Algoritmos de Classificação de Imagens

Heitor Hellou Marcondes e Othon Valentim

A classificação de imagens é o processo de categorizar e rotular grupos de pixels ou vetores em uma imagem com base em regras específicas. O reconhecimento de categoria visual é um problema extremamente desafiador. Neste artigo, apresentamos três abordagens diferentes para resolver esse problema usando técnicas de deep learning.

Keywords—classificação, imagen, deep learning

I. INTRODUÇÃO

A classificação de imagens é uma tarefa fundamental em visão computacional que envolve a atribuição de rótulos ou categorias a imagens com base em seu conteúdo visual. As Redes Neurais Convolucionais (CNNs) surgiram como uma poderosa técnica de aprendizado profundo para teste de classificação de imagens. As CNNs são usadas para extrair progressivamente representações de nível superior e superior do conteúdo da imagem. Em vez de pré-processar a imagem para derivar recursos como texturas e formas, uma CNN usa apenas os dados brutos de pixel da imagem como entrada e "aprende" como extrair esses recursos e, finalmente, inferir qual objeto eles constituem. As CNNs aproveitam o conceito de convolução, que envolve a aplicação de filtros para extrair características significativas das imagens. Esses filtros capturam padrões em diferentes níveis de abstração, aprendendo gradualmente as representações hierárquicas dos dados de entrada. Sua capacidade de aprender automaticamente recursos discriminativos a partir de dados de pixel brutos os torna adequados para o problema de classificação de imagens.

II. O PROBLEMA

A. *Selecionando o Dataset*

O dataset selecionado para esse artigo é composto por uma coleção abrangente de imagens com uma grande variedade de frutas. Neste caso bagas. O conjunto de dados abrange 10 tipos diferentes de bagas, fornecendo um recurso diversificado para treinar e avaliar modelos de classificação de imagens. O conjunto de dados contém imagens coloridas de alta qualidade (RGB) capturadas de vários ângulos e sob diferentes condições de iluminação. Cada categoria de bagas inclui 200 imagens para teste, 200 imagens para treinamento e 200 imagens para validação.

B. *Preprocessando as Imagens*

The Todas as imagens antes de serem alimentadas a CNN passam por um processo de pré-processamento, onde todas tem seu tamanho definido por 224 x 224 e seus pixels normalizados Essa padronização permite que o modelo processe imagens com um formato de entrada fixo, facilitando o processo de aprendizado e também atenuando a influência da variação de iluminação e distribuição de cores nas diferentes imagens.

III. AS ARQUITETURAS

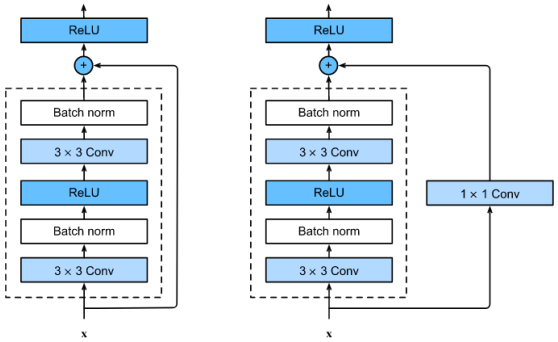
Para este artigo, apresentamos três diferentes arquiteturas de redes neurais convolucionais. A primeira é de um modelo pré-existente chamado ResNet. A segunda arquitetura é a implementação de outro modelo pré-existente, o DenseNet. E terceiro uma arquitetura desenvolvida inteiramente por nós com o intuito de analisar e comparar os resultados de cada algoritmo.

A. *ResNet*

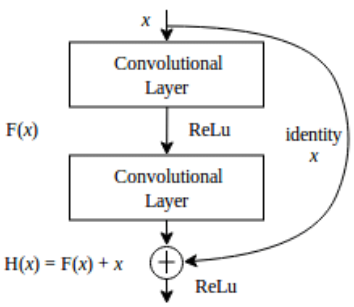
ResNet, ou Rede Neural Residual, é um tipo de arquitetura de rede neural profunda que se destaca na área de visão computacional, em particular no campo de reconhecimento de imagens. Ela foi proposta pela primeira vez em 2015 por Kaiming He, Xiangyu Zhang, Shaoqing Ren e Jian Sun.

O principal desafio enfrentado no treinamento de redes neurais profundas é o problema de desvanecimento do gradiente, onde os gradientes utilizados para atualizar os pesos da rede diminuem gradualmente à medida que são propagados de volta nas camadas anteriores. Isso pode dificultar o treinamento de redes mais profundas, pois as camadas anteriores recebem atualizações de peso muito pequenas, resultando em um aprendizado lento ou estagnação.

A arquitetura ResNet aborda esse problema introduzindo conexões residuais, também conhecidas como atalhos, que permitem que as informações fluam diretamente das camadas anteriores para as camadas posteriores sem passar por transformações lineares. Essas conexões residuais ajudam a aliviar o desvanecimento do gradiente, permitindo que as camadas anteriores recebam gradientes mais fortes durante o treinamento.



A ideia central por trás das conexões residuais é que, se uma camada puder mapear diretamente sua entrada para a saída sem fazer muitas transformações, ela pode aprender a mapear a diferença entre a entrada e a saída (ou seja, o resíduo) em vez de tentar aprender toda a transformação complexa. Essa abordagem simplifica o aprendizado, permitindo que a rede aprenda as características residuais ou diferenças em vez de ter que aprender todas as características do zero.

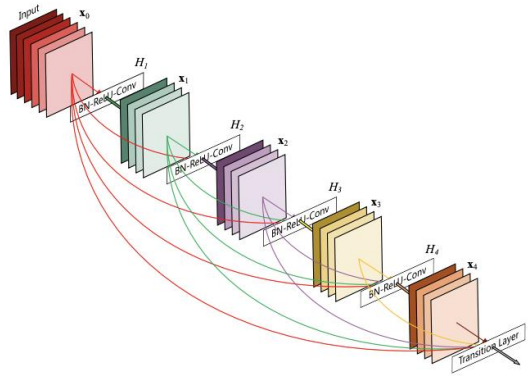


As ResNets geralmente têm uma arquitetura profunda, com dezenas, centenas ou até mesmo milhares de camadas. Devido à presença das conexões residuais, o treinamento dessas redes é facilitado, permitindo que elas sejam mais profundas e obtenham melhores desempenhos em tarefas de visão computacional, como classificação de imagens, detecção de objetos e segmentação semântica.are unavoidable.

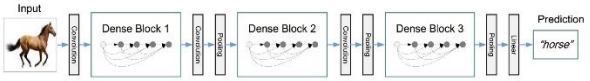
B. DenseNet

DenseNet, ou Rede Neural Dense, é uma arquitetura de rede neural profunda desenvolvida por Gao Huang, Zhuang Liu, Laurens van der Maaten e Kilian Q. Weinberger em 2016. Ela foi projetada para abordar problemas de fluxo de informações e uso eficiente de parâmetros em redes neurais convolucionais.

Ao contrário das arquiteturas convencionais, como as redes ResNet, onde as camadas são conectadas sequencialmente, as DenseNets introduzem conexões densas entre todas as camadas, formando um padrão de conectividade densa. Isso significa que cada camada tem conexões diretas com todas as camadas subsequentes, permitindo que as informações fluam mais facilmente por toda a rede.



Em uma DenseNet, cada camada recebe como entrada os recursos concatenados de todas as camadas anteriores. Dessa forma, todas as camadas contribuem diretamente para as camadas subsequentes, o que promove o reuso de recursos e facilita a propagação de gradientes durante o treinamento. Essa arquitetura densamente conectada traz vantagens significativas, como melhorar o fluxo de informações e aumentar a estabilidade do treinamento.



As DenseNets são usadas principalmente em tarefas de visão computacional, como classificação de imagens, detecção de objetos e segmentação semântica. Elas têm obtido resultados impressionantes em termos de precisão e eficiência, superando várias arquiteturas anteriores. Além disso, as DenseNets foram projetadas para serem altamente escaláveis, permitindo a construção de redes ainda mais profundas, mantendo o fluxo de informações e o treinamento estáveis.in:

C. Arquitetura Autoral

Ao projetar nossa rede neural convolucional, uma das primeiras perguntas que surgiram foi quantas camadas especificar na rede. Começamos com uma arquitetura simples composta por sete camadas, das quais duas são camadas convolucionais, duas são camadas de max pooling e três são camadas totalmente conectadas.

Começamos com nossa primeira camada convolucional tendo 32 filtros e um tamanho de kernel de 3 x 3. Nossa segunda camada convolucional era composta por 64 filtros. Nossa primeira camada densa tinha 64 unidades e a última tinha 10 unidades.

Após nossos testes iniciais, percebemos que isso não era suficiente e começamos a mexer no número de camadas e em suas configurações, juntamente com as especificações dos hiper parâmetros.

O modelo apresentado nesta seção é uma arquitetura de rede neural convolucional (CNN) projetada para tarefas de classificação de imagens. O modelo consiste em várias camadas, incluindo camadas convolucionais, camadas de max pooling e camadas totalmente conectadas.

A entrada do modelo espera ser imagens com uma forma de (224, 224, 3), representando imagens RGB com tamanho de 224x224 pixels. A primeira camada do modelo é uma camada Conv2D com 32 filtros de tamanho (3, 3), passo (1, 1) e padding 'same'. Ela aplica a função de ativação linear retificada (ReLU) para introduzir não-linearidade.

A próxima camada é uma camada MaxPooling2D com tamanho de agrupamento (2, 2) e padding 'same'. Essa camada realiza a operação de max pooling, reduzindo as dimensões espaciais dos mapas de características enquanto preserva as características importantes.

Em seguida, outra camada Conv2D é adicionada com 64 filtros de tamanho (3, 3), passo (2, 2) e padding 'same'. É seguida por uma camada MaxPooling2D com tamanho de agrupamento (4, 4) e padding 'same', diminuindo ainda mais as dimensões dos mapas de características.

O modelo continua com outra camada Conv2D com 128 filtros de tamanho (3, 3), passo (2, 2) e padding 'same'. Essa camada é seguida por uma camada MaxPooling2D com tamanho de agrupamento (4, 4) e padding 'valid'. Essa configuração reduz as dimensões espaciais dos mapas de características, descartando algumas das informações das bordas.

Após as camadas de pooling, é adicionada uma camada Flatten para converter os mapas de características 2D em um vetor 1D, preparando os dados para as camadas totalmente conectadas.

As camadas subsequentes são camadas Dense, que são camadas totalmente conectadas. A primeira camada Dense possui 500 unidades e usa a função de ativação ReLU. Uma camada Dropout com taxa de 0,4 é aplicada após essa camada para regularizar o modelo e evitar overfitting.

Em seguida, uma camada Dense com 250 unidades e ativação ReLU é adicionada, seguida por outra camada Dropout com taxa de 0,3. Essas camadas contribuem ainda mais para a regularização do modelo e ajudam a melhorar a generalização.

Por fim, a última camada Dense consiste em 10 unidades, representando o número de classes na tarefa de classificação, e aplica a função de ativação softmax para produzir pontuações de probabilidade para cada classe.

Nossa arquitetura personalizada foi compilada usando o otimizador Adam com uma taxa de aprendizado de 0,001. Empregamos a função Sparse Categorical Crossentropy loss, configurada com o parâmetro "from_logits" em True. O desempenho do modelo foi avaliado por meio das métricas de precisão.

Durante o processo de treinamento, o modelo ajustou iterativamente seus pesos e vieses para minimizar a função de perda definida e melhorar sua precisão nos dados de treinamento. Simultaneamente, os dados de validação foram usados para avaliar o desempenho de generalização do modelo e detectar quaisquer sinais de overfitting ou underfitting.

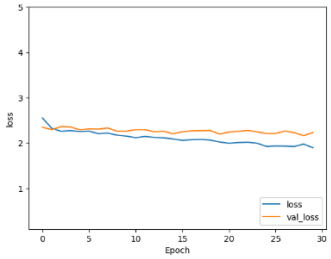
IV. RESULTADOS

Nesta seção, apresentamos os resultados de nosso estudo comparativo em três arquiteturas diferentes para classificação de imagens: ResNet, DenseNet e a arquitetura personalizada descrita na seção anterior. Conduzimos experimentos extensivos em um conjunto de dados diversificado que consiste em imagens de frutas para avaliar o desempenho dessas arquiteturas.

Para começar, treinamos cada arquitetura usando os mesmos conjuntos de dados de treinamento e teste, seguindo uma configuração experimental padronizada. O conjunto de dados de treinamento compreendia 150 imagens por classe, enquanto o conjunto de dados de teste consistia em 50 imagens por classe. Todas as imagens foram redimensionadas para um tamanho uniforme de 224x224 pixels e pré-processadas de acordo com as práticas padrão da área.

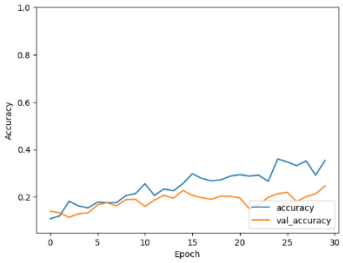
A. *ResNet*

Durante o treinamento da rede ResNet, a perda obtida foi de 3,0247. A perda é uma métrica que indica o quão bem o modelo está ajustando-se aos dados de treinamento. Uma perda mais baixa sugere que o modelo está se aproximando dos rótulos corretos das imagens de treinamento. Nesse sentido, o valor de perda obtido indica que a rede ResNet está minimizando a discrepância entre as previsões e os rótulos durante o treinamento, o que é um bom indicativo de um processo de aprendizado eficaz. Vale ressaltar que o valor de perda obtido sozinho não indica a presença de overfitting.



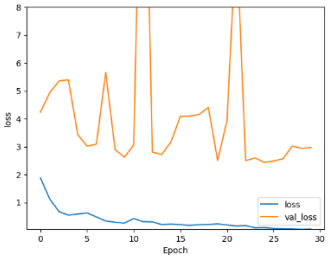
Acurácia alcançada pela rede ResNet foi de 0,1. A acurácia é uma medida comumente utilizada para avaliar o desempenho de modelos de classificação e representa a proporção de exemplos corretamente classificados em relação ao total de exemplos. Nesse contexto, uma acurácia de 0,1 sugere que a rede ResNet obteve um desempenho relativamente baixo na tarefa de classificação.

O resultado obtido se baseou em testes de 10 a 70 épocas, que resultaram em valores semelhantes de acurácia, normalmente variando entre 10% e 20%.

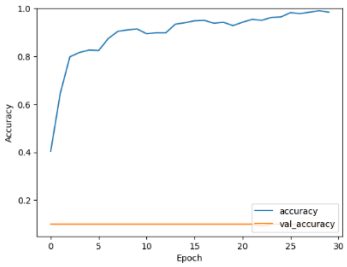


B. *DenseNet*

Durante o treinamento da rede DenseNet, a perda obtida foi de 2.9706. O valor de perda obtido sugere que a rede está fazendo progresso ao minimizar a diferença entre suas previsões e os rótulos desejados, o que indica um bom desempenho do modelo. Vale ressaltar, que como ocorrido na rede ResNet, o valor da perda por si só não conclui a tese do aparecimento de overfitting durante o treinamento.



A acurácia verificada foi de 0,1. Portanto, como na rede ResNet apresentado anteriormente, a acurácia obtida indica um baixo desempenho da rede para desempenhar classificação de imagens.



O resultado obtido se baseou em testes de 10 a 70 épocas, que resultaram em valores semelhantes de acurácia, normalmente variando entre 10% e 20%.

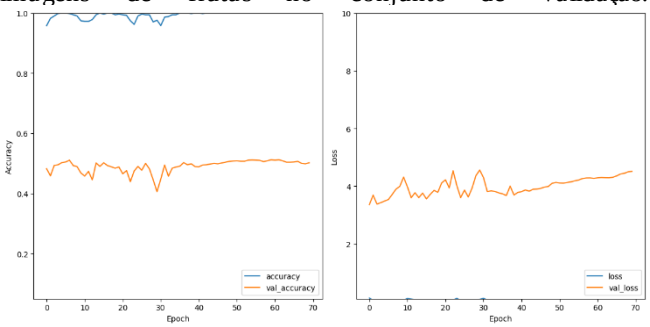
C. *Arquitetura Autoral*

Em relação à nossa arquitetura personalizada, avaliamos seu desempenho na tarefa de classificação de imagens de frutas usando o conjunto de dados Fruits262. Depois de treinar o modelo, obtivemos os seguintes resultados no conjunto de validação: uma perda de validação de 4,5134 e uma precisão de validação de 0,5020.

A perda de validação de 4,5134 indica a diferença média entre as probabilidades de classe preditas e os rótulos de verdade básica. Um valor de perda de validação menor indica um melhor alinhamento entre os rótulos previstos e reais, sugerindo que nosso modelo está fazendo previsões mais precisas no conjunto de validação.

A precisão de validação de 0,5020 representa a proporção de imagens classificadas corretamente do número total de imagens no conjunto de validação. Embora essa precisão seja modesta em comparação com algumas outras arquiteturas, ela indica que nossa arquitetura personalizada é capaz de

classificar corretamente aproximadamente 50,20% das imagens de frutas no conjunto de validação.



Esses resultados lançam luz sobre o desempenho de nossa arquitetura personalizada e fornecem informações sobre seus pontos fortes e limitações. Os valores de perda de validação e precisão obtidos sugerem que há espaço para melhorias nas capacidades preditivas do modelo. Mais investigação e otimização podem ser necessárias para aumentar a precisão e reduzir a perda, potencialmente por meio de ajustes nos hiper parâmetros, arquitetura do modelo ou técnicas de aumento de dados.

É importante observar que esses resultados servem como base para nossa arquitetura personalizada e estabelecema base para futuras iterações e melhorias. O desempenho alcançado até agora encoraja mais exploração e refinamento para liberar todo o potencial da arquitetura para tarefas de classificação de imagens de frutas.

CONCLUSÃO

Ao avaliar os resultados obtidos, observamos que o modelo autoral alcançou uma perda de 4,5134 e uma acurácia de 0,5020. Esses valores indicam um desempenho satisfatório na tarefa de classificação, com uma taxa de acerto superior em comparação com os modelos ResNet e DenseNet.

Por outro lado, os modelos ResNet e DenseNet obtiveram resultados semelhantes, com perdas de 3,0247 e 2,9706, respectivamente, e acurácias de 0,1 para ambos. Esses resultados sugerem que essas arquiteturas não foram tão eficientes na tarefa de classificação avaliada neste estudo.

No entanto, é importante considerar que a avaliação desses modelos se deve exclusivamente ao dataset escolhido pela equipe. Nesse contexto, os modelos ResNet e DenseNet não foram capazes de aprender padrões complexos o bastante para realizar a classificação com precisão. Ao contrário da arquitetura autoral que apresentou resultados mais expressivos.

Em suma, este estudo comparativo destacou o desempenho superior do modelo autoral em relação aos modelos ResNet e DenseNet, com uma acurácia significativamente maior. No entanto, mais pesquisas são necessárias para aprimorar os modelos e explorar abordagens que possam otimizar seu desempenho nas tarefas de classificação de imagens.