Projeto Demonstrativo 5 - Reconhecimento de Cenas

Natalia Oliveira Borges
natioliveira97@hotmail.com
Matrícula:
16/0015863
Lívia Fonseca
liviagcf@gmail.com
Matrícula:
16/0034078

Departamento de Ciência da Comptutação
Universidade de Brasília
Campus Darcy Ribeiro, Asa Norte
Brasília-DF, CEP 70910-900, Brazil,

0 1

Abstract

Visão computacional é a forma de obter informações a partir de imagens. Nesse contexto, é possível obter informações suficientes de uma imagem para classificá-la em determinada categoria. Esse trabalho apresenta métodos de reconhecimento e classificação de imagens usando algoritmos baseados em Bag of Features. Para extração de features, forma usados os métodos de SIFT e ORB e para classificação forma usados os métodos knn e svm.

1 Introdução

Classificação e detecção de objetos em imagens é mais uma importante aplicação de visão computacional muito utilizada atualmente. Sistemas de desbloqueio de celular utilizando reconhecimento de face do usuário e identificação de placas de veículos para aplicação de multas são algumas das muitas utilidades de métodos de detecção e classificação.

Nesse projeto temos o objetivo de propor e testar métodos para reconhecimento de cenas. Para isso utilizamos um banco de 1500 imagens de treinamento e 2985 imagens de teste, separadas em 15 classes diferentes [].

Os métodos propostos são baseados no modelo 'Bag of Features' que explicaremos em seguida.

1.1 Bag of Features

Bag of Features ou Bag of Visual Words é um método bastante utilizado para classificação de imagens. Esse método se utiliza de extração de features de imagens para criar um histograma

[0]Livia: Relatório(Introdução, Metodologia, Resultados e Discussões e Conclusão)

[1]Natalia: Código (Método (SIFT+KNN), Método (SIFT+SVM), Método(ORB+KNN); Relátório(Introdução, Metodologia, Resultados, Discussões e Conclusão)

de ocorrência dessas features em cada classe estudada. Para a extração de features foram usadados os métodos SIFT e ORB.

1.2 Scale-invariant Feature Transform

SIFT é um método de extração de features muito usado em visão computacional. Esse método se baseia na detecção e descrição de pontos de interesse [5] e possui a vantagens de ser quase indiferente a mudança de escala e rotação.

Para encontar os candidatos é usado o método DoG. É feita da convolução da imagem com filtros gaussianos em diferentes escalas. Em seguida, é calculada a diferença entre as imagens após a passagem dos filtros. As máximas e mínimas diferenças são escolhidas como candidatos a ponto de interesse. Para cada ponto de interesse é então calculado um descritor, um vetor de 128 dimensões.

1.3 Oriented Fast and Rotated Brief

Assim como o SIFT, o ORB é um método de extração de features baseado no FAST keypoint detector e no BRIEF descriptor. FAST é um método de detecção de cantos. Nesse método, um círculo de 16 pixels é feito ao redor do ponto a ser classificado. Se um número N de pixels seguidos tiver brilho maior que o brilho do pixel de teste somado a um valor de limiar ou tiver brilho menor que o brilho do pixel de test subtraindo um valor de limiar, o pixel é classificado como canto [4]. O BRIEF é um algoritmo que usa simples testes binários em uma imagem.

Os descritores dos pontos de interesse desse método possuem 32 dimensões gerando um menor custo que o SIFT e tornando-o mais rápido.

1.4 Algoritmo K-Means Clustering

Esse algoritmo é usado para classificar dados de maneira automática [2]. O algoritmo funciona da seguinte maneira: primeiramente, são fornecidos k centroides e algoritmo calcula a distância entre todos os dados e cada um dos centroides. Então, os dados são calssificados de acordo com a distância dos centroides de cada classe, o dado é classificado na classe cujo centroide apresentou a menor distância do dado. Após a classificação, as coordenadas dos centroides são refinadas fazendo a média das coordenadas de todos os dados que pertencem à cada classe. E, então, os dados são classificados novamente. O algoritmo se repete até que haja convergência nos valores das coordenadas dos centroides, ou seja, até que os dados passem a não mais mudar de classe em uma nova classificação.

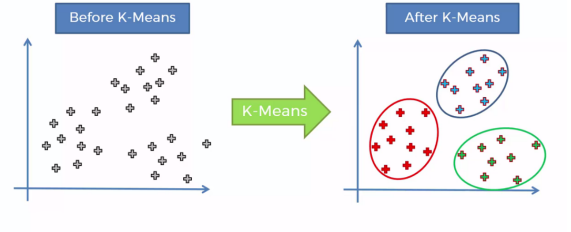


Figure 1: Algoritmo K-Means Clustering

Esse método é relativamente custoso, pois requer o cálculo de k·n distâncias para cada iteração, sendo k o número de centroides e n o número de dados que se deseja classificar.

1.5 Método K-Nearest Neighbours

O método K-Nearest Neighbours é um método de aprendizagem cujo foco é classificação de características [4]. Para o treinamento, é escolhido um certo número de amostras com a sua devida classificação.

Depois do treinamento, é feito o teste usando uma outra amostra. Para cada ponto da amostra de teste é calculada a distância entre esse ponto e todos os pontos do treinamento. Esconhendo um número k, de preferencia impar, vemos quais são os k pontos do treinamento que mais se aproximam do ponto de teste. A classificação mais frequênte nesses k pontos será a classificação do ponto de teste.

Esse método possui um treinamento relativamente rápido, já que é basicamente um armazenamento de dados. Porém para cada teste é necessário calcular a distância do ponto estudado para todos os pontos de treinamento e encontrar os k vizinhos mais próximos, tornando a etapa de teste bastante lenta.

1.6 Método Support Vector Machine

Support Vector Machine [6] é um método de aprendizado de máquina para fazer classificação. O objetivo desse método é dividir o espaço de dados por meio de hiperplano em que de um lado ficam os dados de uma classe e do outro os da outra classe. Esse hiperplano pode ser uma reta que segmenta dois semiplanos em um espaço 2D.

Quando os dados não podem ser classificados de forma linear, realisamos um "kernel trick" para moldar a forma dos dados e conseguir segmentar através de um hiperplano.

2 Metodologia

Esse trabalho foi feito em Linux Xubuntu versão 18.04 e em Linux Ubuntu versão16.04, ambos usando Opencv versão 3.2.0. O código foi feito python3.5.

Para o treinamento, foram usadas 1500 imagens de teste separadas em 15 categorias. Em seguida, os métodos escolhidos foram aplicados em 2985 imagens para classificá-las.

2.1 Bag of Features

Para classificar as cenas, primeiramente devemos retirar características da imagem e selecioná-las em classes, formando uma 'bag of features' para cada classe.

Para extrair características utilizamos detectores de features (SIFT e ORB), explicados em 1. Geralmente, em cada imagem são identificadas vários 'keypoints' e para cada keypoint é calculado um descritor.

Para padronizar todas as classes com a mesma quantidade de features, fazemos um processo de clustering. Escolhemos um valor k que define a quantidade de features de cada classe, esse k será usado como hiperparâmetro para analisar os métodos implementados. Para realizar esse processo utilizamos o algoritmo k-means clustering.

2.2 Método 1 - SIFT + KNN

Utilizamos o detector de features SIFT para extrair as características, tanto das imagens de treinamento quanto nas imagens de teste, e realizamos o clustering dos descritores encontrados.

Treinamos um método de aprendizagem knn com os descritores encontrados para as imagens de treinamento. Aplicamos o treinamento nas imagens de teste e classificamos cada feature em uma classe.

Selecionamos a imagem em uma classe de acordo com a recorrência da classificação de cada feature.

2.3 Método 2 - SIFT + SVM

Como no método anterior, usamos o detector de features SIFT para encontrar os descritores. Realizamos clustering e fizemos um treinamento desses descritores por meio do método de aprendizagem svm.

Aplicamos esse treinamento nas imagens de teste e usamos a mesma abordagem do método 1 para classificar as imagens de teste.

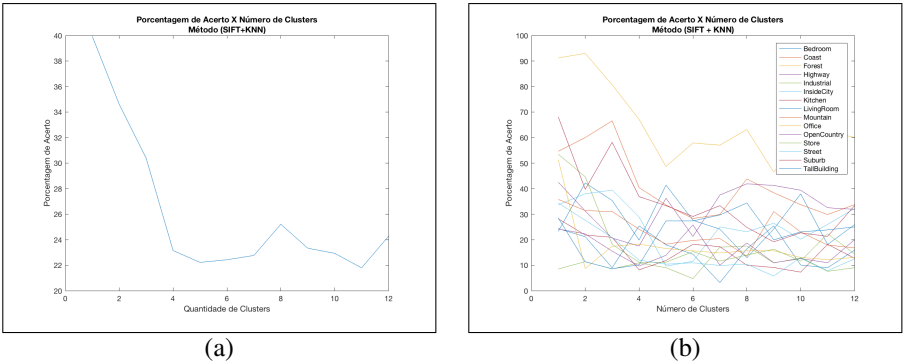
2.4 Método 3 - ORB + KNN

Para esse método, utilizamos o detector de features ORB e realizamos clusterins dos descritores. Com esses descritores treinamos um método de aprendizagem knn e realizamos o mesmo procedimento do método 1 para classificar as imagens.

3 Resultados e Análise

3.1 Método 1 - SIFT + KNN

Para esse método, variamos o número de clusters para obter a acurácia total e plotamos o resultado no gráfico 2(a). Para se ter uma visão mais ampla do resultado, também plotamos a acurácia de cada classe no gráfico 2(b).



(a) (b)

Figure 2: (a) Acurácia total (b) Acurácia por classe

Vimos que para esse método a melhor acurácia aconteceu para k=1, logo, categorizar o ambiente em uma única feature foi a melhor maneira de classificar.

A acurácia para cada classe variou bastante dependendo do número de clusters, mas tende a ser maior para k=1 também. Podemos ver pelo gráfico 2(a) que há uma variação grande de acurácia para cada classe, com isso concluímos que existem classes mais fáceis de classificar que outras. Para esse método, a classe com maior acurácia média foi 'Floresta' e a classe com pior acurácia média foi 'Industria'.

Para o valor de k com maior acurácia total fizemos a matriz de confusão:

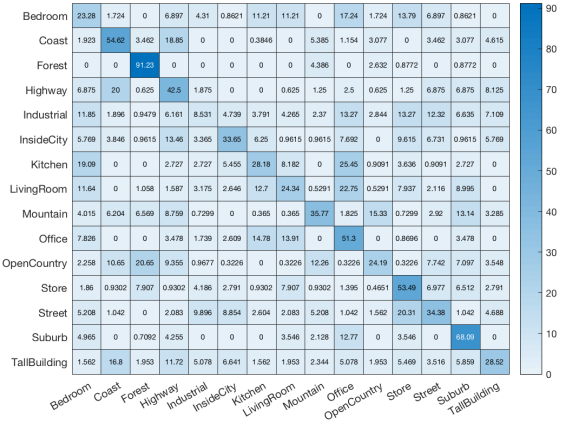
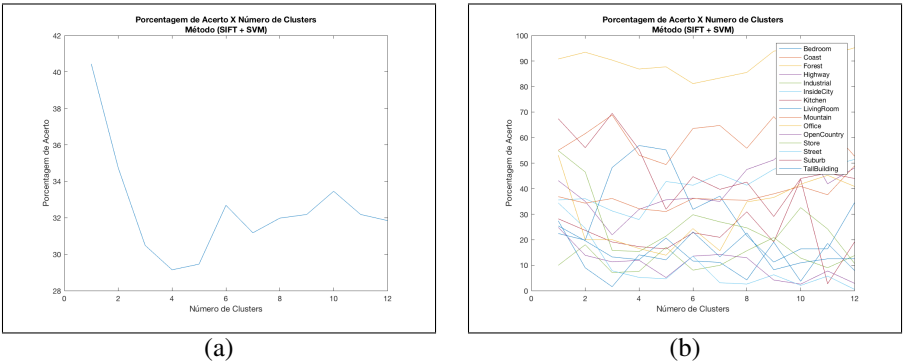


Figure 3: Matriz de Confusão para o Método 1

Vemos que a matriz de confusão segue a tendencia de ter a diagonal acentuada, o que é esperado.

3.2 Método 2 - SIFT + SVM

Fizemos o mesmo prodedimento aplicado no método 1 e obtivemos os resultado em 4.



(a) (b)

Figure 4: (a) Acurácia total (b) Acurácia por classe

O comportamento dos métodos de aprendizagem de acordo com a variação do número de clusters seguiu uma mesma tendência, foi melhor para k=1. A variação de acurácia para cada classe também foi evidente no método svm e novamente a classe 'Floresta' teve a maior acurácia. A menor acurácia foi da classe 'Rua'.

Também montamos a matriz de confusão para o valor de k com maior acurácia:

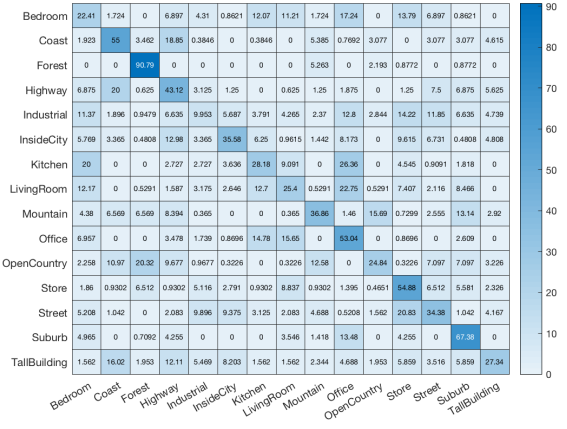
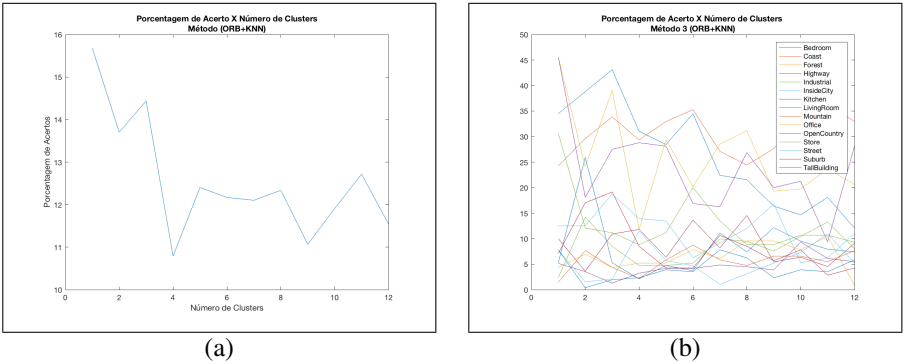


Figure 5: Matriz de Confusão para o Método 2

Essa matriz de confusão também segue a tendência de ser acentuada da diagonal.

3.3 Método 3 - ORB + SVM

Fizemos a análise da acurácia tanto no total das imagens quanto em cada classe e obtivemos os gráfico em 6.



(a) (b)

Figure 6: (a) Acurácia total (b) Acurácia por classe

Para esse método, realizar o clustering para apenas uma característica também foi mais efetivo, porém os resultados desse método não foram satisfatórios. Vimos isso claramente no gráfico de acurácia por classe, que obteve baixos resultados.

Plotamos a matriz de confusão para k=1 e obtivemos 7.

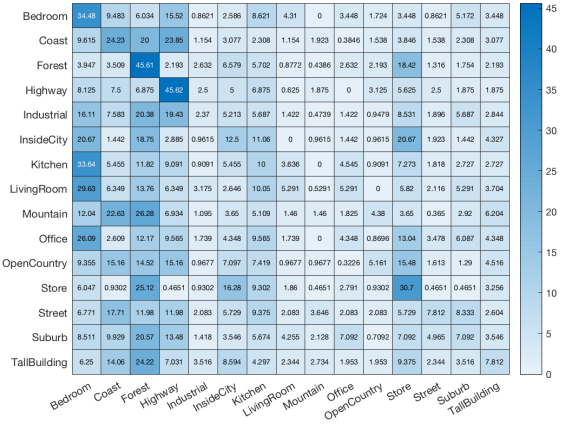


Figure 7: Matriz de Confusão para o Método 3

Os resultados da matriz de confusão mostram que o método não foi efetivo para classificação de cenas. Apesar de não ter sido um método efetivo, seu tempo de preocessamento foi bem menor em comparação com o método baseado em SIFT.

4 Discussão e Conclusões

Vimos por meio desse projeto como utilizar extratores de características para classificar cenas em classes. Aprendemos como utilizar os detectores de features SIFT e ORB e como utilizar métodos de aprendizagem para interpretar esses dados.

Dos métodos utilizados, o que obteve melhores resultados foi o Método 2, que utilizou SIFT para extrair features e o método SVM de classificação. Porém, a acurácia máxima obtida foi de cerca de 40% que é baixa para afirmar que o método é efetivo.

Houve diferenças entre a acurácia obtida para cada classe analisada. Algumas classes foram facilmente identificadas e outras foram muito confundidas. Isso se deve a fatores além dos métodos de detecção e aprendizagem, já que existem cenas que são realmente ambíguas.

Para melhorar esse projeto, seria ideal aplicar uma rede neural convolucional em vez de métodos de aprendizagem, já que essas geralmente apresentam melhores resultados.

Como interesse pessoal do grupo, gostaríamos de aplicar os conceitos aprendidos nesse trabalho praticamente para auxiliar robôs a jogar futebol. Participamos de uma equipe de competição de futebol de robôs humanoides e enfrentamos vários desafios em visão computacional, como detectar bola, gol, outros jogadores. Esse trabalho promover um ótima base para novas implementações.

References

[1] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2169–2178. IEEE, 2006.

[2] Mauro Pichiliani. Data mining na prática: Algoritmo k-means, 2007. https://www.devmedia.com.br/data-mining-na-pratica-algoritmo-k-means/4584.

[3] Wikipedia contributors. Features from accelerated segment test — Wikipedia, the free encyclopedia, 2018. https://en.wikipedia.org/wiki/Features_from_accelerated_segment_test;[Online; accessed 18-November-2018; Page Version ID: 865557299].

[4] Wikipedia contributors. K-nearest neighbors algorithm — Wikipedia, the free encyclopedia, 2018. https://en.wikipedia.org/w/index.php?title=K-nearest_neighbors_algorithm&oldid=865557299;[Online; accessed 30-October-2018; Page Version ID: 865557299].

[5] Wikipedia contributors. Scale-invariant feature transform — Wikipedia, the free encyclopedia, 2018. https://en.wikipedia.org/wiki/Scale-invariant_feature_transform#Algorithm;[Online; accessed 18-November-2018; Page Version ID: 865557299].

[6] Wikipedia contributors. Support vector machine — Wikipedia, the free encyclopedia, 2018. https://en.wikipedia.org/w/index.php?title=Support_vector_machine&oldid=865035110;[Online; accessed 30-October-2018; Page Version ID: 865035110].