Análise de classificadores

Aryane Ast dos Santos
Departamento de Informática
Universidade Federal do Paraná
Email: aras10@inf.ufpr.br

I. Introdução

Um problema de classificação consiste em definir um rótulo ou classe para um elemento a partir de um conjunto de elementos com rótulos definidos. É um problema de aprendizagem supervisionada, cujo objetivo é realizar inferências a partir de um conjunto de dados rotulados, em oposição à aprendizagem não-supervisionada.

Este relatório se propõe a apresentar resultados obtidos com os classificadores *K Nearest Neighbors* (KNN), Árvores de Decisão e *Support Vector Machines* (SVM) para um problema de classificação de imagens, cuja base rotulada possui 1901 imagens divididas 9 classes diferentes. Os algoritmos de classificação não utilizam as imagens "brutas", sendo necessário, então, converter as imagens do formato JPG para vetores de características que os algoritmos de classificação possam utilizar.

Após extraído os vetores de características das imagens, foram realizadas as execuções dos classificadores KNN, Árvores de Decisão e SVM. As implementações dos algoritmos mencionados são da biblioteca Scikit Learn (ref).

Nas seções a seguir são apresentados maiores detalhes da representação, algoritmos utilizados, métricas para comparação e desempenho. São comparados também o desempenho de estratégias de combinação de classificadores e *ensembles*.

II. Representação dos dados

Para cada uma das imagens disponibilizadas para classificação, é realizada uma extração de características, que resulta num vetor com as características

Para a extração dos vetores de caracteristicas, foram utilizados os algoritmos *Local Binary Patterns* (LBP) e *Grey-Level Co-Occurrence Matrix* (GLCM), o que resultou em vetor contendo 24 características, além da classe ao final da linha.

A. Local Binary Patterns

O LBP (Local Binary Patterns) baseia-se no fato de que certos padrões locais à região de vizinhança de um pixeil são propriedades fundamentais da textura de uma imagem.

o método uniforme, com raio 2 e n_point ou vizinhos igual a 16 Método uniforme, raio=2, n_point ou vizinhos = 16, implementação do scikit learn.

B. Grey-Level Co-Occurrence Matrix

A matriz de co-ocorrência utiliza informações sobre a posição relativa dos pixels em relação uns aos outros. Foram utilizadas as características de correlação, dissimilaridade, contrast, homogeneidade, energia, e ASM.

III. Classificação

A partir dos vetores de características, é possível executar os algoritmos de classificação. Como temos apenas uma base de dados, se a utilizarmos inteira para treinar os algoritmos e após isso, testar se a classificação é feita corretamente com essa mesma base, ocorrerá algo chamado de *overfitting*, que ocorre quando a base é muito especializada e acerta predições para um conjunto de dados conhecido, mas para dados desconhecidos costuma errar. Para fugir dessa situação, é boa prática separar a base em treinamento e validação.

Entretanto, ao separar a base em treinamento e validação, reduz-se muito a quantidade de dados dos quais se aprende (dados treinamento). Para evitar tal situação, se faz uso de uma técnica chamada validação cruzada ou *cross-validation*, onde se separa ...

Neste trabalho, para a validação cruzada são utilizados os métodos ShuffleSplit e cross_val_score do módulo model_selection da biblioteca SciKit Learn. Dessa forma, a base é dividida 10 vezes em treinamento e validação nas proporções de 0.6 e 0.4 respectivamente.

A. Métricas

Precisão é a abilidade de um classificador não rotular com positivo uma amostra que é negativa. Recall é a abilidade do classificador de encontrar todas as amostras positivas. Já a métrica F-measure podem ser interpretadas como médias harmônicas da precisão e recall.

B. KNN

O KNN (K-Nearest Neighbors) classifica um dado x atribuindo a ele o rótulo representado mais frequentemente dentre as k amostras mais próximas. O algoritmo recebe apenas um parâmetro: o inteiro k. Variando k de 3 a 30, foi possível perceber que o k que propocionou melhor média de acurácia dentre os 10 folds de validação cruzada foi 5. A média de acurácia foi de 0,51 com margem de erro de 0,01.

C. Árvores de decisão

Em um classificador de Árvore de decisão, o objetivo é criar um modelo que prediz o valor de variáveis a partir da aprendizagem de regras de decisão inferidas dos dados. Em cada nó, é representado um atributo, que implica numa decisão. Cada ramo corresponde a um possível valor deste

atributo. Cada folha está associada à uma classe e os percursos na árvore é uma regra de classificação.

Árvores de decisão tem a vantagem de serem simmples de entender e visualizar. Por outro lado, é possível que seja gerada uma árvore complexa e especialista, o que leva à overfitting. Definindo uma profundidade máxima para a árvore e um limite para características utilizadas, é possível contornar este problema. Neste trabalho, foi adotado a profundidade máxima de 10, e como o número de características é baixo, não se fez necessário limitá-lo.

D. SVM

O classificador do SVM (Support Vector Machines) encontra um hiperplano de separação para dados de duas classes distantas. Busca-se maximizar a distância entre o hiperplano e os dados de treinamento, e à essa distância é dado o nome margem.

Apesar de o SVM ser um classificador linear binário, a maioria dos problemas não possuem apenas duas classes nem são linearmente separáveis, seja pela ocorrência de outliers, mas na maioria dos casos é pela própria distribuição dos dados.

Ainda assim, o SVM se mostra apropriado para ser utilizado em tais casos. Com o Kernel Trick, é possível projetar os dados em um espaço onde eles são linearmente separáveis. E para resolver o problema de várias classes, existe a estratégia de um-contra-todos (one-versus-rest), onde se n é o número de classes, são treinados n classificadores que utilizam os dados de uma das classes contra os dados de todas as outras juntas, obtendo assim n classificadores lineares.

IV. Ensembles

A. Random forests

Random forests funcionam como uma coleçao de árvores de decisão não relacionadas entre si.

Possui dois parâmetros principais, o número de estimadores n_estimators e número máximo de features max_features. O número de estimadores define a quantidade de árvores de decisão da floresta. Intuitivamente, quando mais árvores, melhor o resultado, apesar de levar mais tempo para executar o algoritmo. Porém, ao executar o classificar para números de estimadores variando entre 10 e 100, foi possível observar que a média de acurácia, pois se trabalhou com validação cruzada, foi de 0,79, com desvio padrão de 0,01 para números de estimadores a partir de 59 até 100. E como o algoritmo roda muito mais rápido com um número menor de árvores, 59 foi o n_estimatores escolhido.

Já o parâmetro max_features se refere à quantidade de características utilizadas. De acordo com a documentação do SciKit-Learn, max_features como raiz quadrada do número de características gera bons resultados, que neste caso seria próximo de 5. Obtive as melhores médias de acurácia para max_depth variando de 5 a 19.

V. Resultados

A árvore de decisão, sem limite de características, uma vez que são apenas 24, e com limite de profundidade 10, produziu 10 scores da validação cruzada, exibidos na figura 1, cuja média da acurácia foi 0.68 e desvio padrão de 0.007467.

Figura 1. Árvore de decisão

```
Scores:         0.67838506      0.6896457
0.66273002      0.66712442      0.6858006
0.67673716      0.67206811      0.67481461
0.67948366      0.67508926

Acurácia média: 0.68 (+/- 0.02)
Desvio padrão: 0.007467
```

Já para o KNN, a acurácia média foi de 0.51 e desvio padrão deDesvio padrão: 0.006410, como pode ser visto na figura 2.

Figura 2. KNN

```
Scores:         0.49903873      0.50755287
0.51167262      0.49189783      0.51139797
0.51139797      0.50892612      0.51304587
0.50425707      0.51029937

Acurácia média: 0.51 (+/- 0.01)
Desvio padrão: 0.006410
```

Random forests com acurácia média de 0.79 com desvio padrão de 0.007549 na figura 3, para 59 estimadores (árvores) e limite de características de 5.

Figura 3. Random Forests

```
Scores:         0.78989289      média: 0.79
0.7761604       0.78934359      (+/- 0.01)
0.79318868      0.78742104      Desvio
0.78714639      0.78687174      padrão:
0.78028014      0.79511123      0.007549
0.77808294      Acurácia
```

SVM com kernel linear, gama=2, C=46, estratégia multi-classe de um-contra-todos e classes balanceadas, resultou em acurácia média de 0.73 com desvio padrão de 0.005181, pode ser visto na figura 4.

Figura 4. SVM Linear

```
Scores:         0.72946993      0.73139247
0.73606152      0.72589948      0.73523757
0.73578687      0.73358967      0.73853337
0.73551222      0.74622356

Acurácia média: 0.73 (+/- 0.01)
Desvio padrão: 0.005181
```

Por fim, juntei os dois métodos com melhores resultados, SVM e Random Forests, utilizando o classificador VotingClassifier. Os resultados podem ser observados na figura 5.

Figura 5. Voting Classifier

```
Scores:         0.74237847      0.75556166
0.75254051      0.74045592      0.74897006
0.75336446      0.75583631      0.76215325
0.75720956      0.76599835

Acurácia média: 0.75 (+/- 0.02)
```

Ao mesclar os classificadores, no caso apresentado, a média de acurácia, ao invés de melhorar, ficou próxima da média entre os valores médios do SVM e Random Forests. Assim sendo, é preferível ficar apenas com o método de Random Forests.

Do KNN às Random Forests, obteve-se uma melhora sensível de desempenho. Ainda assim, um erro de 0.21 é muito alto. Atribuo isso à representação, uma vez que se ela não for boa o suficiente, ou seja, bastante similar para objetos de uma mesma classe e bastante diferente para objetos de classes distintas, não muito que os classificadores possam fazer.