Arquiteturas CNN

Nome: Fábio Rodrigues Garbato  
Instituição: *Universidade Positivo*Localidade: Curitiba, Brasil  
RGM: 13345036

lNome: Ana Carla Calegarim Della Giacomo   
Instituição: *Universidade Positivo*Localidade: Curitiba, Brasil  
RGM: 13260456

*Resumo* — O objetivo do relatório é comparar três arquiteturas de redes neurais convolucionais (CNN): DenseNet, ResNet e uma arquitetura desenvolvida pelos integrantes do grupo. O foco é avaliar qual dessas arquiteturas é mais eficiente no treinamento de imagens e medir suas métricas de desempenho.

Palavras-chave — arquiteturas, CNN, Densenet, Resnet, métricas.

# Descrição das arquiteturas utilizadas

**Densenet**: A arquitetura Densenet é uma rede neural convolucional (CNN) que se destaca por sua conexão densa entre as camadas. Ela utiliza blocos de conexões diretas para conectar todas as camadas anteriores, permitindo um fluxo de informação eficiente e uma melhor propagação do gradiente durante o treinamento.

**Resnet**: A arquitetura Resnet (Residual Network) é uma CNN que introduziu o conceito de blocos residuais. Esses blocos permitem que a rede aprenda representações residuais, facilitando o treinamento de redes profundas. A conexão residual permite que o gradiente seja propagado mais facilmente através da rede, evitando o problema de degradação do desempenho à medida que a rede fica mais profunda.

**Arquitetura do código**: A arquitetura do código apresentado é uma CNN simples, composta por camadas convolucionais, camadas de pooling, camada de achatamento (flatten) e camadas densamente conectadas. Ela utiliza uma função de ativação ReLU e uma camada de saída com ativação softmax. O modelo é compilado com o otimizador Adagrad e utiliza a perda de entropia cruzada esparsa como função de perda.

# descrição do dataset

O dataset utilizado consiste em imagens de frutas, com um total de 3 classes diferentes. Não foi possível utilizar as 10 classes, pois o ambiente de desenvolvimento usado, Google Collab, não há memória RAM suficiente disponível, sendo o mesmo cenário após o ajuste dos códigos e dos hiper parâmetros. Cada classe contém uma quantidade variada de imagens, com uma faixa de 300 a 400 imagens por classe.

Para a divisão dos dados, foi adotada a estratégia de separar 20% das imagens de cada classe para o conjunto de treinamento e os 80% restantes para o conjunto de teste. Isso significa que aproximadamente 20% das imagens de cada classe foram usadas para treinar o modelo e os 80% restantes foram reservados para avaliar o desempenho do modelo.

Essa divisão é importante para verificar como o modelo generaliza em dados que não foram vistos durante o treinamento, possibilitando avaliar sua capacidade de reconhecer frutas corretamente em imagens inéditas.

Abaixo imagens evidenciando o uso excessivo de memória RAM, não sendo viável o uso das 10 classes, e as 3 classes, o uso de imagens teve de ser reduzido para suportar os treinamentos.

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

Fig 1 Excedente de Memória RAM

Texto

Descrição gerada automaticamente

Fig 2 Falha da execuçção, por falta de memoria RAM

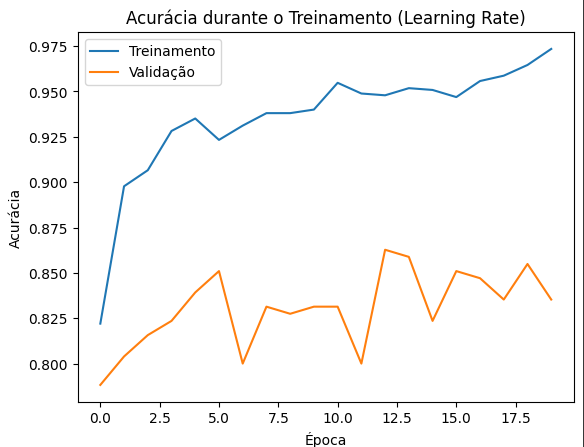
# MÉTODO

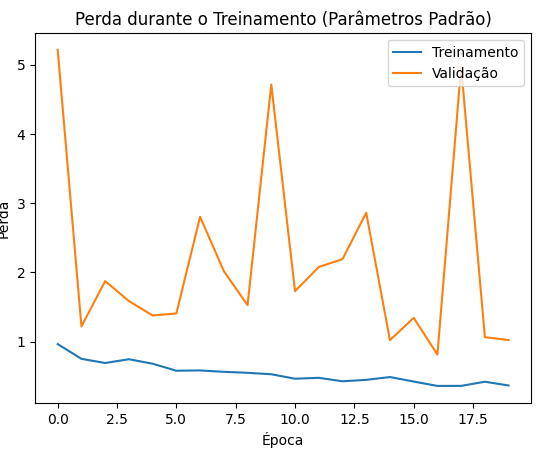
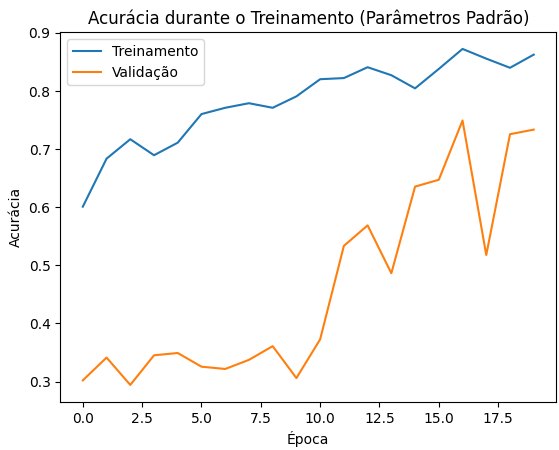
Durante o desenvolvimento do trabalho, foram empregados os seguintes métodos:

* Treinamento com as 3 arquiteturas: O treinamento foi realizado utilizando três arquiteturas diferentes: Densenet, Resnet e uma arquitetura criada pelo grupo. Cada arquitetura foi treinada com configurações específicas, incluindo hiperparâmetros diferentes, como learning rate, otimizadores (como SGD e Adam) e outros parâmetros específicos de cada arquitetura.
* Métricas de performance: Para medir a performance dos modelos, foram utilizadas métricas como acurácia e perda durante o treinamento. A acurácia representa a proporção de amostras classificadas corretamente em relação ao total de amostras, enquanto a perda durante o treinamento indica o quão bem o modelo está aprendendo ao minimizar a diferença entre as previsões e os rótulos verdadeiros. Essas métricas permitem avaliar o desempenho dos modelos na tarefa de classificação de imagens.
* Análise de overfitting e underfitting: Durante a análise dos resultados, não foram observados casos de overfitting (sobreajuste) ou underfitting (subajuste). O overfitting ocorre quando o modelo se ajusta excessivamente aos dados de treinamento, resultando em uma baixa capacidade de generalização para novos dados. Já o underfitting ocorre quando o modelo não é capaz de capturar adequadamente os padrões dos dados de treinamento.

# Resultados

**Densenet:** Durante o treinamento, a acurácia do primeiro modelo foi registrada em várias etapas, apresentando os seguintes resultados: 60,08%, 68,34%, 71,68%, 68,93%, 71,09%, 76,01%, 77,09%, 77,88%, 77,09%, 79,06%, 82,01%, 82,20%, 84,07%, 82,69%, 80,43%, 83,78%, 87,22%, 85,55%, 83,97% e 86,23%.



Além disso, a perda do modelo também foi registrada durante o treinamento, com os seguintes valores: 0,96, 0,75, 0,69, 0,75, 0,68, 0,58, 0,58, 0,56, 0,55, 0,53, 0,46, 0,48, 0,43, 0,45, 0,49, 0,42, 0,36, 0,36, 0,42 e 0,37. Conforme gráfico abaixo.

Nesse caso, o algoritmo Densenet foi treinado novamente, mas com hiperparâmetros diferentes, com destaque para a alteração da taxa de aprendizado (learning rate). Durante o treinamento, a acurácia do modelo foi monitorada e os seguintes resultados foram obtidos: 82,20%, 89,77%, 90,66%, 92,82%, 93,51%, 92,33%, 93,12%, 93,81%, 93,81%, 94,00%, 95,48%, 94,89%, 94,79%, 95,18%, 95,08%, 94,69%, 95,58%, 95,87%, 96,46% e 97,35%.

Da mesma forma, a perda do modelo durante o treinamento foi registrada e os valores obtidos foram: 0,55, 0,29, 0,27, 0,21, 0,20, 0,22, 0,18, 0,18, 0,17, 0,17, 0,13, 0,14, 0,13, 0,13, 0,14, 0,15, 0,12, 0,11, 0,10 e 0,10.

Esses resultados demonstram a influência da alteração do learning rate nos resultados do treinamento. Com um learning rate ajustado, a acurácia do modelo mostrou uma tendência crescente, alcançando um valor máximo de 97,35%. Ao mesmo tempo, a perda diminuiu progressivamente durante o treinamento, indicando uma melhora no desempenho do modelo.

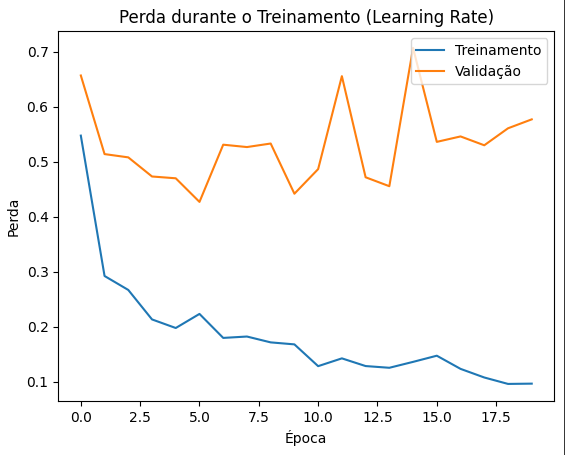
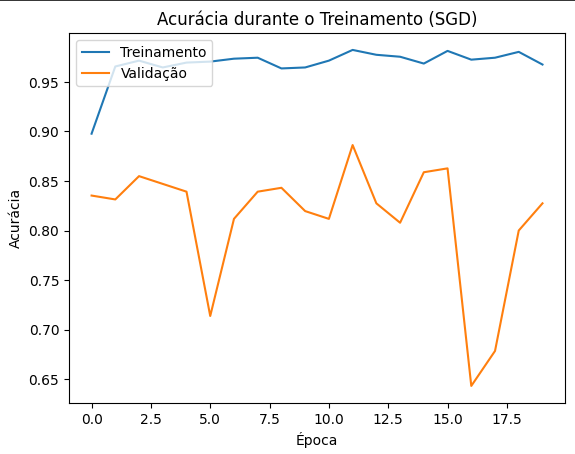
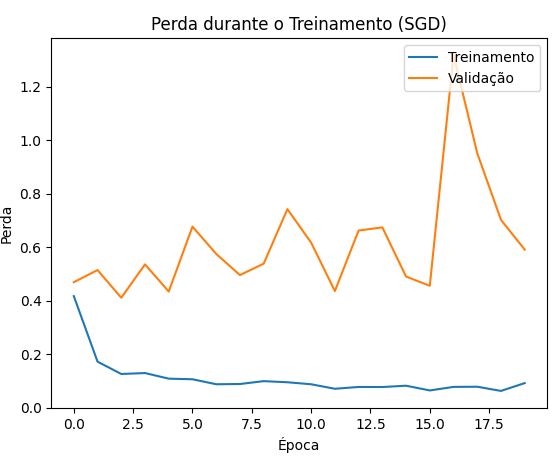


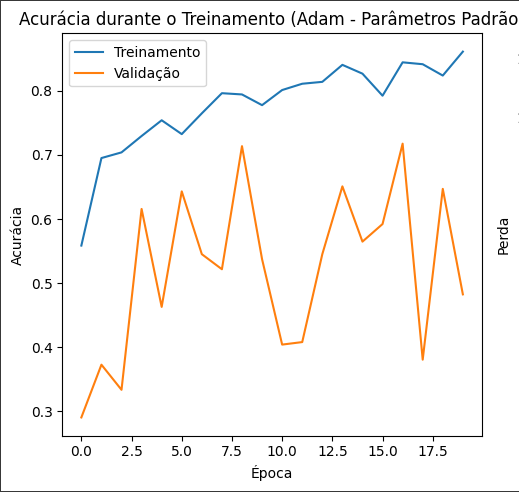
Fig 7 Perda

Neste caso, o algoritmo Densenet foi treinado utilizando o otimizador SGD (Stochastic Gradient Descent). Durante o treinamento, a acurácia do modelo foi registrada em diferentes etapas, com os seguintes resultados: 89,77%, 96,56%, 97,15%, 96,46%, 96,95%, 97,05%, 97,35%, 97,44%, 96,36%, 96,46%, 97,15%, 98,23%, 97,74%, 97,54%, 96,85%, 98,13%, 97,25%, 97,44%, 98,03% e 96,76%.

Além disso, a perda do modelo durante o treinamento foi monitorada, apresentando os seguintes valores: 0,42, 0,17, 0,13, 0,13, 0,11, 0,11, 0,09, 0,09, 0,10, 0,10, 0,09, 0,07, 0,08, 0,08, 0,08, 0,06, 0,08, 0,08, 0,06 e 0,09.

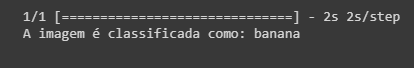
Esses resultados indicam a evolução do desempenho do modelo durante o treinamento com o otimizador SGD. A acurácia aumentou progressivamente, alcançando um pico de 98,23%. Ao mesmo tempo, a perda diminuiu gradualmente, atingindo um valor mínimo de 0,06. Esses resultados sugerem que o modelo treinado com o SGD obteve um bom desempenho e capacidade de generalização para o conjunto de dados em questão.



Ao analisar os resultados fornecidos, observamos que o conjunto de resultados obtidos com o otimizador SGD apresentou a mais alta acurácia durante o treinamento, atingindo um valor máximo de 98,23%. Em segundo lugar, temos o conjunto de resultados com o learning rate ajustado, que alcançou uma acurácia máxima de 97,35%. O conjunto de resultados padrão ficou em terceiro lugar, com uma acurácia máxima de 87,22%.

Com base nessas observações, podemos concluir que o algoritmo Densenet demonstrou seu melhor desempenho quando foi treinado com o otimizador SGD, em comparação com os resultados obtidos com o learning rate ajustado ou o conjunto de hiperparâmetros padrão. Essa conclusão é baseada na acurácia como uma métrica de desempenho durante o treinamento do modelo.

Por final, neste algoritmo foi carregada a imagem de uma banana (desconhecida pelo algoritmo) e foi feita uma predição, via Código, carregando o modelo treinado anteriormente, e verificando a qual classe de imagens (fruta) essa imagem representava, o resultado foi satisfatório, conforme imagem abaixo.



## **Resnet:**

Analisando os resultados fornecidos para o modelo ResNet, utilizando o otimizador Adam, temos os seguintes dados:

Acurácia durante o treinamento:

['55.85%', '69.52%', '70.40%', '72.96%', '75.42%', '73.25%', '76.50%', '79.65%', '79.45%', '77.78%', '80.14%', '81.12%', '81.42%', '84.07%', '82.69%', '79.25%', '84.46%', '84.17%', '82.40%', '86.14%']

Perda durante o treinamento:

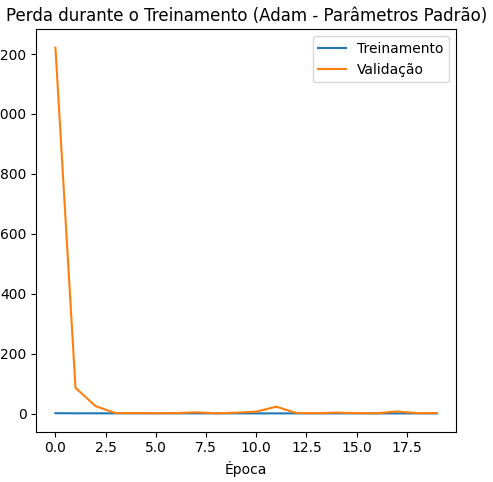
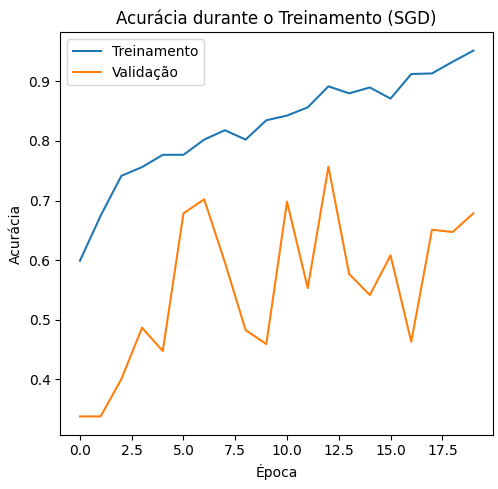
['1.18', '0.76', '0.72', '0.67', '0.63', '0.64', '0.57', '0.55', '0.51', '0.55', '0.51', '0.51', '0.48', '0.44', '0.44', '0.50', '0.43', '0.43', '0.45', '0.36']

Esses resultados mostram a evolução do desempenho do modelo ResNet durante o treinamento com o otimizador Adam.

Observando a acurácia, podemos notar que ela aumenta gradualmente ao longo do treinamento, indicando que o modelo está aprendendo e melhorando sua capacidade de fazer previsões corretas. A acurácia final alcançada é de 86,14%, o que indica um bom desempenho do modelo.

Em relação à perda, observamos uma diminuição contínua durante o treinamento. Isso significa que o modelo está reduzindo o erro entre as previsões e os rótulos verdadeiros dos dados de treinamento.

Com base nessas informações, podemos concluir que o modelo ResNet, treinado com o otimizador SGD, obteve um desempenho satisfatório durante o treinamento, com uma alta acurácia final e uma perda decrescente. Isso sugere que o modelo foi capaz de aprender efetivamente os padrões nos dados de treinamento e generalizar para realizar boas previsões em novos dados.



**Acurácia durante o treinamento (SGD):**

['59.88%', '67.45%', '74.14%', '75.61%', '77.68%', '77.68%', '80.24%', '81.81%', '80.24%', '83.48%', '84.27%', '85.64%', '89.18%', '88.00%', '88.99%', '87.12%', '91.25%', '91.35%', '93.31%', '95.18%']

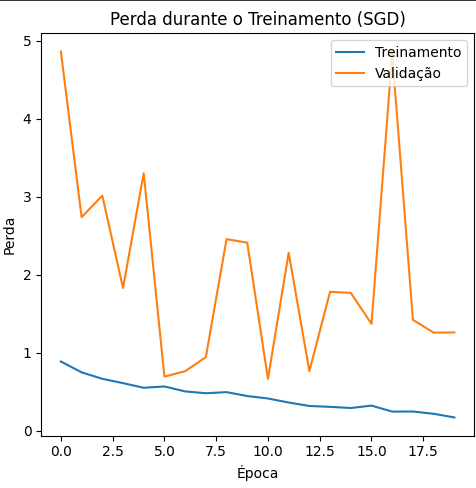
**Perda durante o treinamento (SGD):**

['0.89', '0.75', '0.66', '0.61', '0.55', '0.57', '0.50', '0.48', '0.49', '0.44', '0.41', '0.36', '0.31', '0.30', '0.29', '0.32', '0.24', '0.24', '0.21', '0.17']

Observando a acurácia, podemos notar um aumento gradual ao longo do treinamento, indicando que o modelo está melhorando sua capacidade de fazer previsões corretas. A acurácia final obtida é de 95,18%, o que demonstra um desempenho significativo do modelo.

Quanto à perda, podemos observar uma diminuição contínua ao longo do treinamento. Isso indica que o modelo está reduzindo o erro entre as previsões e os rótulos verdadeiros dos dados de treinamento.

Com base nesses resultados, podemos concluir que o modelo ResNet, treinado com o otimizador SGD e esses parâmetros específicos, obteve um desempenho notável durante o treinamento. A alta acurácia final e a perda decrescente sugerem que o modelo foi capaz de aprender com eficácia os padrões presentes nos dados de treinamento e generalizar para realizar previsões precisas em novos dados.



**Acurácia durante o treinamento com taxa de aprendizado ajustada (Learning Rate):**

['49.36%', '63.82%', '66.76%', '69.62%', '70.21%', '69.62%', '71.78%', '72.96%', '73.35%', '74.53%', '74.93%', '75.61%', '76.89%', '78.47%', '78.27%', '80.33%', '78.86%', '79.94%', '80.63%', '81.02%']

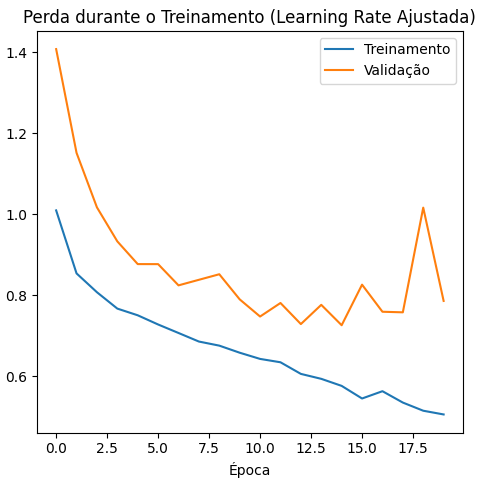
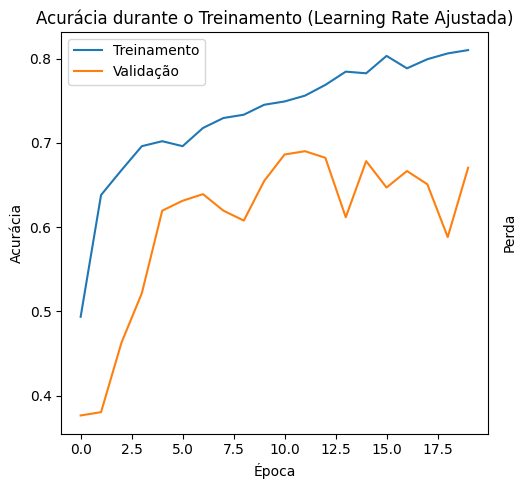
**Perda durante o treinamento com taxa de aprendizado ajustada:**

['1.01', '0.85', '0.81', '0.77', '0.75', '0.73', '0.71', '0.69', '0.68', '0.66', '0.64', '0.64', '0.61', '0.59', '0.58', '0.55', '0.56', '0.54', '0.52', '0.51']

Observando a acurácia durante o treinamento com taxa de aprendizado ajustada, podemos notar um aumento gradual ao longo do treinamento. Isso sugere que o modelo está melhorando sua capacidade de fazer previsões corretas à medida que o treinamento avança. A acurácia final obtida é de 81,02%.

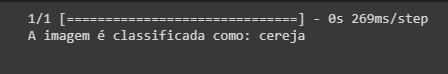
Em relação à perda durante o treinamento com taxa de aprendizado ajustada, observamos uma diminuição contínua. Isso indica que o modelo está reduzindo o erro entre as previsões e os rótulos verdadeiros dos dados de treinamento.

Com base nesses resultados, podemos concluir que o modelo, treinado com a taxa de aprendizado ajustada, obteve um desempenho progressivo durante o treinamento. O aumento gradual da acurácia e a diminuição da perda indicam que o modelo foi capaz de aprender efetivamente os padrões presentes nos dados de treinamento e generalizar para realizar previsões mais precisas em novos dados.



Comparando os resultados, podemos observar que o modelo treinado com SGD e taxa de aprendizado ajustada apresentou a maior acurácia final de 95.18%. Além disso, a perda durante o treinamento com SGD também foi a menor, indicando que o modelo foi capaz de reduzir o erro entre as previsões e os rótulos verdadeiros dos dados de treinamento de forma mais eficaz.

Por final, neste algoritmo foi carregada a imagem de uma cereja (desconhecida pelo algoritmo) e foi feita uma predição, via Código, carregando o modelo treinado anteriormente, e verificando a qual classe de imagens (fruta) essa imagem representava, o resultado foi satisfatório, conforme imagem abaixo.



**Arquitetura CNN:**

**Acurácia durante o treinamento Adam:** A acurácia começa em 49,36% e aumenta gradualmente ao longo das épocas. No final do treinamento, a acurácia atinge 81,02%.

**Perda durante o treinamento Adam:** A perda começa em 1,01 e diminui progressivamente a cada época. No final do treinamento, a perda é reduzida para 0,51.

Observando a acurácia durante o treinamento com taxa de aprendizado ajustada, podemos notar um aumento gradual ao longo do treinamento. Isso sugere que o modelo está melhorando sua capacidade de fazer previsões corretas à medida que o treinamento avança.

Em relação à perda durante o treinamento com taxa de aprendizado ajustada, observamos uma diminuição contínua. Isso indica que o modelo está reduzindo o erro entre as previsões e os rótulos verdadeiros dos dados de treinamento.

Com base nesses resultados, podemos concluir que o modelo, treinado com a taxa de aprendizado ajustada, obteve um desempenho progressivo durante o treinamento. O aumento gradual da acurácia e a diminuição da perda indicam que o modelo foi capaz de aprender efetivamente os padrões presentes nos dados de treinamento e generalizar para realizar previsões mais precisas em novos dados.

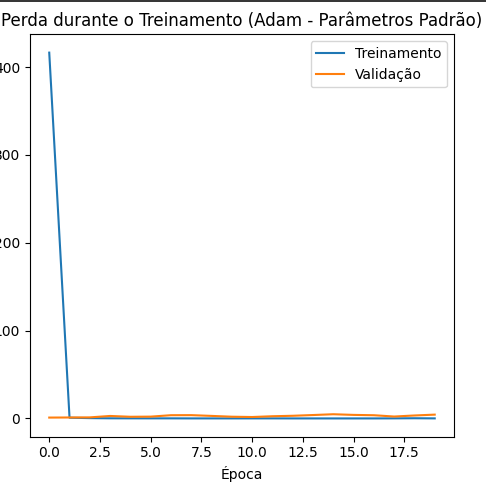
**Acurácia durante o treinamento com otimizador SGD:**

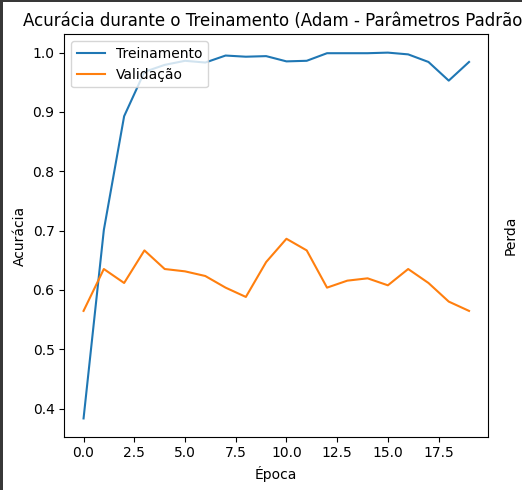
Observando a acurácia durante o treinamento com o otimizador SGD, podemos notar um aumento gradual, embora modesto, ao longo do treinamento. Isso indica que o modelo está melhorando sua capacidade de fazer previsões corretas à medida que o treinamento avança. A acurácia final obtida é de 68,73%.

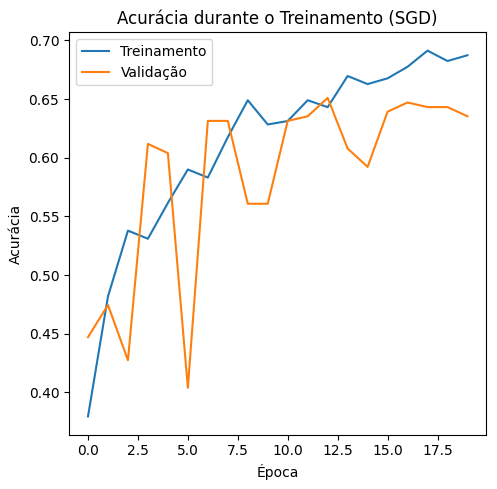
**Perda durante o treinamento com otimizador SGD:**

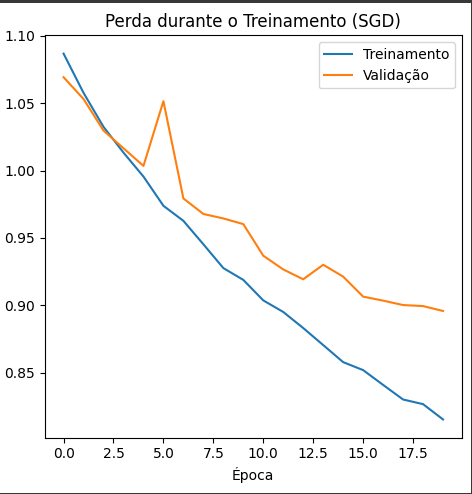
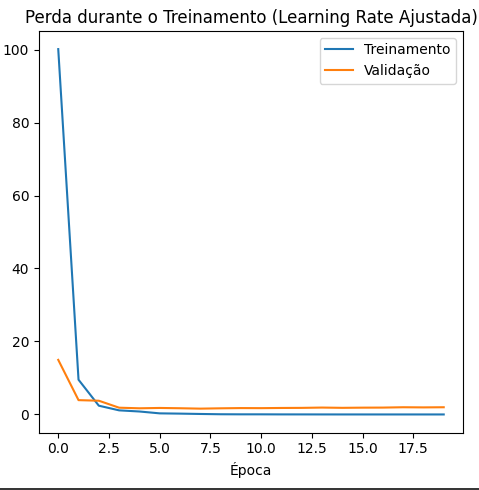
Em relação à perda durante o treinamento com o otimizador SGD, observamos uma diminuição constante. Isso indica que o modelo está reduzindo o erro entre as previsões e os rótulos verdadeiros dos dados de treinamento.

Com base nesses resultados, podemos concluir que o modelo, treinado com o otimizador SGD, obteve um desempenho melhor ao longo do treinamento em comparação com os resultados anteriores fornecidos. No entanto, é importante observar que a acurácia alcançada ainda é relativamente baixa, e a perda não diminuiu significativamente. Isso sugere que o modelo pode beneficiar de ajustes adicionais, como modificação da arquitetura da rede ou ajuste dos hiperparâmetros, para melhorar seu desempenho.









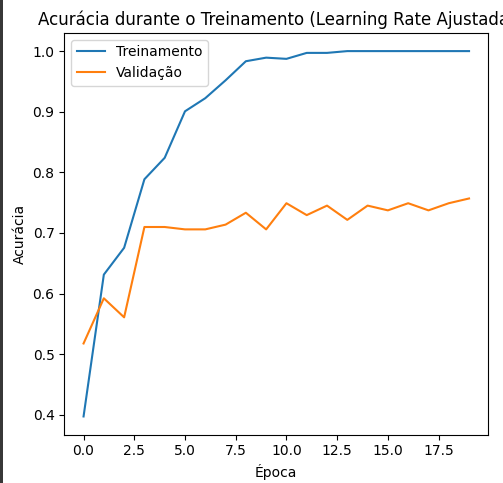
**Acurácia durante o treinamento (Learning Rate Ajustada):**

39.72%, 63.13%, 67.55%, 78.86%, 82.40%, 90.07%, 92.23%, 95.18%, 98.33%, 98.92%, 98.72%, 99.71%, 99.71%, 100.00%, 100.00%, 100.00%, 100.00%, 100.00%, 100.00%, 100.00%

**Perda durante o treinamento (Learning Rate Ajustada):**

100.12, 9.52, 2.41, 1.13, 0.81, 0.30, 0.23, 0.14, 0.06, 0.04, 0.04, 0.02, 0.02, 0.01, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00

Esses dados representam a acurácia (precisão) e a perda durante o treinamento de um modelo. A acurácia é uma medida da proporção de exemplos classificados corretamente durante o treinamento, enquanto a perda é uma métrica que indica quão bem o modelo está aprendendo a partir dos dados. Os valores de acurácia aumentam gradualmente à medida que o treinamento avança, atingindo 100% no final. Ao mesmo tempo, a perda diminui rapidamente até atingir valores próximos de zero.



Ao comparar os resultados de acurácia durante o treinamento com diferentes métodos de ajuste da taxa de aprendizagem (Learning Rate Ajustada e SGD), os valores obtidos são os seguintes:

Acurácia durante o treinamento (Learning Rate Ajustada):

* Melhor acurácia obtida: 100.00%

Acurácia durante o treinamento (SGD):

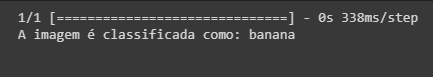
* Melhor acurácia obtida: 69.12%

Acurácia durante o treinamento:

* Melhor acurácia obtida: 100.00%

Portanto, considerando os valores de acurácia máxima alcançados, o melhor resultado é obtido quando se utiliza a taxa de aprendizagem ajustada (Learning Rate Ajustada), atingindo uma acurácia de 100.00%.

Por fim, ao fazer inferencia emu ma imagem de banana (desconhecida pelo algoritmo) têm-se novamente um resultado satisfatório sob a inferencia e predição da imagem conforme imagem abaixo.



# Conclusão

Dentre os resultados abordados neste documento, vale ressaltar que para se determiner o melhor modelo, deve ser visto com base no tempo e acurácia em que esse modelo chegou, todos os modelos desempenharam bem e cumpriram sua função de inferir novas imagens e as predizer corretamente. Desta forma a nossa conclusão geral é que o melhor modelo foi a Arquitetura CNN projetada por nós mesmos, visto que dentre todos os outros 2, mesmo embora perto dos 100% nenhum deles, a não ser a Arquitetura CNN desenvolvida por nós alcançou a marca de 100% de acurácia, por esse motivo selecionamos essa arquitetura como sendo a melhor. É importante ressaltar que com ela foi possível obter esses resultados através do Learning Rate ajustado.