Relatório do Projeto 3 – DTW

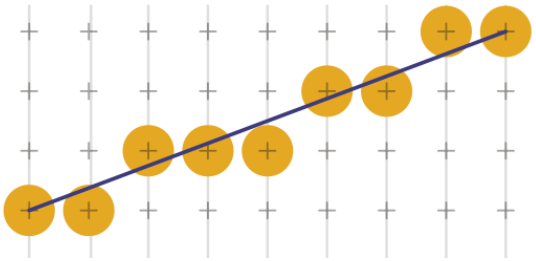
Disciplina: Projeto de Algoritmos

Aluno: Cézanne Alves - 10846548

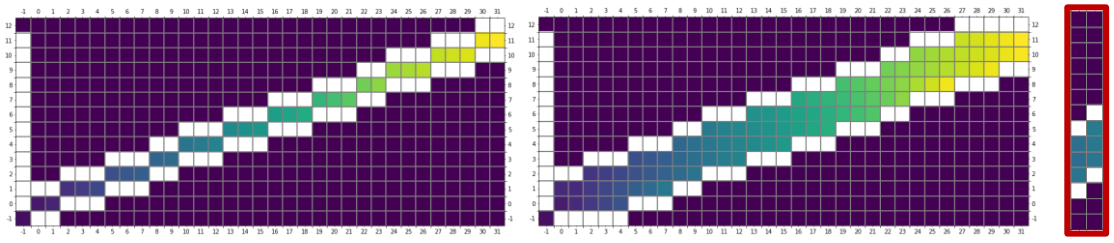
O Algoritmo foi implementado em python3, inicialmente foram usados *arrays numpy* para lidar elegantemente com dimensões variáveis, mas operações pontuais em *numpy* são muito caras, e devido as etapas de decisão da programação dinâmica não é possível fazer as operações em lotes (o que resultaria numa performance melhor que a execução em *python* nativo)

A versão implementada foi a com as Bandas de Sakoe-Chiba e usando distância euclidiana (ao quadrado por não necessitar da distância exata) entre pontos de 3 dimensões, chamada de DTW_D na referência provida. A DTW_I não foi usada. A suposição é que, mesmo que um participante possa atrasar, de maneiras diferentes, aspectos diferentes do movimento (como rotação e abdução) em execuções distintas, é improvável que esses aspectos fiquem isolados em eixos distintos do acelerômetro. Isso iria requerer, ironicamente, precisão ao ser impreciso.

As bandas apresentam um desafio não explícito: em bandas estreitas uma implementação ingênua pode levar a buracos no preenchimento, impossibilitando o cálculo correto. Para o caso da banda de largura 0, que aproxima (aproximado pela discretização dos domínios) um *warp* linear, é necessário que cada célula participante da solução, baseie-se em apenas uma célula vizinha. O problema é semelhante ao de *rasterização* de retas. A forma mais simples de *rasterizar* uma reta corretamente é iterar sobre a maior componente e calcular arredondamentos para a outra, dada a inclinação obtida pelas extremidades.



Desta forma, na DTW implementada, a maior série é sempre colocada na abcissa da tabela e sobre ela se itera. As posições dos *padding* de casos-base e de células a serem calculadas são obtidas pela inclinação m da diagonal principal. Como $m \leq 1$, sempre haverá um vizinho pré-calculado à esquerda ou à diagonal baixo-esquerda.



Além disso, o cálculo é feito utilizando apenas a última coluna, o que é armazenado numa matriz global com apenas duas colunas, requerendo apenas espaço $O(\min(n, m))$, e o cálculo é abortado quando a menor distância de uma coluna ultrapassa a do vizinho mais próximo já encontrado (informada por parâmetro).

Cada célula tem seu valor calculado em tempo constante, e devido às bandas estarem definidas em porcentagem da banda máxima, a complexidade de tempo é de fato $O(n \times m)$, com exceção da largura 0, obviamente. Porém se as bandas forem fixadas em uma largura constante a

complexidade se torna $O(\max(n,m))$. A execução em séries de 3 dimensões não afeta a complexidade de tempo e espaço.

Execução

Comparar a acurácia entre a classificação por séries 1D e 2D é impreciso, pois não se sabe o método utilizado para gerar a série 1D, e conjuntos são distintos: não têm a mesma quantidade de séries, e as séries não têm o mesmo comprimento. Mas aparentemente é mais difícil fazer casamentos espúrios com 3 dimensões. Como esperado, o tempo difere aproximadamente de um fator constante.

Os resultados de tempo e acurácia mostram que as bandas melhoram a acurácia e o desempenho. A acurácia não cai muito com a DTW irrestrita, principalmente nas séries 3D. Mas isso pode ser pela densidade de vizinhos: uma série de classe diferente pode ser a mais próxima se houver grande diferença de *warping* ou intensidade para os exemplares da classe correta. Mas à medida que se aumentam os exemplares, aumenta a chance de que algum vizinho tenha intensidade e *warping* semelhantes, o que não significa que a medida de distância está ideal.