Reconhecimento de caracteres manuscritos baseados em números

Marcos Antonio da Silva, Elias José
Irismar N. de Souza

Departamento de Computação – Universidade Federal Rural de Pernambuco (UFRPE)
52.171-900 – Recife – PE – Brazil

Abstrato. This article refers to the project for the evaluation of the discipline of Artificial Intelligence, which deals with the classification of numbers from a base of handwritten images.
For the classification we used the algorithm KNN Three different metrics

Resumo. Este artigo é referente ao projeto para avaliação da disciplina de Inteligência Artificial, que aborda a classificação de números provenientes de uma base de imagens manuscritas. Para a classificação utilizamos o algoritmo KNN e três métricas diferentes.

1. Introdução Este projeto se trata da execução de um algoritmo para treinamento e depois classificar imagens de números manuscritos, utilizando uma base de dados com dez mil conjuntos A fim de tratar esse problema, foi escolhido o algoritmo K-Nearest Neighbor (KNN), pois é um algoritmo simples e com uma acurácia relativamente alta, fazendo com que seja muito útil em classificação de objetos. A base de dados foi convertida para arquivo .csv com cada linha representando um número, como a base de dados contém 10.000 elementos, um arquivo de 10.000 linhas foi gerado. Cada linha possui 784 atributos, pois a imagem possui 28x28 pixels, então cada atributo corresponde à um pixel. Os atributos possuem valores de 0 à 255, que se refere ao tons de preto, 0 é um pixel branco e 255 um pixel totalmente preto. Cada linha foi rotulada como um número de 0 à 9, que são as classes do problema tratado. Desenvolvemos 2 programas, um para classificar a base de dados com 10.000 elementos e outro para classificar a base de dados com 1.000 elementos. Os parâmetros utilizados para variação do algoritmo, foi o número K, que foi testado com o valor 3, 5 e 19, porém o K foi variado apenas no programa que classifica dados com 1.000 elementos, por conta do alto custo computacional. A conversão da base de dados para .csv, foi utilizada um programa escrito na linguagem Python. Base de dados utilizada foi a t10k-images.idx3-ubyte. Neste projeto foi utilizado a ferramenta WEKA para converter o arquivo .csv para .arff

2. Pré-processamento

- Classificar uma base de dados, que contém imagens de caracteres de 0 até 9.
- As imagens são de tamanho 28x28 pixels
- Imagens em tons de preto
- Imagens binárias (preto e branco)

3. Especificações

- Foi utilizada uma base de dados do The MNIST database of handwritten digits.
- A base de dados foi dividida em ⅔ para treinamento e ⅓ para teste
- A conversão das imagens foi feita usando um programa escrito em Python e salva como um vetor de 784 posições.
- Notepad++ foi utilizado para facilitar a rotular cada linha com a sua classe especificada.

4. Implementação dos algoritmos

4.1. Algoritmo KNN

Este um dos algoritmos classificação mais simples. Utilizado para classificar objetos com base em exemplos de treinamento que estão mais próximos no espaço de características. Para utilizar o KNN é preciso de um conjunto de exemplos de treinamento, definir uma métrica para calcular a distância entre os exemplos de treinamento e definir o valor de k (número de vizinhos mais próximos que serão considerados pelo algoritmo).

4.2Pseudocódigo do KNN

```
classificar(x, y, k) // x: conjunto treinamento, y: rótulo das classes
para i = 0 até n faça:
calcular distância D(x, k)
calcular o conjunto que contém as menores distâncias D(x, k)
retorna rótulo do elemento
```

4.3 Pseudocódigo da distância Euclidiana

```
distanciaEuclidiana(instancia1, instancia2, tamanho)
para i = 0 até tamanho faça:
distancia += (instancia1[i] - instancia2[i])²
fim
retorna sqrt(distancia)
```

4.4 Pseudocódigo da distância de Manhattan

```
distanciaManhattan(instancia1, instancia2, tamanho)
para i = 0 até tamanho faça:
distancia += mod(instancia1[i] - instancia2[i])
fim
retorna distancia
```

4.5 Pseudocódigo da distância de Bray-Curtis

```
distanciaBrayCurtis(instancia1, instancia2, tamanho)
para i = 0 até tamanho faça:
distancia += mod(instancia1[i] - instancia2[i])
distancia2 += mod(instancia1[i] + instancia2[i])
fim
retorna distancia/distancia2
```

5. Avaliação do algoritmo

O algoritmo KNN foi avaliado utilizando 3 parâmetros para o K, que foram 3, 5 e 19. Para calcular a taxa de acertos foi usada 3 métricas diferentes, a distância Euclidiana, distância de Manhattan e distância de Bray-Curis. 5.1. Resultados

Utilizando a base de dados com 10.000 elementos, com o parâmetro k igual à 3, a taxa de acertos da distância Euclidiana ficou muito próximo ao resultado obtido usando o programa Weka, com a mesma base de dados, porém as outras métricas também tiveram uma acurácia satisfatória.

6.Conclusão

Podemos verificar que o algoritmo KNN é bom algoritmo para classificação de objetos, devido à sua simples implementação e uma taxa de acerto relativamente alta, porém um alto custo computacional é exigido, pois o algoritmo guarda na memória todo conjunto de treinamento. As 3 métricas utilizadas, a que teve a menor acurácia foi a distância de Manhattan, devido que é uma simplificação da distância Euclidiana, porém devido à sua simplicidade, teve o melhor desempenho. A que teve a melhor acurácia foi a distância de Bray-Curis, juntamente com um desempenho satisfatório. Distância Euclidiana obteve uma taxa de acertos alta, porém com o pior desempenho entre as três métricas, devido que esta métrica precisa realizar mais cálculos matemáticos que as outras duas.

7.Referência

https://inferir.com.br/artigos/algoritimo-knn-para-classificacao/
pesquisado em 05/07/2019.
https://gist.github.com/marcoscastro/d71db621b83d2aac6be441c231f414c5
pesquisado em 14/07/2019
http://edirlei.3dgb.com.br/aulas/ia_2012_1/IA_Aula_16_KNN.pdf
pesquisado em 15/07/2019
http://computacaointeligente.com.br/algoritmos/k-vizinhos-mais-proximos/ pesquisado em 15/07/2019