Nanodegree Engenheiro de Machine Learning

Proposta de projeto final

Arthur Sena 01/05/2018

Proposta

Histórico do assunto

Problemas que envolvam *Machine Learning* podem ser encontrados em diversas áreas. Uma das áreas mais antigas e que apresenta estudos bastante consolidados é o de reconhecimento de imagem. Ou seja, ensinar um determinado padrão de imagem pra um modelo, de modo que o mesmo consiga identificar esse exato padrão em imagens que ele ainda não tenha visto tem sido objeto de estudo dos especialistas da área já há algum tempo.

A resolução desse tipo de problema pode ser útil para diversos contextos, sendo que o âmbito de segurança, talvez, seja o mais conhecido. Em outras palavras, o desafio de reconhecer a face de pessoas que estejam autorizadas a acessar determinado local, dispositivo ou funcionalidade de algum sistema é bastante clássico. Contudo, empresas como Facebook, por exemplo, fazem uso de reconhecimento de imagem de uma maneira um pouco diferente. No caso, ele usam tal ideia para fazer marcação de fotos no perfil dos usuários da rede social.

É importante notar que o reconhecimento de imagens não se limita apenas à faces. Muito pelo contrário, podemos criar modelos pra reconhecer imanges de ambientes, animais, objetos, etc. Ou seja, qualquer forma que possa ser extraído um determinado padrão.

Descrição do problema

O desafio de reconhecer imagens é bastante amplo, visto que há infinitos padrões de imagens que podem ser aprendidos. Dessa forma, eu pretendo dirigir meus esforços na criação de um classificador de imagens de ambientes de três tipos:

```
        - Urbano: Imagens relacionadas com ambientes de cidades, préd
ios, casas, etc.
        - Natureza: Imagens relacionadas com ambientes de florestas,
matas, jardins, etc.
        - Praia/Litoral: Imagens realacionadas com praias, mares, are
ia, etc.
```

Uma vez contruído o classificador, eu pretendo avaliá-lo segundo as métricas de acurácia, precisão, recall e f-measure, e utilizando uma base de dados que tenha sido escondida dele na fase de treinamento, a fim de não obter resultados enganosos. Além disso, pretendo disponibilizar públicamente a base de imagens usada no aprendizado juntamente com as técnicas usadas para treinar o classificador.

Conjuntos de dados e entradas

O primeiro desafio para resolver esse problema é montar uma base de fotos dos ambientes a serem classificados. Para isso, pretendo utilizar a rede social de imagens mais famosa do mundo: *Instagram*. Nela existe diversos perfis com fotos dos ambientes que estamos interessados. Abaixo, segue o link de alguns perfis que podem ser úteis:

- Perfil-1 - Urbano
- Perfil-2 - Praias
- Perfil-3 - Natureza

Uma vez identificado os perfis mais úteis, realizarei o download de uma quantidade razoável de fotos de cada um dos ambientes. Acredito que algo entre 300 e 400 fotos para cada um dos ambientes deva ser suficiente. Além disso, programarei um script em python com o intuito de automatizar o download das fotos. Feito isso, será aplicado algum tipo de pré-processamento nas fotos, de forma que elas possam ser utilizadas como entrada na fase de treinamento do modelo. Por exemplo, caso decida por treinar uma rede neural, então penso em transformar as imagens numa matriz de *pixels* para que a mesma possa servir como entrada na rede.

Descrição da solução

A solução final para esse problema é um modelo que consegue distinguir, razoávelmente bem, imagens relacionadas a três tipos de ambientes: Urbano, Praias e Natureza. Aplicaremos técnicas de *machine learning*, a fim de encontrar o melhor modelo. Tal modelo será salvo e poderá ser utilizado para classificar novas imagens, de forma que caso tais imagens já foram previamente classificadas, então podemos usá-las pra coletar métricas de performance como acuŕacia e precisão. O alvo aqui é encontrar o melhor modelo, de forma que pretendo utilizar diferentes técnicas de aprendizagem, contudo pretendo focar meus esforços em algoritmos que envolvam redes neurais.

Modelo de referência (benchmark)

Resolvi utilizar o algoritmo KNN como benchmark, visto que o mesmo é simples e intuitivo de entender e aplicar. No caso, aplicarei tal algoritmo usando a base de dados construída a fim de obter as métricas de avaliação (Acurácia, precisão, revocação) e salvar as mesmas. Dessa forma, posso compará tais resultados com o meu modelo.

Métricas de avaliação

A fim de mensurar a performance dos modelos criados e identificar o melhor, irei aplicar as seguintes métricas de avaliação de machine learning:

- **Accurácia**: (True_negative + True_positive)/(Total_data)
- **Precisão**: (True_positive)/(True_positive + False_positive)
- **Revocação**: (True_positive)/(True_positive + False_negative)
- **F-measure**: (2*Revocação*Precisão)/(Precisão+Revocação)

É importante lembrar que como temos três classes pra classificar, então as métricas de precisão e revocação serão avaliadas por classes também. Tais métricas serão aplicadas usando dados de treino, teste e validação.

Design do projeto

O fluxo de trabalho desse projeto será divido nas seguintes etapas:

- **Etapa 1) Montagem de uma base de dados**:

1. Identificar perfis do *instagram* que contenham fotos públicas dos ambientes que quero classificar.
2. Fazer um script em Python que automatize o download dessas fotos.
3. Executar o script e salvar as fotos.

- **Etapa 2) Pré-processamento dos dados**:

1. Identificar possíveis técnicas de processamento de imagens que possam ser utilizadas para discriminar melhor as classes.
2. Aplicar tais técnicas.

- **Etapa 3) Treinamento e Avaliação do modelo**

1. Dividir minha base de dados em treino, teste e validação.
2. Adequar os dados para o formato que o modelo espera receber.
3. Possilvemente, aplicar algoritmos de clusterização e/ou árvores de decisão com o intuito de verificar se as features discriminam bem os dados.
4. Treinar o modelo. No caso, o modelo de benchmark, também, está incluso nessa etapa.
5. Mensurar a performance do modelo usando as bases de treino, teste e validação.
6. Construir a matrix de confusão para uma melhor observação dos resultados.

- **Etapa 4) Refinamento do modelo**

1. Utilizar estratégias de model-tuning, como Cross-validation, a fim de escolher os melhores parâmetros e diminuir a chance de overfitting.

- **Etapa 5) Conclusão**

1. Documentar e tornar público todos os resultados obtidos, de forma a tornar todo o experimento aqui realizado replicável.

Segue anexado um simples *workflow* para melhor representar as etapas acima. Repare que existe uma seta bidirecional entre as etapas de pré-processo de dados e avaliação de modelo, pois dependendo do resultado da avaliação, talvez seja preciso usar uma estratégia diferente de pré-processamento.