Projeto de Algoritmos
Projeto 3
Reconhecimento de Movimentos usando Dynamic Time Warping

Relatório de Entrega

17/06/2015

Sérgio Oliveira Campos

Nº USP: 5240400

1. Introdução

Para o projeto 3 da disciplina Projetos de Algoritmos (1º semestre 2015) foi solicitado que os alunos implementassem um software para reconhecer e classificar movimentos utilizando o algoritmo de Dynamic Time Warping (programação dinâmica).

O código do projeto foi escrito com o uso da linguagem Python (PyPy versão 2.6.0) e está contido no arquivo *moves.py*. Os dados de entrada fornecidos pelo professor, conjuntos de treino e teste, se encontram no diretório *data*.

Para a execução do código utilize o *moves.py* (suas opções de execução podem ser visualizadas ao se executar *"moves.py --help".* Seguem exemplos de uso:

1. Execução com uma dimensão e sem otimizações:

```
python moves.py data/treino.txt data/teste.txt
```

2. Execução com uma dimensão e com o uso da banda de Sakoe Chiba de 10%:

```
python moves.py --sakoe-chiba=0.1 data/treino.txt data/teste.txt
```

3. Execução com três dimensões e com o uso da banda de Sakoe Chiba de 10%:

```
pypy moves.py --sakoe-chiba=0.1 --3d data/treino3D.txt data/teste3D.txt
```

Este relatório introduz a implementação do aluno e discute brevemente a performance da mesma. Na seção 2 é apresentada a estratégia utilizada para a implementação e na seção 3 é realizada uma análise de performance da implementação e seus resultados.

2. Estratégia de Implementação

A solução implementada utiliza o algoritmo de Dynamic Time Warping (DTW). A entrada é realizada através de parâmetros de configuração e dois arquivos, de treino e de teste respectivamente. A função utilizada para calcular o DTW é a função ***DTWDistance*** que por sua vez utiliza como função de distância a função $f(x, y) = (x - y)^2$.

O funcionamento da função ***DTWDistance*** consiste na comparação de cada entrada do conjunto de teste com todas as entradas do conjunto de treino com o objetivo de encontrar qual a entrada no treino mais similar a entrada de teste. Após o processo de classificação a classe encontrada e a classe conhecida são comparadas e caso sejam a mesma um acerto é computado. Ao final do processo a quantidade de acertos é dividida pela quantidade de entradas no conjunto de testes retornando a porcentagem de acertos do processo de classificação.

2.1. Banda de Sakoe Chiba

Para reduzir o uso de memória foi utilizada a banda de Sakoe-Chiba. Este algoritmo consiste em um redução de área da matriz de memoização que utilizada para calcular a distância DTW. Uma banda de 100% representa toda a matriz enquanto uma banda de 10% representa apenas 10% ao dos pontos ao lado de cada ponto da diagonal principal.

Para calcular a porcentagem dos pontos é utilizada a menor entrada. A implementação tradicional do DTW é de complexidade $O(n^2)$ porém o uso da banda de Sakoe-Chiba esta complexidade é reduzida a uma porcentagem disso (dependendo da banda).

2.2. Calculo para entrada em 3 dimensões

Para realizar o calculo em 3 dimensões a função ***DTWDistance*** realiza três chamadas recursivas (uma para cada dimensão) e ao final soma os valores obtidos por cada dimensão.

3. Resultados Obtidos

A tabela abaixo compara os resultados e tempos obtidos pela execução do software com uma dimensão e diferentes bandas de Sakoe-Chiba:

Sakoe-Chiba	100%	50%	20%	10%	5%	1%	0%
Resultado	0.84687	0.85312	0.87083	0.86145	0.75520	0.43125	0.38958
Tempo de execução (segundos)	15.6	12.0	7.9	5.9	3.8	2.8	2.8

A tabela a seguir compara as mesmas configurações anteriores porém para a execução com três dimensões:

Sakoe-Chiba	100%	50%	20%	10%	5%	1%	0%
Resultado	0.7898	0.82587	0.79353	0.73507	0.66293	0.49502	0.47388
Tempo de execução (segundos)	62.8	46.3	29.8	25.9	20.3	12.3	11.3

Dado que a banda de 100% é equivalente a implementação sem banda e a banda de 0% é equivalente a distância euclidiana, fica claro que o uso do DTW aliado a uma banda Sakoe-Chiba intermediaria melhora a porcentagem de acertos da distância (em ambos os casos 1D e 3D). Com base nos resultados encontrados provavelmente uma banda em torno de 20% seria a ideal em termos de trade-off entre performance e qualidade.