

Relatório - Trabalho Cassava

Juan Burtet - 20103652

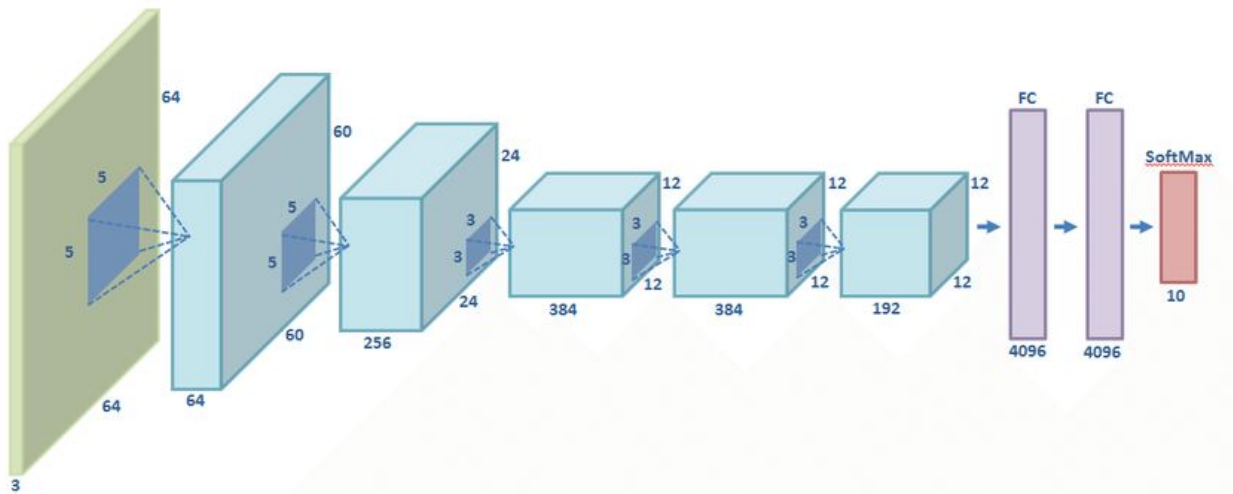
Classificação de Doenças em Folhas de Cassava

O Trabalho em questão irá apresentar todo o processo utilizado para encontrar modelos de aprendizado de máquina capazes de classificar doenças através de fotos de folhas de Cassava. Este relatório será dividido em: Primeiros testes, Comparação de Modelos e finalizando com ajustes ao modelo de melhor resultado.

Primeiros Testes

O problema em questão utiliza um conjunto de dados de imagens para ser feita a classificação das doenças. Como é um problema de classificação de imagens, as redes neurais são os modelos de aprendizado de máquina que possuem a maior capacidade de aprendizagem neste problema. Entretanto, um dos grandes problemas nas redes neurais (principalmente nas profundas) é o elevado tempo de processamento para se obter resultados interessantes. Uma possível solução para este problema é a utilização de **Transfer Learning**, onde pegamos uma rede treinada para solução de um problema, e a reutilizamos para encontrar a solução para o nosso. Todas as redes treinadas neste trabalho irão utilizar este método.

O primeiro modelo testado foi a rede **AlexNet**. Esta rede já está implementada (e pré-treinada) dentro da Biblioteca do **FastAI**. A **AlexNet** possui 8 camadas; as 5 primeiras são camadas de convolução, algumas delas seguidas por camadas de max-pooling e as últimas duas sendo camadas FC (Fully Connected). Além disso, utiliza a função de Ativação ReLU e decide o resultado através de uma SoftMax.



O Notebook disponibilizado pelo professor Ricardo utiliza a **AlexNet** e pode ser encontrado neste [link](#). Neste primeiro teste, foi reutilizado o mesmo Notebook, com a diferença de treinar o modelo por 100 épocas. O modelo obteve 73% de acurácia ao conjunto de validação e 60,5% de acurácia no conjunto de teste (submissão do Kaggle).

S00 - Primeira Submissão
(version 2/2)

21 hours ago by Juan Burtet

Succeeded

0.605

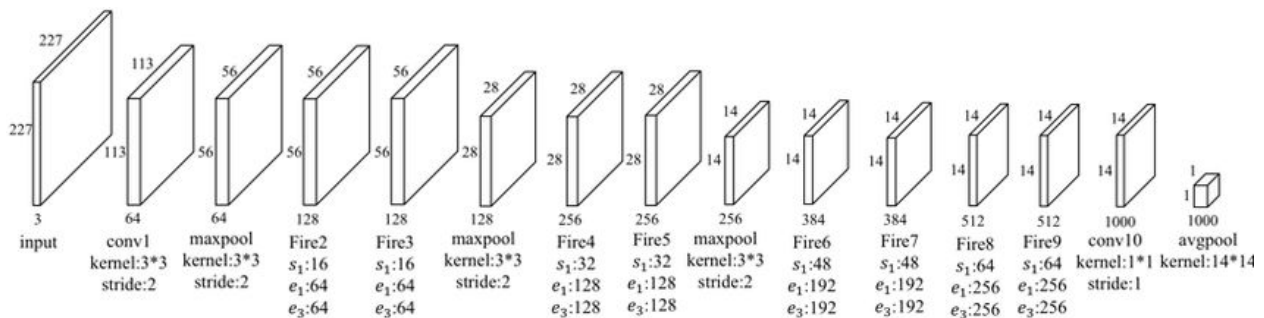
From Notebook [S00 - Primeira Submissão]

Comparação de Modelos

Com a facilidade de ter diferentes métodos já implementados e pré-treinados ao **FastAI**, foi avaliado 4 modelos de CNN, para verificar qual possui a maior capacidade de aprendizado para o problema. Os modelos selecionados foram: **SqueezeNet1_1**, **Densenet121**, **VGG19_bn** e **ResNet152**. Cada um dos modelos será apresentado separadamente, junto de seus resultados. Para decidir qual é a melhor rede a se utilizar, foi avaliado o desempenho ao fim de 10 épocas usando **fine_tune** (reutilizar os pesos pré-treinados para encontrar novos valores que sejam melhores para o nosso problema).

SqueezeNet1_1

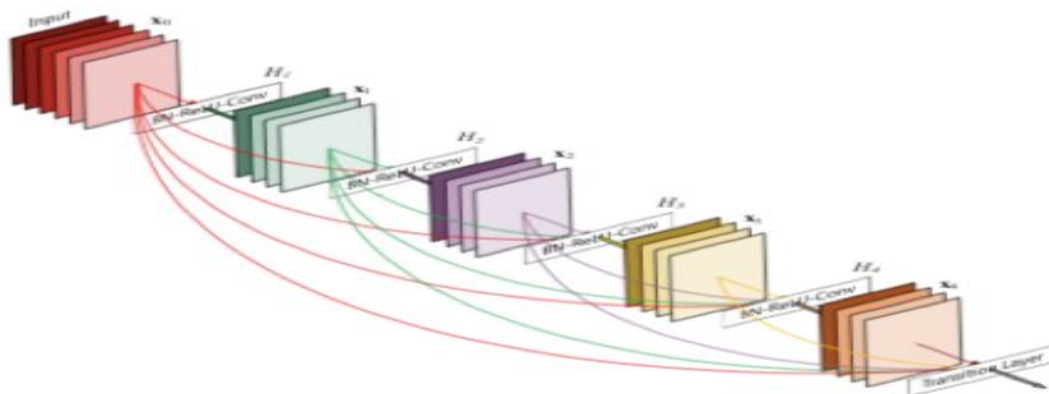
A **SqueezeNet1_1** é um update da SqueezeNet padrão, que tinha como objetivo possuir a mesma acurácia da AlexNet, só que utilizando 50x menos parâmetros e com um modelo de tamanho reduzido. Sendo assim, a **SqueezeNet1_1** tem como objetivo diminuir o tempo de processamento sem sacrificar a acurácia.



Ao fim das 10 épocas de treinamento, o modelo chegou a um acurácia de 81% no conjunto de validação e tendo uma loss de 0.3874 no conjunto de treino.

DenseNet121

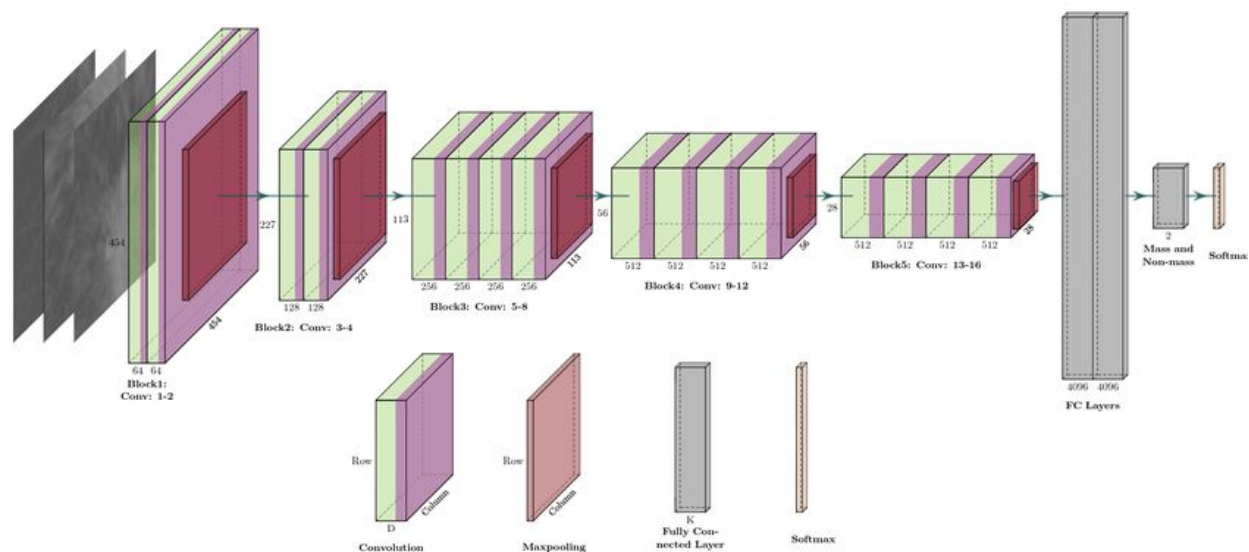
Um dos grandes problemas das redes neurais profundas é a perda total (ou explosão) de gradiente. Sendo que as partes iniciais da rede irão receber cada vez menos informação, levando a um grande tempo de treinamento. As **DenseNets** resolvem esse problema conectando a saída de uma camada com todas as próximas.



Ao fim das 10 épocas de treinamento, o modelo chegou na acurácia de 82.9%, o que não parece ser um grande aumento em relação a **SqueezeNet**, mas obteve um loss de 0.0295, o que indica uma melhor capacidade de aprendizagem na mesma quantidade de épocas.

VGG19_bn

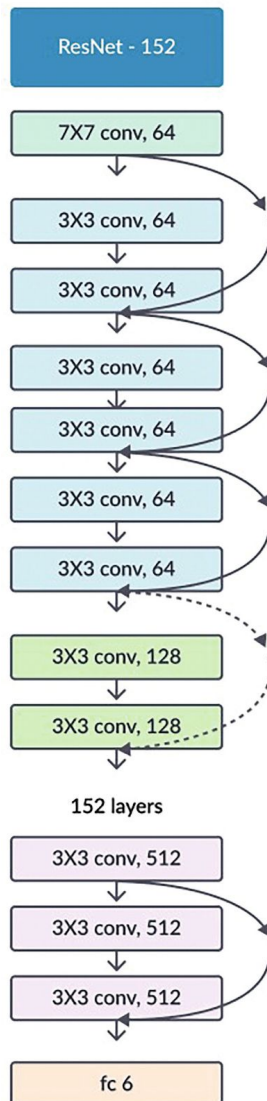
A **VGG19** é considerada uma das sucessoras da **AlexNet**, só que possuindo uma arquitetura mais profunda. Como o nome já diz, é uma rede com 19 camadas usadas para classificar imagens, possuindo diversas camadas de convolução, seguidas pelas Fully Connecteds no fim.



Após o fim do treino de 10 épocas, a rede obteve uma acurácia de 84,2% no conjunto de validação, junto de uma loss de 0.0128. É possível verificar que todas as redes obtiveram um melhor resultado que as anteriores, principalmente ao verificar a loss.

ResNet152

As **ResNet** surgiram devido ao mesmo problema explicados na **DenseNet**. Enquanto as **DenseNets** "solucionam" o problema conectando a saída de cada camada com todas as seguintes, as **ResNets** acabam passando as saídas para as próximas camadas, pulando de 2 em 2 (vai fazer mais sentido vendo a imagem). As **ResNet152** possuem 152 camadas, como o nome já indica.



Após o treinamento por 10 épocas, a rede obteve acurácia de 84.62% no conjunto de validação, com uma loss de 0.0098. Com todos os resultados obtidos, foi percebido que a **ResNet152** possui a maior capacidade de aprendizado durante 10 épocas, seja pela loss ou acurácia. Pelo resultado da loss, é indicado que a rede consegue ter uma boa capacidade de se adaptar ao conjunto de treinamento, e pela acurácia do conjunto de validação que a rede não possui overfit ao conjunto de treino. Com essas informações, a rede **ResNet152** será utilizada para mais uma etapa de treinamento para ser avaliada no conjunto de teste do Kaggle.

“Melhorando” a ResNet512

Com os resultados encontrados, foi verificado que a **ResNet152** obteve o melhor desempenho para este problema. Sendo assim, foi recuperado o modelo da **ResNet152** treinado por 10 épocas, sendo utilizado por mais uma rodada de treino (por mais 25 épocas). Ao fim de todo esse processamento, a acurácia no conjunto de validação chegou ao resultado de 90%. Este modelo possui alguns resultados diferentes ao início do treino, pois ao abrir o modelo e treinar por mais 25 épocas, a divisão treino/validação acabou sendo diferente do teste anterior, causando uma loss maior e uma acurácia menor (pois muitos dos exemplos do treino passado, estavam na validação), que foram se estabilizando com o passar das épocas. O exemplo dessa variação pode ser vista na imagem abaixo. (Infelizmente, a saída não ficou salva no Kaggle, mas foi recuperado uma imagem do meio processamento).

epoch	train_loss	valid_loss	error_rate	accuracy	time
0	0.130022	0.120361	0.032017	0.967983	05:34
1	0.057972	0.146544	0.034821	0.965179	05:36
2	0.055309	0.180320	0.041599	0.958402	05:30
3	0.076930	0.212332	0.054452	0.945548	05:30
4	0.098909	0.252761	0.068240	0.931760	05:23
5	0.114367	0.260657	0.076420	0.923580	05:19
6	0.120322	0.460879	0.130872	0.869128	05:24
7	0.118628	0.411405	0.109839	0.890161	05:25
8	0.104808	0.498953	0.121991	0.878009	05:20
9	0.096955	0.428018	0.104931	0.895069	05:16
10	0.071178	0.406080	0.104230	0.895770	05:24
11	0.049312	0.481387	0.113812	0.886188	05:32
12	0.047079	0.433162	0.103763	0.896237	05:37

85.02% [227/267 03:54<00:41 0.0403]

Com o modelo treinado, iremos submeter ao conjunto de teste do Kaggle para ver sua pontuação. (Que neste caso, é a acurácia no conjunto de teste)



S02 - Submissão do Modelo ResNet512 (fine_tune=35)
Teste final (version 1/1)

Succeeded

0.843

3 hours ago by Juan Burtet

Com este resultado, foi possível alcançar a posição 1271 do Ranking.

1271	Juan Burtet		0.843	3	3h
Your Best Entry ↑					
Your submission scored 0.843, which is an improvement of your previous score of 0.605. Great job!					
					 Tweet this!

Analisando os melhores resultados do Ranking, foi verificado que a melhor acurácia encontrada foi a de 0.903, com esse resultado sendo adquirido através de outras arquiteturas de rede. Um dos motivos da utilização dessas redes para esse trabalho, foi a já implementação no **FastAI**, junto do pré-treinamento. O pré-treinamento acaba por ajudar muito na solução de problemas de imagem, pois em uma rede profunda, as camadas iniciais são utilizadas para encontrar padrões nas imagens que podem ajudar em outros tipos de problemas. Com isso, uma possível nova solução para este problema é ir atrás de redes que possuam uma maior capacidade de aprendizagem (provavelmente a **EfficientNet**), junto de uma busca por melhores hiperparâmetros. Além disso, acredito que um maior estudo das imagens, através de um bom pré-processamento, pode levar a uma acurácia maior e uma melhor colocação no Ranking Geral.

Curiosidades

A ResNet152 (treinada por 10 épocas) foi adicionada como Submissão do Kaggle e obteve um péssimo resultado, com apenas 0.353 de Acurácia. Não sei dizer se houve algum problema no código de previsão ou poucas épocas acabaram não dando conhecimento suficiente para solucionar o problema (ou talvez, a modificação do conjunto de treino/validação pode ter adicionado variedade ao conhecimento).

S01 - Submissão do Modelo ResNet512 (fine_tune=10)
Treinado por apenas 10 épocas (fine_tune) (version 1/1)
3 hours ago by Juan Burtet

Succeeded

0.353

Todos os códigos podem ser encontrados na pasta Notebooks, junto de algumas imagens (pasta img) dos resultados. Os modelos podem ser encontrados nesta [pasta do Google Drive](#). Os modelos não foram adicionados junto do zip, por ter modelos com quase 1GB de tamanho. O Notebook da submissão final é chamado de "S02 - Submissão da Resnet152 (25+10 épocas)" e possui o passo a passo caso você queira fazer uma submissão desse modelo. Os outros Notebooks estão incluídos, mas não estão devidamente comentados. Além disso, alguns foram feitos no Colab, e outros no Kaggle. Por isso, tenha cuidado quando for ler o dataset.

Ignorar que eu escrevo ResNet512 ao invés de ResNet152 nas submissões do Kaggle.