazendo com que ela possua um melhor desempenho computacional em relação as redes VGG, durante seu treinamento [19, 21].

3.3 Inception-ResNetV2

A Inception-ResNet trata-se basicamente da substituição da concatenação de filtros utilizadas na Inception por conexões residuais, realizando assim a combinação das duas arquiteturas ResNet e Inception.

Como as redes Inception tendem a ser muito profundas, é natural substituir o estágio de concatenação do filtro da arquitetura Inception por conexões residuais. Isso permite a Inception colher

todos os benefícios da abordagem residual, mantendo sua eficiência computacional [22].

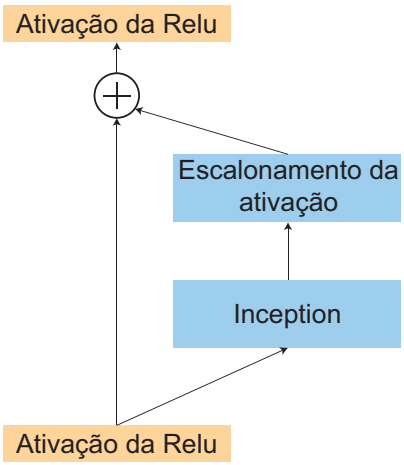

Figura 4: Módulo Inception-ResNet (Adaptada [22]).

Na Figura 4 é possível observar o módulo utilizado na Inception-ResNet-v2. Essa versão da Inception é mais custosa, com desempenho de reconhecimento significativamente aprimorado.

4 METODOLOGIA

A linguagem de programação Python foi utilizada para a implementação das RNCs, com o auxílio da biblioteca Keras, amplamente utilizada na construção de redes neurais profundas, executando sobre a biblioteca de código aberto TensorFlow.

Keras é uma biblioteca que fornece blocos de construção para o desenvolvimento de RNCs. Os blocos de construção que o Keras fornece são construídos usando Theano e TensorFlow [23].

O TensorFlow é um sistema de aprendizado de máquina que opera em grande escala e em diversos ambientes. Faz um mapa de nós com o fluxo de dados em várias máquinas e em uma máquina faz a partir de seus componentes: CPUs, GPUs e TPUs [24].

O treinamento das arquiteturas propostas foi executado em uma máquina virtual (VM) fornecida pelo Google Colaboratory (Colab). O Colab tem o objetivo disseminar e incentivar a aprendizagem e a pesquisa em aprendizagem de máquina. O Colab disponibiliza VMs pré-configurados com as bibliotecas essenciais de aprendizagem de máquina e inteligência artificial, como TensorFlow, Matplotlib e Keras. A VM após um período de execução é desativada, e todos os dados e configurações do usuário são perdidos. [25].

4.1 Arquitetura

Neste trabalho foram utilizadas as arquiteturas ResNet50, InceptionV3 e InceptionResNetV2 com e sem pesos pré-treinados na *ImageNet*, todas essas arquiteturas são implementadas pelo Keras, no qual é possível utilizar os pesos da rede treinada com *ImageNet*.

A utilização de pesos pré-treinados nos modelos selecionados trata-se de uma técnica de aprendizado de máquina conhecida como transferência de aprendizagem. Tal técnica possui o objetivo de melhorar a aprendizagem da tarefa atual, aproveitando o conhecimento adquirido em outra tarefa [26].

No trabalho em questão foi utilizado o conhecimento adquirido do treinamento com a base de dados *ImageNet* com o auxílio da técnica de ajuste fino da transferência de aprendizagem, o qual irá refinar as camadas para que se possa classificar uma nova base de dados. A base de dados *ImageNet* é um conjunto de mais de 15 milhões de imagens rotuladas, pertencentes, a aproximadamente, 22.000 categorias. As imagens foram coletadas da web e rotuladas por humanos com auxílio da ferramenta de *crowdsourcing Mechanical Turk* da Amazon [27].

4.2 Hiperparâmetros utilizados

Todas as arquiteturas foram testadas no Google Colaboratory. Para a realização dos experimentos foram adotados alguns hiperparâmetros que se mantiveram iguais em todas as RNCs.

Por se tratar de um problema que envolve a classificação de múltiplas classes, foi selecionada a *Softmax* como função da camada de saída. Para a função de custo foi utilizada a *Categorical Crossentropy*, uma das principais funções utilizadas na literatura para obter o valor de custo em problemas com múltiplas classes.

Por meio de observações e experiências anteriores, foi possível definir alguns dos parâmetros como a quantidade de épocas definida como 200, o que permitiu que as redes em seus valores finais obtivessem um baixo desvio padrão, e o *dropout* em 0,5 a fim de evitar *overfitting*. Para o tamanho do lote foi definido o valor de 128 e na taxa de aprendizagem foi realizada a seguinte estratégia: valor inicial de $1x10^{-3}$, sendo multiplicada por taxas menores no decorrer das épocas, isto é: a partir da época 81, $1x10^{-1}$; época 121, $1x10^{-2}$; época 161, $1x10^{-3}$; e época 181: $0{,}5x10^{-3}$.

4.3 Matriz de Confusão

A matriz de confusão é comumente utilizada em aprendizado de máquina, possuindo informações sobre as classificações reais e previstas realizadas por um classificador [28]. Em uma matriz de confusão as linhas são valores reais em cada classe, enquanto as colunas são as predições realizadas pelo modelo.

A partir da matriz de confusão é possível obter o número de classificações corretas e previstas pelo classificador em cada classe, no conjunto de imagens utilizadas para teste [29].

Quando as imagens da classe em análise são classificadas corretamente pelo modelo, elas são verdadeiros positivos (VP), enquanto aquelas que são classificadas de forma errada são falsos positivos (FP). Imagens que não pertencem a classe em análise, mas foram identificadas como pertencentes são falsos negativos (FN) e imagens que não foram previstas pertencentes a classe e que de fato não pertencem são verdadeiros negativos (VN).

A partir da matriz de confusão, é possível calcular as métricas para avaliar se o algoritmo está ou não conseguindo bons resultados.

4.4 Base de dados

Para comparação dos algoritmos foi utilizada a base de dados *Places*. O banco de dados *Places* é um repositório quase exaustivo de 10 milhões de fotografias de cena, rotuladas com 434 categorias semânticas de cena, compreendendo cerca de 98% dos tipos de lugares que um humano pode encontrar no mundo [4].

A utilização de RNCs compreende bilhões de operações, isto é, requer grande custo computacional. Devido a limitações de *hardware*,

Figura 5: Exemplos de imagens das classes selecionadas.

este trabalho limitou-se a seleção de 10 classes, sendo que para cada uma delas foram selecionadas, de forma aleatória, 2500 imagens. A fim de diminuir o custo computacional, todas as imagens foram redimensionadas para 75x75 *pixels*. As imagens foram divididas em 70% para treino e 30% para teste.

Foram selecionadas as seguintes classes:

(1) Fachada de construções;
(2) Faixa de pedestres;
(3) Deserto (areia);
(4) Campo de cultivo;
(5) Floresta (Folhas largas);
(6) Geleira;
(7) Rodovia;
(8) Aterro;
(9) Montanha;
(10) Oceano.

No decorrer deste artigo, as classes serão evocadas de acordo com o número de cada classe dado pela lista acima.

A base de dados utilizada está disponível na versão *Standard* e *Challenge*. Para este trabalho foi utilizada a *Places*365 *Standard*. Na Figura 5 é possível observar algumas imagens das classes selecionadas para os testes.

4.5 Métricas de avaliação dos resultados

Os valores obtidos a partir da matriz de confusão são utilizados para gerar algumas métricas de suma importância para a avaliação dos modelos, tais como: Acurácia, Precisão, Revocação e pontuação-F1. Todas essas métricas são comumente utilizadas na avaliação de modelos de aprendizagem de máquina.

4.5.1 Acurácia. É a quantidade total de classificações corretas (VP e VN) pelo número total de classicações [28], tal que

$$Acurácia = \frac{VP + VN}{VP + FP + FN + VN}. \quad (1)$$

4.5.2 Precisão. É calculada como a exatidão das classificações. É a razão entre o número de exemplos corretamente classificados como VP e o número total de imagens da classe em análise (VP e FP) [30]:

$$precisão = \frac{VP}{VP + FP}. \quad (2)$$

4.5.3 Revocação. É calculada como a integridade da classificação. É a razão entre o número total de exemplos classificados como VP e as imagens totais que são de fato da classe em análise (VP e FN) [30]:

$$revocação = \frac{VP}{VP + FN}. \quad (3)$$

4.5.4 Pontuação-F1. É a média harmônica entre a precisão e a revocação [30]:

$$pontuação\text{-}F1 = \frac{2 \times precisão \times revocação}{precisão + revocação}. \quad (4)$$

Com o objetivo de gerar maior segurança nos resultados obtidos, foi realizada a média aritmética simples das métricas acima, a partir de um total de 10 simulações, sendo que em cada simulação se manteve os mesmos dados no treino e teste, para cada modelo testado.

5 RESULTADOS

Nas Figuras 6 e 7, é possível observar o comportamento da acurácia no treino e no teste durante 200 épocas. A partir dessas figuras,

é possível identificar um melhor resultado das arquiteturas que utilizaram os pesos da *ImageNet*. Dentre elas a que obteve o melhor resultado foi a ResNet50, com uma acurácia média de 99,82% no treino e 85,53% no teste. As arquiteturas InceptionV3 e InceptionResNetV2, ambas com *ImageNet*, apresentaram resultados bem semelhantes, sendo que a InceptionV3 obteve 99,52% no treino e 83,43% no teste e a InceptionResNetV2 obteve 99,44% no treino e 83,47% no teste.

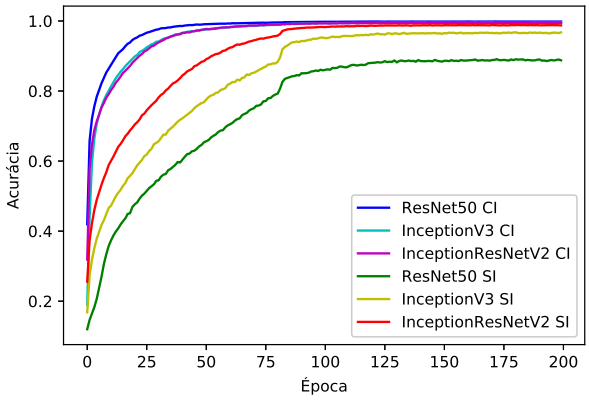

Figura 6: Acurácia média do treinamento em 10 simulações (Com *ImageNet* - CI, Sem *ImageNet* - SI).

Somente a partir dos modelos sem *ImageNet* foi possível observar uma maior diferença entre as arquiteturas, se contrapondo aos resultados dos modelos com *ImageNet*, a ResNet50 obteve o pior resultado 88,76% no treino e 71,66% no teste. O melhor resultado obtido, sem os pesos da *ImageNet*, foi o da InceptionResNetV2 obtendo 98,74% no treino e 76,80% no teste. Em relação ao teste, a Inception, obteve um resultado semelhante ao da ResNet50, mas conseguiu obter um melhor resultado no treino, sendo 96,75% no treino e 72,60% no teste.

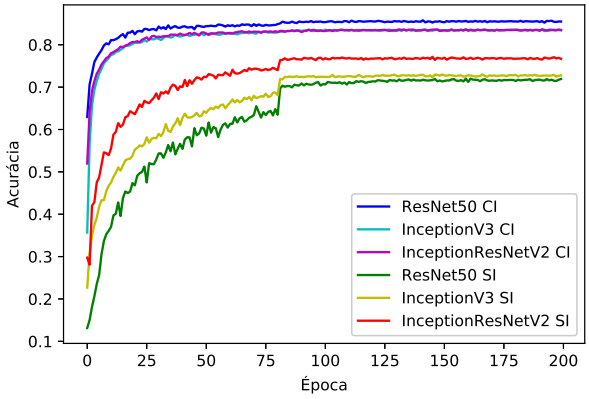

Figura 7: Acurácia média do teste em 10 simulações (Com *ImageNet* - CI, Sem *ImageNet* - SI).

Nas Figuras 8, 9 e 10 são apresentadas as matrizes de confusão, respectivamente, dos modelos ResNet50 com *ImageNet*, ResNet50 sem *ImageNet* e InceptionResNetV2 sem *ImageNet*, nos quais nas linhas observam-se as classes preditas e, nas colunas, as classes que de fato pertence. A partir da matriz de confusão, é possível observar com mais facilidade algumas informações, como a quantidade de vezes que um modelo classifica uma classe nas demais classes. De modo geral, um ponto interessante de analisar são que as classes 6 e 9 possuem características semelhantes. A partir, disso a maior quantidade de erros da classe 6 foram predições na classe 9 e a maior quantidade de erros da classe 9 foram predições na classe 6.

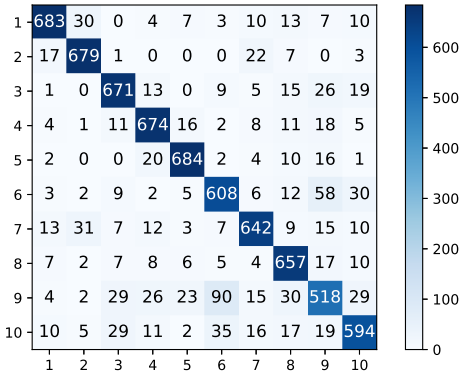

Figura 8: Média da matriz de confusão da ResNet50 com *ImageNet* em 10 simulações.

Na Figura 8, é possível visualizar com clareza a dificuldade na classificação da classe 9, que obteve o verdadeiro positivo somente de 518, visto que as demais classes conseguiram valores maiores que 600. Os falsos positivos da classe 9 foram de 248, sendo um valor alto em relação as demais classes. As classes com menor complexidade na identificação foram as classes 1 e 5, com verdadeiro positivo de 683 e 684, respectivamente, em relação ao falso positivo obteve-se 84 e 55.

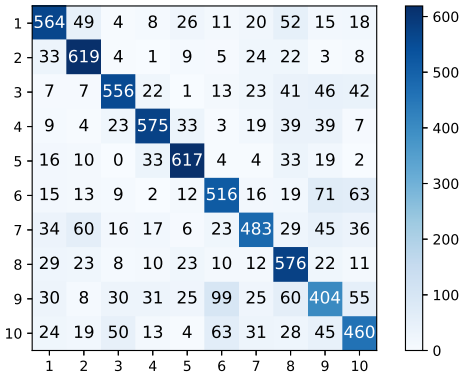

Figura 9: Média da matriz de confusão da ResNet50 sem *ImageNet* em 10 simulações.

Na Figura 9 é apresentada a matriz de confusão do modelo que apresentou pior desempenho em relação aos demais modelos. Nesse modelo somente as classes 2 e 5 conseguiram obter o valor de verdadeiro positivo superior a 600. A classe 9 se manteve como a mais complexa para a classificação com o verdadeiro positivo de 404 e falso positivo de 363.

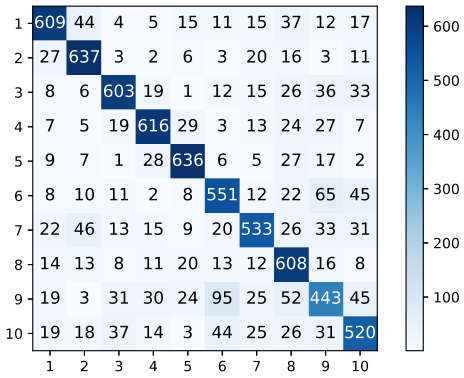

Figura 10: Média da matriz de confusão da InceptionResNet sem *ImageNet* em 10 simulações.

A Figura 10 apresenta a matriz de confusão do melhor modelo sem *ImageNet*, apresentando quatro classes em que o valor do verdadeiro positivo é inferior a 600, a classe 9 se manteve com o menor valor do verdadeiro positivo.

A partir das matrizes de confusão geradas, obteve-se as métricas precisão, revocação e pontuação-F1, que são apresentadas nas Tabelas 1, 2 e 3, respectivamente. A partir dessas métricas é possível identificar as classes 1, 2 e 5 como as mais fáceis para a identificação com os valores da pontuação-F1 em média de 89,65% e desvio padrão de 1,38% para os modelos com *ImageNet*. Para modelos sem *ImageNet*, as classes com melhor pontuação-F1 foram as classes 2, 4 e 5 obtendo 81,48% na média e 2,24% de desvio padrão, um possível motivo seria o fato das características dessas cenas serem mais distintas das demais.

Tabela 1: Média das métricas precisão, revocação e pontuação-F1 da ResNet50 em 10 simulações.

	ResNet50 CI | SI		
Classe	precisão	revocação	pontuação-F1
1	91,55% | 73,79%	88,50% | 73,08%	90,00% | 73,43%
2	89,78% | 75,90%	92,71% | 84,54%	91,22% | 79,98%
3	87,32% | 78,92%	87,95% | 72,85%	87,63% | 75,76%
4	87,04% | 80,36%	89,32% | 76,21%	88,16% | 78,23%
5	91,14% | 81,13%	92,10% | 83,16%	91,62% | 82,13%
6	79,41% | 68,62%	82,38% | 69,89%	80,86% | 69,24%
7	87,30% | 73,25%	85,29% | 64,16%	86,28% | 68,40%
8	83,79% | 63,82%	90,29% | 79,15%	86,91% | 70,65%
9	74,30% | 56,67%	67,25% | 52,49%	70,59% | 54,49%
10	83,05% | 65,27%	80,15% | 62,05%	81,57% | 63,61%

Já as classes em que existiram maiores dificuldades para identificação foram a 6, 9 e 10, as quaisapresentaram um valor médio na *pontuação-F1* de 76,27% e desvio padrão de 5,87% para os modelos com *ImageNet*. Para modelos sem *ImageNet*, obteve-se 64,88% na média e desvio padrão de 6,59%, sendo que uma possível justificativa seria o fato das características geométricas serem bem semelhantes, por exemplo, montanhas congeladas podem ser bem semelhantes as geleiras.

Tabela 2: Média das métricas precisão, revocação e pontuação - F1 da InceptionV3 em 10 simulações.

	InceptionV3 CI | SI		
Classe	precisão	revocação	pontuação-F1
1	88,29% | 76,84%	86,27% | 74,08%	87,26% | 75,43%
2	87,31% | 77,16%	91,61% | 83,51%	89,40% | 80,21%
3	86,83% | 78,12%	84,74% | 76,18%	85,76% | 77,13%
4	85,36% | 79,73%	87,50% | 77,11%	86,41% | 78,39%
5	89,46% | 79,76%	90,86% | 82,87%	90,15% | 81,28%
6	77,93% | 68,42%	80,72% | 70,81%	79,28% | 69,59%
7	85,75% | 72,63%	81,69% | 66,45%	83,66% | 69,38%
8	81,36% | 67,85%	88,26% | 78,46%	84,66% | 72,76%
9	70,31% | 59,46%	65,08% | 53,99%	67,59% | 56,58%
10	81,28% | 66,36%	78,27% | 64,19%	79,74% | 65,25%

Para as demais classes com *ImageNet* obteve-se 85,80% em média e 1,50% de desvio padrão em relação a pontuação-F1. Para os modelos sem *ImageNet*, obteve-se 74,53% em média e 3,86% de desvio padrão.

Tabela 3: Média das métricas precisão, revocação e pontuação-F1 da InceptionResNetV2 em 10 simulações.

	InceptionResnetV2 CI | SI		
Classe	precisão	revocação	pontuação - F1 e
1	89,08% | 81,74%	87,36% | 78,89%	88,21% | 80,28%
2	87,55% | 80,26%	90,37% | 86,97%	88,93% | 83,48%
3	86,13% | 82,10%	84,18% | 78,97%	85,13% | 80,49%
4	85,76% | 82,68%	87,81% | 81,68%	86,77% | 82,17%
5	90,32% | 84,23%	89,97% | 85,69%	90,14% | 84,95%
6	78,83% | 72,37%	78,62% | 74,63%	78,72% | 73,47%
7	84,16% | 78,38%	82,55% | 70,85%	83,33% | 74,42%
8	81,55% | 70,06%	88,57% | 83,54%	84,91% | 76,20%
9	70,73% | 64,51%	66,16% | 57,46%	68,35% | 60,77%
10	80,25% | 71,84%	79,76% | 70,19%	79,99% | 71,00%

6 CONCLUSÃO

Em geral, os modelos que já iniciaram com os pesos da *ImageNet* obtiveram um desempenho superior. Em um cenário com a utilização de pesos pré-treinados na *ImageNet*, o uso da ResNet50 se mostrou mais interessante, mas em situações em que não é possível utilizar a *ImageNet*, a InceptionResNetV2 obteve melhores resultados.

Outro ponto que vale destacar são as classes 6 e 9 que, em geral, obtiveram os piores resultados. Um possível motivo da baixa acurácia da classe 9 seria a presença de características dela nas demais

classes. A classe que teve mais predições erradas em relação a classe 9 foi a classe 6, que possui características geométricas semelhantes aos da classe 9. Portanto imagens em que aparecem simultaneamente geleiras e montanhas são, a princípio, indistinguíveis.

Como fora observado, classes com certo nível de similaridade apresentam maior dificuldade para a classificação das cenas, gerando certo grau de confusão para o modelo, como: mar, costa, rio e lagos.

A utilização de pesos pré-treinados em modelos gera grande vantagem comparada a modelos que iniciam do zero, sendo a *ImageNet* uma base específica para a identificação de objetos. A utilização de pesos, vindo de base de dados com foco em cenas, deve trazer uma maior acurácia para os modelos em estudo futuros.

REFERÊNCIAS

[1] Y. Yuan, L. Mou, and X. Lu. Scene recognition by manifold regularized deep learning architecture. *IEEE Transactions on Neural Networks and Learning Systems*, 26(10):2222–2233, Oct 2015. doi: 10.1109/TNNLS.2014.2359471.

[2] Xianglin Meng, Zhengzhi Wang, and Lizhen Wu. Building global image features for scene recognition. *Pattern Recognition*, 45(1):373 – 380, 2012. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2011.06.012. URL http://www.sciencedirect.com/science/article/pii/S0031320311002834.

[3] Eduarda Almeida Leão Marques. Estudo sobre redes neurais de aprendizado profundo com aplicações em classificação de imagens. Monografia, Departamento de Estatística, Universidade de Brasília, Brasília, Brasil, 2016. URL http://bdm.unb.br/handle/10483/15147.

[4] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, June 2018. doi: 10.1109/TPAMI.2017.2723009.

[5] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3485–3492, June 2010. doi: 10.1109/CVPR.2010.5539970.

[6] J. S. Gill and A. S. Brar. Support vector based indoor scene classification technique using different features. In *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, pages 685–689, June 2019. doi: 10.1109/ICECA.2019.8822153.

[7] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–420, June 2009. doi: 10.1109/CVPR.2009.5206537.

[8] T. Liu, C. Rosenberg, and H. A. Rowley. Clustering billions of images with large scale nearest neighbor search. In *2007 IEEE Workshop on Applications of Computer Vision (WACV '07)*, pages 28–28, Feb 2007. doi: 10.1109/WACV.2007.18.

[9] Guilherme Defreitas Juraszek. Reconhecimento de produtos por imagem utilizando palavras visuais e redes neurais convolucionais. Dissertação, Centro de Ciências Tecnológicas, Universidade do Estado de Santa Catarina, Joinville, Brasil, 2014. URL http://tede.udesc.br/tede/tede/1761.

[10] Wafa Mousser and Salima Ouadfel. Deep feature extraction for pap-smear image classification: A comparative study. In *Proceedings of the 2019 5th International Conference on Computer and Technology Applications*, ICCTA 2019, pages 6–10, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-7181-0. doi: 10.1145/3323933.3324060. URL http://doi.acm.org/10.1145/3323933.3324060.

[11] Alan Santos, Kelson Aires, Rodrigo Veras, Valeska Uchôa, and Luís Santos. Uma abordagem de classificação de imagens dermatoscópicas utilizando aprendizado profundo com redes neurais convolucionais. In *Anais do XVII Workshop de Informática Médica*, Porto Alegre, RS, Brasil, 2017. SBC. URL https://sol.sbc.org.br/index.php/sbcas/article/view/3717.

[12] Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In Mark W. Jones Xianghua Xie and Gary K. L. Tam, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 41.1–41.12. BMVA Press, September 2015. ISBN 1-901725-53-7. doi: 10.5244/C.29.41. URL https://dx.doi.org/10.5244/C.29.41.

[13] J. Kawahara, A. BenTaieb, and G. Hamarneh. Deep features to classify skin lesions. In *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, pages 1397–1400, April 2016. doi: 10.1109/ISBI.2016.7493528.

[14] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv e-prints*, art. arXiv:1409.1556, Sep 2014.

[15] Karel Vedaldi, Andrea e Lenc. Matconvnet: redes neurais convolucionais para o matlab. In ACM, editor, *Anais da 23ª Conferência Internacional da ACM sobre Multimídia*, MM '15, pages 689–692, Nova York, NY, EUA. ISBN 978-1-4503-3459-4. doi: 10.1145/2733373.2807412. URL http://doi.acm.org/10.1145/2733373.2807412.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

[17] Muhammad Rizwan. Residual networks (resnets), oct 2018. URL https://engmrk.com/residual-networks-resnets/. Acessado: 07-02-2020.

[18] Francois Chollet. *Deep Learning with Python*. Manning Publications Co., Greenwich, CT, USA, 1st edition, 2017. ISBN 1617294438, 9781617294433.

[19] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015. URL http://arxiv.org/abs/1512.00567.

[20] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, June 2015. doi: 10.1109/CVPR.2015.7298594.

[21] Rodrigo Emerson Valentim da SILVA. Um estudo comparativo entre redes neurais convolucionais para a classificação de imagens. Monografia, Graduação em Sistemas de Informação, Universidade Federal do Ceará, Quixadá, Brasil, 2018. URL http://www.repositorio.ufc.br/handle/riufc/39475.

[22] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261, 2016. URL http://arxiv.org/abs/1602.07261.

[23] Nikhil Ketkar. *Introduction to Keras*, pages 97–111. Apress, Berkeley, CA, 2017. ISBN 978-1-4842-2766-4. doi: 10.1007/978-1-4842-2766-4_7. URL https://doi.org/10.1007/978-1-4842-2766-4_7.

[24] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zhang. Tensorflow: A system for large-scale machine learning. *CoRR*, abs/1605.08695, 2016. URL http://arxiv.org/abs/1605.08695.

[25] T. Carneiro, R. V. Medeiros Da Nóbrega, T. Nepomuceno, G. Bian, V. H. C. De Albuquerque, and P. P. R. Filho. Performance analysis of google colaboratory as a tool for accelerating deep learning applications. *IEEE Access*, 6:61677–61685, 2018. doi: 10.1109/ACCESS.2018.2874767.

[26] Emilio Soria Olivas, Jose David Martin Guerrero, Marcelino Martinez Sober, Jose Rafael Magdalena Benedito, and Antonio Jose Serrano Lopez. *Handbook Of Research On Machine Learning Applications and Trends: Algorithms, Methods and Techniques - 2 Volumes*. Information Science Reference - Imprint of: IGI Publishing, Hershey, PA, 2009. ISBN 1605667668, 9781605667669.

[27] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[28] Xinyang Deng, Qi Liu, Yong Deng, and Sankaran Mahadevan. An improved method to construct basic probability assignment based on the confusion matrix for classification problem. *Information Sciences*, 340-341:250 – 261, 2016. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2016.01.033. URL http://www.sciencedirect.com/science/article/pii/S002002551600044X.

[29] C. Silva, D. Welfer, F. P. Gioda, and C. Dornelles. Cattle brand recognition using convolutional neural network and support vector machines. *IEEE Latin America Transactions*, 15(2):310–316, Feb 2017. doi: 10.1109/TLA.2017.7854627.

[30] Abinash Tripathy, Ankit Agrawal, and Santanu Kumar Rath. Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications*, 57:117 – 126, 2016. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2016.03.028. URL http://www.sciencedirect.com/science/article/pii/S095741741630118X.