Vision Challenge

Anna Caroline Bozzi[1], **Pamella A. de L. Mariano**[1]

[1]Departamento de Informática – Universidade Federal do Paraná (UFPR)
Curitiba – PR – Brazil

Abstract. *This article addresses the application of the K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) classification methods in the task of classifying the popular CIFAR-10 database, which contains images from 10 different classes. The main objective of this study is to compare the performance of these two machine learning algorithms regarding classification accuracy, considering variations in the parameters of each method. Initially, we describe the CIFAR-10 database, which consists of a set of 60,000 color images divided equally into 10 categories, such as cars, planes, cats, dogs, among others. Next, we present a brief overview of the KNN and SVM algorithms, highlighting their characteristics and functioning.*

Resumo. *Este artigo aborda a aplicação dos métodos de classificação K-Nearest Neighbors (KNN) e Support Vector Machine (SVM) na tarefa de classificar a popular base de dados CIFAR-10, que contém imagens de 10 classes distintas. O objetivo principal deste estudo é comparar o desempenho desses dois algoritmos de aprendizado de máquina em relação à precisão de classificação, considerando variações nos parâmetros de cada método. Inicialmente, descrevemos a base de dados CIFAR-10, que consiste em um conjunto de 60.000 imagens coloridas divididas igualmente em 10 categorias, como carros, aviões, gatos, cachorros, entre outros. Em seguida, apresentamos uma breve visão geral dos algoritmos KNN e SVM, destacando suas características e funcionamento.*

1. Introdução

A aplicação eficaz de métodos de classificação é uma questão crítica na área de aprendizado de máquina e análise de dados. No contexto da classificação de imagens, a escolha dos algoritmos e a otimização de seus parâmetros desempenham um papel fundamental na obtenção de resultados precisos e confiáveis. Neste trabalho, voltamos nossa atenção para dois métodos amplamente reconhecidos: o K-Nearest Neighbors (KNN) e o Support Vector Machine (SVM), com o intuito de explorar sua aplicabilidade na classificação de uma das bases de dados mais desafiadoras e relevantes na área, a CIFAR-10.

A base de dados CIFAR-10, composta por 60.000 imagens coloridas distribuídas em 10 classes distintas, oferece um ambiente realista e diversificado para a avaliação do desempenho de algoritmos de classificação. Essas classes abrangem uma ampla variedade de objetos e animais, incluindo carros, aviões, gatos, cachorros, entre outros. A complexidade inerente a essa base de dados torna-a um campo de testes ideal para métodos de aprendizado de máquina.

O objetivo principal deste estudo é comparar o desempenho dos métodos KNN e SVM em relação à precisão de classificação das imagens da CIFAR-10. Esta comparação

será conduzida considerando a variação dos parâmetros essenciais de cada algoritmo, possibilitando avaliar como diferentes configurações afetam a capacidade do classificador.

Neste contexto, este trabalho inicia-se com uma descrição completa da base CIFAR-10. Em seguida, forneceremos uma visão geral dos métodos KNN e SVM, explicando suas fundações teóricas e destacando suas peculiaridades. Posteriormente, abordaremos a metodologia experimental, descrevendo como visamos realizar o trabalho proposto.

2. CIFAR-10

O CIFAR-10 é um conjunto de dados multiclasse que inclui 60.000 imagens coloridas com dimensões de 32×32 pixels, distribuídas em 10 categorias distintas, cada uma contendo 6.000 imagens. Este conjunto de dados é dividido em 50.000 imagens para treinamento e 10.000 imagens para teste [Abouelnaga et al. 2016].

As classes são divididas em:

- Avião;
- Carro;
- Pássaro;
- Gato;
- Veado;
- Cão;
- Sapo;
- Cavalo;
- Barco;
- Caminhão.

O desafio apresentado pelo CIFAR-10 é notável, uma vez que imagens de 32×32 pixels frequentemente não contêm informações suficientes para permitir que a maioria dos classificadores estabeleça limites de decisão nítidos. Isso se reflete na dificuldade de distinguir algumas classes, como as de ”gato” e ”cachorro”. As imagens também variam amplamente em termos de escala, rotação, posição e fundo, tornando a tarefa de classificação ainda mais complexa. Além disso, algumas imagens são de qualidade inferior e desafiadoras até mesmo para observadores humanos [Abouelnaga et al. 2016].

3. K-Nearest Neighbors (KNN)

O K-Nearest Neighbors (KNN) é um algoritmo de aprendizado de máquina usado para classificação e regressão. Funciona com base no princípio de que objetos semelhantes tendem a estar próximos uns dos outros em um espaço de características [Didática Tech 2020].

O algoritmo funciona, brevemente, da seguinte maneira [Didática Tech 2020]:

1. Coleta de Dados de Treinamento:
   Primeiramente, um conjunto de dados de treinamento é fornecido. Cada exemplo nesse conjunto possui um rótulo de classe ou um valor de destino associado a ele, juntamente com um vetor de características que descreve suas propriedades.

2. Escolha do Parâmetro K:
   O usuário especifica um parâmetro K, que determina quantos vizinhos mais próximos serão considerados ao fazer uma previsão. A escolha de K é crucial, pois afeta diretamente a qualidade das previsões do algoritmo.
3. Cálculo da Distância:
   Quando uma nova amostra de teste é apresentada ao algoritmo, o KNN calcula a distância entre essa amostra e todos os exemplos do conjunto de treinamento. A distância pode ser calculada usando métricas como a distância euclidiana.
4. Seleção dos K Vizinhos mais Próximos:
   Com base nas distâncias calculadas, o KNN seleciona os K vizinhos mais próximos da amostra de teste. Esses vizinhos são aqueles com as menores distâncias em relação à amostra de teste.
5. Votação ou Média Ponderada:
   Para problemas de classificação, o KNN realiza uma votação entre os K vizinhos para determinar a classe da amostra de teste. A classe mais frequente entre os vizinhos é atribuída à amostra de teste. Para problemas de regressão, o KNN calcula a média ponderada dos valores de destino dos vizinhos para prever o valor da amostra de teste.
6. Previsão Final:
   A classe ou valor previsto é atribuído à amostra de teste como sua classificação ou valor alvo.

4. Support Vector Machine (SVM)

O algoritmo SVM, ou Support Vector Machine, é um poderoso método de aprendizado de máquina usado tanto para classificação quanto para regressão. Sua principal característica é a capacidade de encontrar hiperplanos de separação que maximizam a margem entre classes em um espaço de características. Aqui está uma descrição do algoritmo SVM [Coutinho 2019].

O algoritmo SVM é uma técnica de aprendizado de máquina que opera da seguinte maneira [Coutinho 2019]:

1. Coleta de Dados de Treinamento:
   Começa com um conjunto de dados de treinamento, onde cada exemplo é associado a uma classe (para classificação) ou a um valor alvo (para regressão). Cada exemplo é representado por um vetor de características que descreve suas propriedades.
2. Escolha do Hiperplano de Separação:
   O objetivo do SVM é encontrar o hiperplano de separação que melhor divide as classes no espaço de características. Esse hiperplano é escolhido de forma a maximizar a margem, que é a distância entre o hiperplano e os exemplos de treinamento mais próximos, chamados vetores de suporte.
3. Mapeamento de Dados em Espaço de Maior Dimensão (Opcional):
   Em muitos casos, os dados não são linearmente separáveis em seu espaço de características original. Nesses casos, o SVM pode mapear os dados para um espaço de maior dimensão usando funções de kernel, como o kernel polinomial ou o kernel radial (RBF). Isso permite que o SVM encontre hiperplanos não lineares para separação.

4. Treinamento:
   Durante o treinamento, o SVM otimiza os parâmetros do hiperplano de separação, de forma que a margem seja maximizada e o erro de classificação seja minimizado.
5. Classificação/Regressão:
   Após o treinamento, o SVM pode ser usado para classificar novos exemplos (no caso de classificação) ou fazer previsões (no caso de regressão). Ele atribui classes ou valores com base em qual lado do hiperplano as amostras de teste caem.

5. Metodologia

Para a realização do trabalho intermediário da disciplina, decidimos adotar uma abordagem que envolve o uso de dois métodos de classificação amplamente reconhecidos: *K-Nearest Neighbors (KNN) e Support Vector Machine (SVM)*. Nosso objetivo central é aplicar esses algoritmos à base de dados conhecida como CIFAR-10. Utilizaremos ainda a biblioteca o *Scikit-learn*, que é uma biblioteca de aprendizado de máquina de código aberto que oferece suporte ao aprendizado supervisionado e não supervisionado. Ele também fornece várias ferramentas para ajuste de modelo, pré-processamento de dados, seleção de modelo, avaliação de modelo e muitos outros utilitários. E ainda foi utilizado o *Colab*, ou *Colaboratory*, que permite escrever e executar *Python* no navegador e conta com acesso a GPUs sem custo financeiro além de ter um compartilhamento fácil e não requer nenhuma configuração específica.

6. Desenvolvimento

A implementação foi feita tomando o cuidado de mantê-la modularizada a título de entendimento. Portanto ela é composta das seguintes funções auxiliares:

- *load_batch* : Essa função é usada para carregar um único lote (batch) do conjunto de dados CIFAR-10. Ela lê um arquivo binário, onde estão os dados e os rótulos de classe das imagens.
- *Obter_Todos_Batch* : Esta função é usada para carregar todos os lotes do conjunto de dados CIFAR-10. Ela chama a função load_batch para cada lote e concatena os dados e rótulos de classe de todos os lotes em arrays.
- *Calcular_Best_K_OPT* : Essa função calcula o número ótimo de componentes principais (K) a serem mantidos para explicar uma porcentagem significativa da variância nos dados.
- *Calcular_Best_K* : Esta função calcula o valor de K a ser usado com base em um critério de manter 99% da variância explicada. O valor de K é usado na próxima etapa.

Durante o desenvolvimento sentimos a necessidade de usar um método chamado de Análise de Componentes Principais - PCA [1]. Utilizado para analisar grandes conjuntos de dados contendo um elevado número de dimensões/recursos, aumentando a interpretabilidade dos dados enquanto preserva a quantidade máxima de informações e permite a visualização de dados multidimensionais. Formalmente, PCA é uma técnica estatística para reduzir a dimensionalidade de um conjunto de dados. Em palavras simples, a técnica PCA foi utilizada para calcular o número de componentes do modelo. Além disso, buscou-se um k ótimo para que houvesse uma variância aproximada de 99%, sendo esse valor 658.

[1] https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

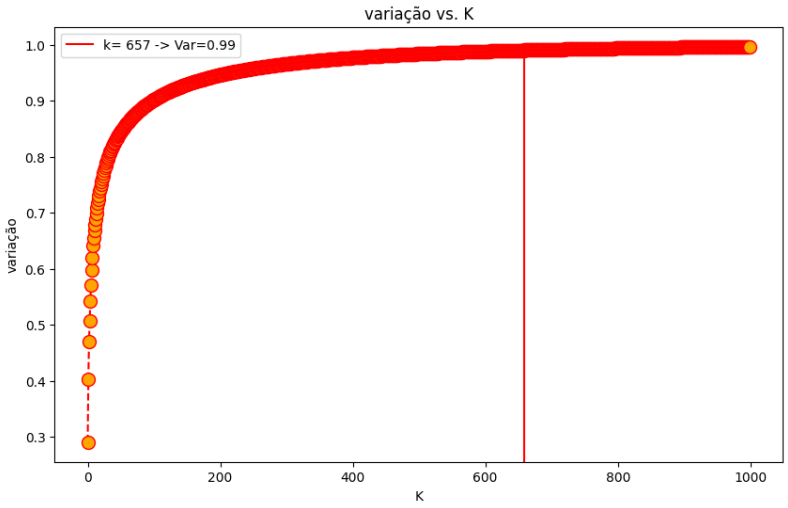

Figure 1. Imagem do gráfico que representa a normalização PCA

Assim que foi feito o carregamento dos dados e a aplicação do PCA, seguimos para o treinamento e teste com KNN utilizando os dados de treinamento após a redução de dimensionalidade com PCA. O modelo KNN é então testado com os dados de teste. A precisão do modelo KNN é calculada e métricas como matriz de confusão e relatório de classificação são exibidas, e serão apresentadas na seção de Resultados.

O outro classificador usado foi o SVM, foi treinado da mesma maneira que o KNN com os dados de treinamento após a redução de dimensionalidade. O modelo SVM foi testado com os dados de teste, e a precisão do modelo SVM calculada.

7. Resultados

7.1. KNN

A acurária do modelo KNN foi de 0.1448. O KNN obteve uma acurácia geral de aproximadamente 14%, o que é muito baixo para um problema de classificação de imagens. Isso significa que o modelo não é capaz de classificar corretamente a maioria das imagens do conjunto de dados.

A matriz de confusão revela que o classificador KNN está cometendo erros significativos em todas as classes. Por exemplo, a classe 1 (carro) tem uma taxa de *recall* muito baixa (apenas 2%), o que significa que o modelo está tendo dificuldade em identificar carros corretamente. O mesmo padrão se repete em todas as classes.

As métricas macro *avg e weighted avg* também refletem o desempenho global insatisfatório do modelo. A métrica F1-score média ponderada é de apenas 0.08. A 3 apresenta esses resultados.

7.2. SVM

A acurácia do modelo SVM foi de 0.4799. O classificador SVM obteve uma acurácia geral de aproximadamente 47.99%, o que é uma melhoria substancial em relação ao desempenho do KNN. Isso sugere que o SVM é capaz de classificar as imagens com mais precisão do que o KNN.

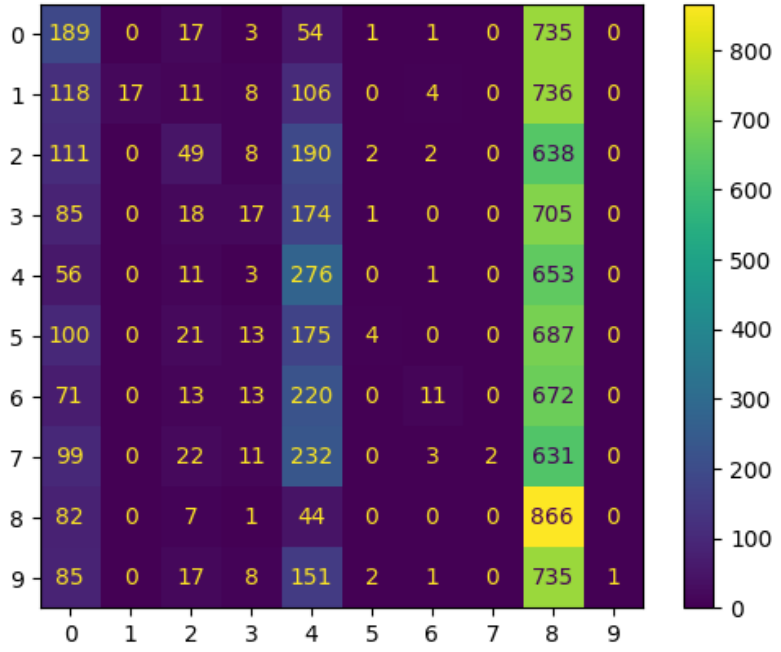

Figure 2. Matriz de confusão do classificador knn

	precision	recall	f1-score	support
0	0.18	0.20	0.19	1000
1	1.00	0.02	0.03	1000
2	0.29	0.05	0.09	1000
3	0.21	0.02	0.03	1000
4	0.17	0.28	0.21	1000
5	0.25	0.00	0.00	1000
6	0.48	0.01	0.03	1000
7	1.00	0.00	0.00	1000
8	0.13	0.87	0.22	1000
9	1.00	0.00	0.00	1000
accuracy			0.15	10000
macro avg	0.47	0.15	0.08	10000
weighted avg	0.47	0.15	0.08	10000

Figure 3. Relatório das métricas do classificador knn

A matriz de confusão mostra que o classificador SVM comete menos erros de classificação em comparação com o KNN. As previsões corretas em cada classe (diagonal principal) são notavelmente maiores, conforme é possível notar na Figura 4 a seguir.

O SVM apresenta um desempenho aceitável em várias classes, com valores de precisão, *recall e F1-score* superiores a 0.4. Esse é um sinal positivo de que o classificador é capaz de distinguir entre diferentes objetos e animais nas imagens.

As métricas *avg e weighted avg* também refletem o desempenho satisfatório do modelo. A métrica F1-score média ponderada é de 0.48, o que é bastante razoável. A 3 apresenta esses resultados.

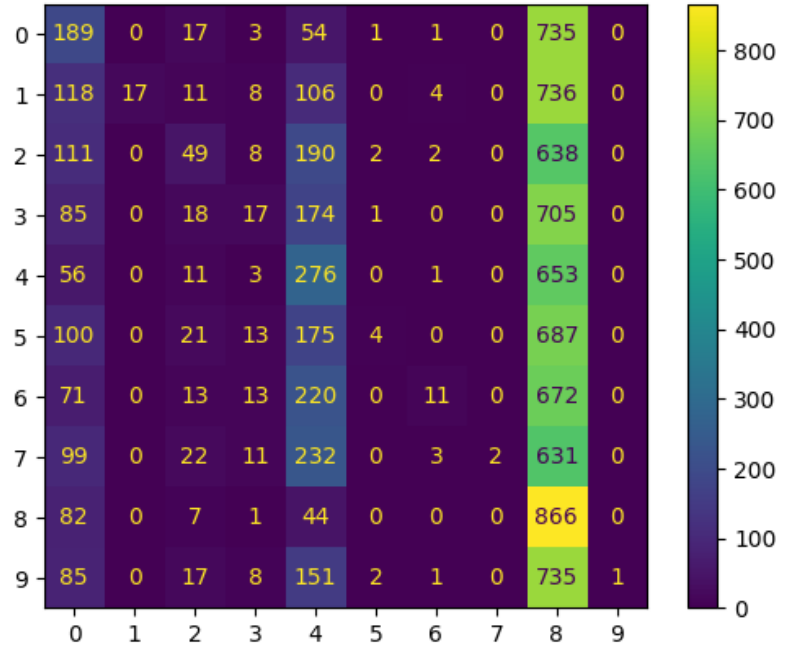

Figure 4. Matriz de confusão do classificador svm

	precision	recall	f1-score	support
0	0.56	0.52	0.54	1000
1	0.49	0.63	0.55	1000
2	0.39	0.36	0.37	1000
3	0.35	0.36	0.35	1000
4	0.45	0.38	0.41	1000
5	0.41	0.36	0.38	1000
6	0.54	0.52	0.53	1000
7	0.56	0.49	0.52	1000
8	0.64	0.60	0.62	1000
9	0.46	0.61	0.52	1000
accuracy			0.48	10000
macro avg	0.48	0.48	0.48	10000
weighted avg	0.48	0.48	0.48	10000

Figure 5. Relatório das métricas do classificador svm

8. Conclusão

Ao treinar os modelos com os dados não normalizados/normalizados, observou-se que a variação na precisão dos modelos não era alta, portanto, normalizando ou não, não foi relevante para a predição de imagens de cada um deles.

Além dos problemas de classificação, a distribuição desigual de amostras entre as classes (por exemplo, a classe 1 tem apenas 2% do total de amostras) pode ter contribuído para o baixo desempenho do KNN. Em resumo, o classificador KNN não é adequado para o conjunto de dados CIFAR-10 nas condições atuais. Para melhorar o desempenho, podem ser consideradas outras técnicas de classificação, ou ajustes nos hiperparâmetros

do modelo. Além disso, o desequilíbrio de classes deve ser tratado para permitir uma melhor generalização do modelo.

Em relação ao classificador SVM, ele demonstrou ser uma escolha melhor do que o KNN para o conjunto de dados CIFAR-10. Ele conseguiu classificar as imagens com uma precisão global significativamente maior, com melhor desempenho em várias classes. No entanto, ainda há espaço para melhorias. O resultado geral de 47.99% indica que o modelo tem capacidade de aprendizado, mas ainda há margem para otimizações futuras. Outra consideração é que o SVM levou cerca de 32 minutos até rodar todas as células correspondentes, não conseguimos identificar o que levou a demorar esse tempo todo.

9. Link do Projeto no GitHub

https://github.com/ACBozzi/T1-Rob-tica-m-vel

References

[Abouelnaga et al. 2016] Abouelnaga, Y., Ali, O. S., Rady, H., and Moustafa, M. (2016). Cifar-10: Knn-based ensemble of classifiers. In *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*.

[Coutinho 2019] Coutinho, B. (2019). Modelos de predição — svm. In *Turing Talks*.

[Didática Tech 2020] Didática Tech (2020). O que é e como funciona o algoritmo knn? In *Didática Tech: Inteligência Artificial Data Science*.