Relatório Final - MAC0215
Gabriel Araujo - NUSP: 10297689
Supervisor: Prof. Roberto Hirata

Introdução: Por que estudar a vegetação urbana?

O primordial nesse projeto foi entender essa questão. Qual a utilidade de analizar toda a vegetação em um município? O que fazer e o que pode ser feito com os dados coletados?

Primeiro, é necessário compreender os possíveis cenários de uma árvore num ambiente metropolitano. No melhor caso, temos um crescimento vegetativo que não entra em conflito com nenhuma estrutura, seja ela calçada, fios de energia/comunicações ou até mesmo residências e estabelecimentos comerciais. Quando alguma dessas interferências acontece, é necessária uma poda ou remoção da árvore, o que, por si só, já traz consequências ambientais negativas, tanto em micro quanto macro escala.

Além do mal ambiental, também há o custo, não só das podas e remoções, mas também dos danos causados por esses conflitos: Pode haver desde um curto elétrico, a até mesmo incêndios causados por danos nos fios, além de danos que comprometam as estruturas previamente mencionadas.

O INACITY entra nesse cenário como uma medida preventiva, ao contrário de hoje, onde, na Prefeitura de São Paulo, por exemplo, só é tomada ação quando há alguma reclamação ou é registrado algum problema derivado da intromissão da vegetação no meio urbano. Especificamente, o sistema busca, através da análise de imagens fornecidas pelo Google Maps, identificar cenários onde isso esteja ocorrendo para poder informar o órgão responsável com antecedência, minimizando danos ou até mesmo os evitando completamente.

Minha parte no projeto foi implementar uma solução para um desses problemas, a interseção de fios em árvores. Como queria aprofundar meu estudo em Machine Learning, foi concordado com meu supervisor que abordaria esse cenário com essa metodologia; especificamente, faria a implementação de uma Rede Neural em Python 3.6, que é a linguagem em que o INACITY está sendo desenvolvido.

A Rede Neural: Desenvolvimento, conceitos e tratamento de imagens:

Como mencionado brevemente no pôster do meu projeto, a implementação da Rede Neural dependeu da conceitualização do que é, de fato, uma interseção de fio em árvore. Como haviam limitações na qualidade das imagens, foi optado por não considerar possíveis interseções que geravam dúvida sobre sua real existência. Especificamente, fios muito finos ou ocorrências de interseção em ângulos de difícil visualização ou muito distantes da câmera foram descartados, assim como interseções próximas das bordas das imagens.

O útlimo caso foi adotado por que, caso uma interseção ocorra em uma das extremidades da imagem, devido ao modo como são registradas, em uma das imagens adjacentes a mesma interseção também ocorreria, e provavelmente de forma mais clara.

Dito isso, o próximo passo, que na verdade ocorreu simultaneamente, foi de escolher qual biblioteca de Redes Neurais/Deep Learning deveria ser utilizada. Devido à sua simplicidade e confiabilidade, a Rede Neural Convolucional (CNN) da biblioteca do pacote Keras foi escolhida para essa tarefa. Os primeiros testes foram feitos com apenas 154 imagens, o que, como se pode imaginar, não resultou em um bom treinamento - a acurácia da rede oscilou entre 48% e 50%, e como haviam apenas duas categorias para a classificação (sem interseção - “semFio”, com interseção - “comFio”), é trivial que nada estava sendo aprendido pela rede.

Com meus conhecimentos prévios de Machine Learning, estava claro que a prioridade era aumentar meu conjunto de imagens. Nos testes seguintes, reuni em torno de 1400 imagens a mais, seguidas de mais outras 600, para completar um total de pouco mais 2200 imagens somando os conjuntos de treinamento, validação e teste (seguindo uma respectiva partição de 70%/15%/15% para os conjuntos). As planilhas com minha respectiva classificação para cada imagem do conjunto de dados estão no meu GitHub do projeto (github.com/Ga-briel/IC).

Paralelo a obtenção das imagens, fui testando diferentes técnicas para otimizar o treinamento da rede. Primeiro, verifiquei se havia diferença entre a utilização do formato RGB e Grayscale na acurácia da rede - nenhum teste com Grayscale passou de 50% de acurácia (portanto foi descartado e escolhido o RGB). Depois, mudei o foco para a resolução das imagens: As imagens fornecidas eram, originalmente 640x640, uma resolução relativamente baixa - dada a natureza do problema, não é difícil imaginar que, um redimensionamento muito drástico pode ofuscar, ou até eliminar, os possíveis fios presentes em uma imagem. Dito isso, treinei a rede neural nas seguintes resoluções: 32x32, 64x64, 128x128, 220x220 e 640x640. Nas resoluções maiores que 64x64, todas as redes ficaram com desempenho exato de 77.58%, que é exatamente a proporção de imagens da categoria "semFio", portanto não estavam sendo, de fato, treinadas. 32x32 e 64x64 tiveram acurácias relativamente parecidas (73.5% e 74.4% - sem tratamento de imagem, respectivamente), mas como a posterior tinha menos chances de não estar detectando os fios, pelo fato de ser maior, e por ter uma acurácia melhor, 64x64 foi escolhida como a resolução óptima para o projeto. Vale ressaltar que, após cada modelo ser treinado e testado, este também era testado em um conjunto teste perfeitamente equilibrado, com 50% de imagens em cada categoria, para verificar se o desempenho do mesmo não está sendo baseado em responder a maioria dos testes com apenas uma categoria de classificação.

Vejamos, como exemplo, o comportamento de um treinamento bem sucedido e um mal sucedido:

```
772/772 [==============================] - 203s 263ms/step - loss: 0.0348 - acc: 0
.9858 - val_loss: 1.2809 - val_acc: 0.7394
Epoch 19/20
  1/772 [..............................] - ETA: 31s - loss: 9.2607e-04 - acc: 1.00
  3/772 [..............................] - ETA: 32s - loss: 0.0027 - acc: 1.0000
772/772 [==============================] - 279s 362ms/step - loss: 0.0408 - acc: 0
.9837 - val_loss: 1.1232 - val_acc: 0.7515
Epoch 20/20
772/772 [==============================] - 283s 366ms/step - loss: 0.0361 - acc: 0
.9869 - val_loss: 0.9883 - val_acc: 0.8273
Model saved
jupyter-gabriel@greenery01:~$
```

Aqui, temos as últimas épocas de um treinamento com imagens 32x32, onde *val_acc* é a acurácia no conjunto de validação, *acc* é a acurácia no conjunto teste, e *loss* e *val_loss* são as acurácias nos conjuntos teste e de validação, respectivamente. Como é evidente, a rede está de fato aprendendo, com uma acurácia final no conjunto validação de 82%. Abaixo, temos o cenário oposto: Na resolução 220x220 , diversas épocas de treinamento já se passaram, mas *val_acc* continua o mesmo, 77.58%, que é o caso mencionado acima, onde a Rede simplesmente trata todos os casos dados como uma categoria só, já que os conjuntos não são iguais em tamanho (77.58% em "semFio" e o restante em "comFio").

```
772/772 [==============================] - 1433s 2s/step - loss: 4.9741 - acc: 0.6
880 - val_loss: 3.5750 - val_acc: 0.7758
Epoch 7/20
772/772 [==============================] - 1434s 2s/step - loss: 4.9796 - acc: 0.6
877 - val_loss: 3.5750 - val_acc: 0.7758
Epoch 8/20
772/772 [==============================] - 1433s 2s/step - loss: 4.9782 - acc: 0.6
877 - val_loss: 3.5750 - val_acc: 0.7758
Epoch 9/20
772/772 [==============================] - 1434s 2s/step - loss: 4.9713 - acc: 0.6
882 - val_loss: 3.5750 - val_acc: 0.7758
Epoch 10/20
772/772 [==============================] - 1430s 2s/step - loss: 4.9658 - acc: 0.6
885 - val_loss: 3.5750 - val_acc: 0.7758
Epoch 11/20
772/772 [==============================] - 1435s 2s/step - loss: 4.9920 - acc: 0.6
869 - val_loss: 3.5750 - val_acc: 0.7758
Epoch 12/20
772/772 [==============================] - 1432s 2s/step - loss: 4.9617 - acc: 0.6
888 - val_loss: 3.5750 - val_acc: 0.7758
Epoch 13/20
772/772 [==============================] - 1433s 2s/step - loss: 4.9727 - acc: 0.6
881 - val_loss: 3.5750 - val_acc: 0.7758
Epoch 14/20
```

Após isso, iniciei, como sugerido pelo meu supervisor, o tratamento de imagens dos conjuntos de dados. Em vários casos, **como no mostrado abaixo**, tínhamos sombras que ofuscavam as características das árvores, ou até mesmo dificultavam a visibilidade dos fios. A técnica usada, a equalização de histograma, permite que normalizemos o contraste da

imagem, afim de reduzir essa diferença. Em outras situações, a presença dos fios se tornou mais clara com esse procedimento. Abaixo, na primeira imagem, temos uma imagem sem tratamento. É possível perceber que a sombra criada pela folhagem da árvore oculta certas partes do fio ao observador.

Já na próxima imagem, com a equalização de histograma, o espaço assinalado mostra como a visibilidade do fio aumentou drasticamente.

Rede final e carga horária:

Após o desenvolvimento e tratamento de imagem, obtive uma Rede com acurácia de 75.67%, utilizando a resolução de 64x64, como podemos ver abaixo (O modelo, script de teste e conjunto de imagens estão disponíveis na minha máquina virtual em Jupyter. Para acessá-la, entre em contato comigo em ga.briel@usp.br).

```
/opt/tljh/user/lib/python3.6/importlib/_bootstrap.py:219: RuntimeWarning: numpy.dtype size changed, may indicate binary inc
ompatibility. Expected 96, got 88
  return f(*args, **kwds)
/opt/tljh/user/lib/python3.6/importlib/_bootstrap.py:219: RuntimeWarning: numpy.dtype size changed, may indicate binary inc
ompatibility. Expected 96, got 88
  return f(*args, **kwds)
2018-11-30 11:55:17.887857: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this T
ensorFlow binary was not compiled to use: AVX2 FMA
Loaded model from disk
Found 333 images belonging to 2 classes.
acuracia = 75.6756770315471%
jupyter-gabriel@greenery01:~$
```

Podemos verificar, utilizando o relatório inicial como base, que os objetivos propostos no início da disciplina foram cumpridos; são eles:

1. Estudo dos métodos de aprendizado modernos (identificando o mais adequado para a situação)
2. Estudo da arquitetura e as APIs do projeto INACITY
3. Implementação de algum dos métodos de aprendizado estudados
4. Executar experimentos para validar os métodos estudados
5. Relatório semanal

Os itens 1 até 4 foram cumpridos em todas as semanas, juntamente com discussões de projeto com outros membros do grupo. Foram feitas 5 reuniões presenciais para estudo da arquitetura do INACITY e implementação da rede neural no sistema, além de uma reunião de projeto com o supervisor, para criterizar a classificação das imagens. Cada discussão durou, em média, trinta minutos. Reuniões presenciais tiveram duração de 3 horas e 30 minutos, e a reunião com meu supervisor e o outros membros do grupo durou cerca de 1 hora e meia. A elaboração de cada relatório foi, em média, 40 minutos (nas semanas em que não foi feito nenhum progresso relevante, não houve relatório). O desenvolvimento da rede, que inclui desde o estudo da arquitetura, elaboração de testes e até a classificação e reclassificação das imagens, teve um mínimo de 5 horas e meia semanais (Nas semanas em que recebi mais imagens para classificar e testar, a carga horária foi de, no mínimo, 7 horas).

Supondo apenas 5 horas semanais de desenvolvimento (abaixo do que de fato foi usado), temos ~103 horas de carga horária para a disciplina MAC0215, satisfazendo o que foi pedido.