Relatório TA4 – Visão Computacional

Gabriel Henrique Oliveira de Mello – GRR20190602
Universidade Federal do Paraná – UFPR

Resumo
Este relatório detalha o experimento de implementação de um modelo de rede neural usando a biblioteca PyTorch e para classificar as espécies de íris a partir do dataset Iris junto com a comparação com a classificação utilizando KNN. O dataset Iris, um conjunto de dados amplamente usado na literatura de aprendizado de máquina, consiste em 150 amostras de íris de três espécies diferentes (setosa, versicolor, virginica), com 50 amostras de cada. Cada amostra tem quatro atributos: comprimento da sépala, largura da sépala, comprimento da pétala e largura da pétala. O objetivo foi treinar um modelo de aprendizado de máquina para prever corretamente a espécie de uma íris com base nesses quatro atributos e compararar com o resultado da classificação utilizando KNN. O Jupyter Notebook está disponível no GitHub: https://github.com/gabrielhom/TA4-Visao-Computacional/blob/main/ta4.ipynb.

Sumário

1 Introdução

O presente relatório delineia uma experiência de aprendizado de máquina em que comparamos o desempenho de um modelo de Rede Neural construído com a biblioteca PyTorch com um algoritmo k-Nearest Neighbors (KNN) do Scikit-Learn. Ambos os modelos foram treinados e testados no conhecido dataset Iris, um conjunto de dados padrão na área de aprendizado de máquina.

O dataset Iris contém 150 amostras de três espécies de flores de íris (setosa, versicolor, virginica), cada uma com 50 amostras. Cada amostra possui quatro características distintas: comprimento da sépala, largura da sépala, comprimento da pétala e largura da pétala. O objetivo do experimento foi treinar ambos os modelos para prever corretamente a espécie da flor de íris a partir dessas quatro características.

O experimento foi dividido em duas partes principais. Na primeira parte, utilizamos a biblioteca PyTorch para construir e treinar uma Rede Neural. O modelo foi estruturado com uma camada de entrada correspondente às quatro características das flores, duas camadas ocultas e uma camada de saída correspondente às três espécies de íris.

Na segunda parte, utilizamos o algoritmo KNN usando a biblioteca Scikit-Learn. Esse algoritmo classifica uma amostra com base na maioria das classes de seus vizinhos mais próximos.

O intuito deste relatório é apresentar os resultados obtidos, comparar o desempenho de ambos os modelos e discutir as vantagens e desvantagens de cada método para a tarefa de classificação proposta.

2 O Dataset Iris

O dataset Iris é um dos datasets mais famosos e acessíveis na área de aprendizado de máquina. Foi introduzido pelo estatístico e biólogo britânico Ronald Fisher em 1936 em seu artigo "The use of multiple measurements in taxonomic problems" como um exemplo de análise discriminante linear.

O dataset Iris é composto por 150 observações de íris, uma flor que tem variedades distintas. As amostras pertencem a uma de três espécies de íris: Iris setosa, Iris versicolor e Iris virginica. Cada espécie é representada por 50 observações, tornando o conjunto de dados balanceado.

Cada observação no conjunto de dados consiste em quatro medidas (em cm) dessas flores:

1. Comprimento da sépala
2. Largura da sépala
3. Comprimento da pétala
4. Largura da pétala

Essas quatro características constituem a base do aprendizado de máquina para este conjunto de dados. A tarefa consiste em prever a espécie da íris com base nessas medidas.

O dataset Iris é adequado para classificação e clusterização. Em termos de classificação, a tarefa é geralmente definida como um problema de classificação multiclasse, onde o objetivo é prever uma das três espécies possíveis.

Este conjunto de dados tem sido extremamente influente, pois é pequeno, não possui valores faltantes, e exige pouca preparação. Além disso, apesar de sua simplicidade, pode-se explorar diferentes técnicas de aprendizado de máquina e pré-processamento. Dada a sua natureza multiclasse, é ideal para técnicas de classificação e avaliação, e também é adequado para visualização de dados e técnicas de redução de dimensionalidade.

No experimento, utilizamos o dataset Iris para treinar e testar um modelo de Rede Neural usando a biblioteca PyTorch e um algoritmo KNN do Scikit-Learn, comparando posteriormente o desempenho de ambos os modelos.

3 Os Modelos

3.1 Rede Neural

A rede neural construída para este experimento é uma estrutura composta por várias camadas de nós, ou neurônios, cada um dos quais realiza cálculos simples em seus dados de entrada. Cada neurônio leva múltiplas entradas, multiplica cada entrada por um peso específico, soma todas as entradas ponderadas e, em seguida, aplica uma função de ativação ao resultado para produzir sua saída. Estas saídas então se tornam entradas para os neurônios na próxima camada da rede.

A rede neural é composta de três tipos de camadas: entrada, oculta e saída.

Camada de entrada: É a primeira camada da rede e recebe dados brutos como entrada. Para o nosso modelo, a camada de entrada possui quatro neurônios, correspondendo aos quatro atributos de cada amostra no dataset Iris (comprimento da sépala, largura da sépala, comprimento da pétala, largura da pétala).

Camadas ocultas: São camadas intermediárias entre a entrada e a saída. Elas recebem as entradas da camada anterior e passam suas saídas para a próxima camada. Elas são chamadas de "ocultas" porque não interagem diretamente com o mundo externo.

Para o nosso modelo, utilizamos duas camadas ocultas. A quantidade de neurônios nessas camadas pode variar dependendo do design da rede e a complexidade do problema a ser resolvido.

Camada de saída: É a última camada da rede. Ela recebe entradas das camadas ocultas e produz a saída final da rede. Para o nosso modelo, a camada de saída tem três neurônios, correspondendo às três espécies de íris que queremos prever (setosa, versicolor, virginica).

Esse tipo de modelo de rede neural, com múltiplas camadas de neurônios e retropropagação, é conhecido como uma rede neural multicamadas (Multilayer Perceptron, ou MLP). São modelos poderosos que podem aprender representações complexas dos dados.

3.2 KNN (K-Nearest Neighbors)

O k-Nearest Neighbors (KNN) é um algoritmo de aprendizado de máquina simples e intuitivo que é amplamente usado em classificação e regressão. É um algoritmo do tipo "lazy learner", pois não aprende uma função discriminativa a partir do conjunto de treinamento, mas memoriza o conjunto de treinamento em vez disso.

A ideia principal por trás do KNN é que os exemplos de treinamento são distribuídos em um espaço n-dimensional, onde n é o número de características no dataset. Quando recebemos um exemplo não rotulado para classificação (ou regressão), o algoritmo KNN identifica os 'k' exemplos de treinamento mais próximos a este exemplo não rotulado. A "proximidade" é normalmente medida usando uma métrica de distância, como a distância euclidiana ou a distância de Manhattan.

Na classificação, o KNN atribui ao exemplo não rotulado a classe mais comum entre seus vizinhos mais próximos.

O número 'k' de vizinhos considerados é um hiperparâmetro importante para o algoritmo KNN e precisa ser cuidadosamente escolhido. Um 'k' muito pequeno torna o algoritmo sensível a exemplos de treinamento ruidosos, enquanto um 'k' muito grande torna o algoritmo insensível às variações locais dos exemplos de treinamento.

No nosso experimento, utilizamos o algoritmo KNN do Scikit-Learn para classificar as espécies de íris a partir do dataset Iris com k = 3.

4 Treinamento e desempenho dos modelos

4.1 Treinamento da Rede Neural

Para o treinamento da rede neural, foram utilizadas 100 épocas. A cada época, o modelo é treinado com o conjunto de treinamento e testado com o conjunto de teste. O erro de treinamento e o erro de teste são calculados e armazenados para cada época. Ao final do treinamento, os erros de treinamento e teste são plotados em um gráfico para visualização.

4.2 Desempenho dos modelos

O desempenho dos modelos foi avaliado com base na acurácia, que é a proporção de amostras corretamente classificadas. O resultado dos dois é muito próximo, com a Rede Neural tendo uma acurácia média de 0.97 e o KNN de 0.96. Houve algumas variações na acurácia dos modelos em diferentes execuções do Notebook, acredito que pela divisão dos conjuntos de teste/treinamento.

5 Conclusão

O desempenho dos dois classificadores é muito próximo, demonstrando que para o dataset Iris, o KNN é uma alternativa viável à Rede Neural. Provavelmente seja por questão do tamanho do dataset e quantidade de características.

Essa simplicidade do dataset, faz uma Rede Neural, com tensores, treinamento em GPU, etc. ser um overkill. A dificuldade da implementação supera muito o ganho marginal de desempenho.

O meu foco no experimento foi aprender a utilizar o PyTorch para classificação, pois só possuia experiência com o TensorFlow. Infelizmente a implementação em GPU não foi um sucesso e acabei perdendo tempo demais nisso. Novamente, o foco estava no aprendizado pois o treinamento em CPU já é muito rápido. Meu ambiente utilizando WSL2 e bypass de GPU para o treinamento não é o ideal.

Apesar desses desafios, o experimento provou ser um exercício valioso de aprendizado. Aprendi que, embora as Redes Neurais possam ser poderosas ferramentas de aprendizado de máquina, elas não são sempre a melhor ou mais eficiente escolha, especialmente para conjuntos de dados menores e mais simples. Ao mesmo tempo, ganhei experiência valiosa com o PyTorch, que certamente será útil para futuros projetos de aprendizado de máquina mais complexos.