Título do Trabalho : estimar gesto ou posição atual com base em imagens

Introdução

O grupo é composto por Tobias Aguiar e Leandro de Souza. O relatório em questão consiste em análise e reconhecimento de padrões a partir de fotos de usuários fazendo determinado padrão (em pé, sentado, braço levantado, etc.), e foi usado um algoritmo de Machine Learning que, a partir, de uma foto qualquer, ele possa identificar que tipo de gesto ou posição encontra-se o indivíduo da foto. Antes, para isso, foram gerados uma base de dados para as imagens mencionadas, para poder implementar o algoritmo em cima disso. A motivação deste trabalho vem do fato de que, com o processamento da imagem e um certo reconhecimento de padrões, é possível expandir o projeto para algo mais complexo, a exemplo de usar uma câmera que reconhece quem entra e quem sai de um recinto, além de poder prever alguém que esteja realizando comportamentos suspeitos, por exemplo. Algo como isso pode facilitar e muito a vida dos oficiais de uma cidade e desenvolver mais a segurança dela.

Metodologia

O modelo de Machine Learning (ML) utilizado neste relatório é o Support Vector Machine (SVM), um modelo de aprendizado supervisionado que analisa dados muito utilizados geralmente para classificação e análise de regressão. Foi utilizado por apresentar algumas características consideradas relevantes, a exemplo de gerar boa generalização, evitando sobretreinamento, algo que é interessante para os projetos envolvendo ML, além de apresentar uma robustez na caracterização de dados. Inicialmente o algoritmo aplica uma transformação de espaços, tornando uma função que antes não era linearmente separável em uma nova que se torna linearmente separável através da denominada funções de Kernel, que possuem papel de mapear funções não-lineares e mapeá-las em outro espaço, onde gera um hiperplano ótimo sobre o espaço das características. Algumas funções kernel mais conhecidas são a linear, polinomial e RBF.

No campo de aprendizado de máquina e especificamente no problema de classificação estatística, uma matriz de confusão, também conhecida como matriz de erros, é um layout de tabela específico que permite a visualização do desempenho de um algoritmo, tipicamente um aprendizado supervisionado (em aprendizado não supervisionado, geralmente é chamado de matriz correspondente). Cada linha da matriz representa as instâncias em uma classe prevista, enquanto cada coluna representa as instâncias em uma classe real (ou vice-versa). O nome deriva do fato de facilitar ver se o sistema está confundindo duas classes (ou seja, geralmente classificado incorretamente uma como outra).

Códigos

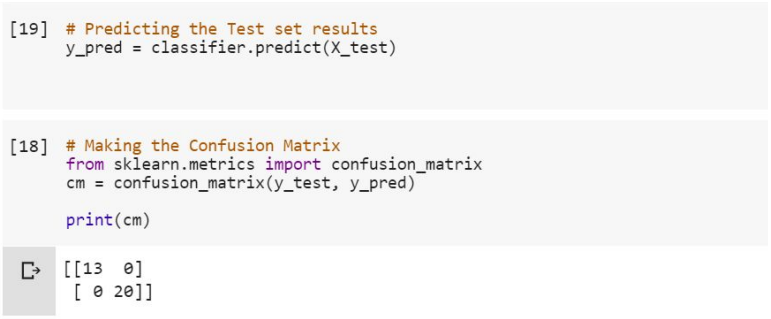
Normalmente, todo código de machine learning inicia-se com a adição da biblioteca, link dos dados e definição da entrada e saída, e não é diferente com esse código da SVM. tendo feito, inicia o processo descrito pela figura 01: definição dos dados de treino e teste, ajuste da escala e classificando o treino através do SVW. Considerou que a proporção do teste era 25% e treino 75%. Assim, é gerado as classes do treino por meio do “classifier.fit”.

Figura 01- Estrutura Inicial

```
[9]  # Splitting the dataset into the Training set and Test set
     from sklearn.model_selection import train_test_split
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

[10] # Feature Scaling
     from sklearn.preprocessing import StandardScaler
     sc = StandardScaler()
     X_train = sc.fit_transform(X_train)
     X_test = sc.transform(X_test)

[11] # Fitting SVM to the Training set
     from sklearn.svm import SVC
     from sklearn.pipeline import Pipeline

     #classifier = SVC(kernel = 'linear', random_state = 0)
     classifier = Pipeline((
     ("scaler", StandardScaler()),
     ("svm_clf", SVC(kernel="rbf", gamma=0.01, C=1000000))
     ))


     classifier.fit(X_train, y_train)
```

Para gerar o resultado da matriz de confusão, é preciso adicionar a entrada de teste(figura 02). O comando que permite isso é o “classifier.predict”, responsável por classificar o teste conforme as configurações do treino. A matriz de confusão é logo gerada pelos parâmetros do “y_teste” e “y_pred”, informando a precisão do código de reconhecimento de posição.

Figura 02- Código final para produzir a matriz de confusão.



```
[19]  # Predicting the Test set results
      y_pred = classifier.predict(X_test)
```

```
[18]  # Making the Confusion Matrix
      from sklearn.metrics import confusion_matrix
      cm = confusion_matrix(y_test, y_pred)

      print(cm)
```

```
[[13  0]
 [ 0 20]]
```

Experimentos

Os dados usados são 132 fotos com posições de partes do corpo. As 78 fotos iniciais são da posição com braços relaxados e as últimas com braço levantado. Os parâmetros de entrada escolhido foram os punhos e cotovelos, pois assim infere se os braços estão levantados.

A matriz de confusão, o qual mede a precisão do código, teve 100% de acerto. Conforme a figura 02, das 33 fotos para o teste, 13 fotos dizem que o braço está normal e o teste previu certo, e a mesma coisa para o restante das 20 imagens com o braço levantado.