

Multimédia

Trabalho Prático nº 2

Music Information Retrieval

Objectivo

Pretende-se que o aluno adquira sensibilidade para as questões fundamentais de **Multimedia Information Retrieval**, em particular de sistemas de recomendação de música baseados em conteúdo (**content-based music recommendation systems**).

Planeamento

Prazo de Entrega:

12 de Maio, sexta-feira, 23h59

Esforço extra-aulas previsto: 18h/aluno

Formato de Entrega:

- 1) Entrega final (código completo + relatório): InforEstudante
- 2) Notas:
 - a) Gerar o ficheiro zip, contendo o pdf do relatório, os ficheiros com o código e outros ficheiros que considere relevantes;
 - b) Para evitar erros de submissão no inforestudante, gerar acrescentar a extensão .pdf ao ficheiro (e.g., ficheiro.zip → **ficheiro.zip.pdf**)

Período de execução: 6 aulas práticas laboratoriais

Ritmo de execução esperado para avaliação contínua:

- Semana 1: alínea 1
- Semana 2: alíneas 2.1 e 2.2
- Semana 3: alínea 3
- Semana 4: alínea 4
- Semana 5 e 6: correcções e relatório (e alínea 2.3 - opcional)

Trabalho Prático

Implementação e análise de um sistema de recomendação de música baseado em conteúdo, usando Python.

1. Preparação.
 - 1.1. Ganhar familiaridade com um sistema de recomendação real, e.g., Jango.com, Spotify, Last.fm, Torch ou outro.
 - 1.2. Descarregar a base de dados a utilizar (4Q audio emotion dataset) do seguinte URL: <http://mir.dei.uc.pt/downloads.html>.
 - 1.3. Analisar a base de dados.
 - 1.3.1. Excertos áudio: ficheiros mp3 (em particular as 4 queries).
 - 1.3.2. Metadados: ficheiro *panda_dataset_taffc_metadata.csv*, colunas Song, Artist, Quadrant, MoodsStrSplit e GenresStr.
 - 1.3.3. Features: ficheiros *top100_features.csv* (features extraídas) e *features.csv* (descrição das features).
 - 1.4. Estudar a framework de processamento áudio *librosa*
 - 1.4.1. Instalar a framework a partir do seguinte URL: <https://librosa.org/>. Será também necessário instalar a framework ffmpeg para leitura de ficheiros .mp3: <https://ffmpeg.org/>.
 - 1.4.2. Analisar o código fonte de base fornecido no InforEstudante (ficheiro *mrs.py*).
 - 1.4.3. Estudar a documentação da framework: <https://librosa.org/doc/>.

Sugestão de funções a utilizar nas alíneas seguintes:

- *os.path.isfile*
- *os.listdir*
- *numpy.genfromtxt*
- *numpy.savetxt*
- *numpy.sort*
- *numpy.argsort*
- *numpy.where*
- *list.split*

2. Extracção de Features.
 - 2.1. Processar as features do ficheiro *top100_features.csv*.
 - 2.1.1. Ler o ficheiro e criar um array numpy com as features disponibilizadas.
 - 2.1.2. Normalizar as features no intervalo [0, 1].
 - 2.1.3. Criar e gravar em ficheiro um array numpy com as features extraídas (linhas = músicas; colunas = valores das features).
 - 2.2. Extrair features da framework *librosa*.
 - 2.2.1. Para os 900 ficheiros da BD, extrair as seguintes features (sugestão: guardar todas as músicas na mesma pasta):
 - Features Espectrais: *mfcc*, *spectral centroid*, *spectral bandwidth*, *spectral contrast*, *spectral flatness* e *spectral rolloff*.

- Features Temporais: *F0*, *rms* e *zero crossing rate*.
 - Outras features: *tempo*.
 - Utilize os parâmetros por omissão do librosa (*sr* = 22050 Hz, *mono*, *window length* = *frame length* = 92.88 ms e *hop length* = 23.22 ms).
- 2.2.2. Calcular as 7 estatísticas típicas sobre as features anteriores: média, desvio padrão, assimetria (*skewness*), curtose (*kurtosis*), mediana, máximo e mínimo. Para o efeito, utilizar a biblioteca *scipy.stats* (e.g., *scipy.stats.skew*).
 - Guarde as features num array numpy 2D, com **número de linhas = número de músicas** e **número de colunas = número de features**
 - 2.2.3. Normalizar as features no intervalo [0, 1].
 - 2.2.4. Criar e gravar em ficheiro o array numpy com as features extraídas.
- 2.3. **Alínea com bonificação de 10% na nota final! (Sugestão: desenvolver nas semanas 5 e 6)**

Implementar features de raiz. Neste ponto, não é permitido utilizar o librosa nem qualquer outra biblioteca, à excepção do *scipy* e *numpy*, e.g., *scipy.fftpack*.

 - 2.3.1. Desenvolver o código Python/numpy para extrair as features anteriores (à excepção de *spectral contrast*, *F0* e *tempo*), usando a mesma parametrização. Comparar os resultados obtidos com os resultados do librosa.
 - 2.3.2. Criar e gravar em ficheiro um array numpy com as features extraídas (i.e., as estatísticas correspondentes, com normalização).
3. Implementação de métricas de similaridade.
 - 3.1. Desenvolver o código Python/numpy para calcular as seguintes métricas de similaridade:
 - 3.1.1. Distância Euclidiana
 - 3.1.2. Distância de Manhattan
 - 3.1.3. Distância do Coseno
 - 3.2. Para cada query, criar e gravar em ficheiro 6 matrizes de similaridade (900x900), uma para cada conjunto de features e métrica de distância utilizada.
 - 3.3. Para cada query, criar os 6 rankings de similaridade (para as 4 queries fornecidas). Considere apenas recomendações de 20 músicas.
 - 3.4. 📌 **Apresentar, comparar e discutir os resultados.**
 4. Avaliação
 - 4.1. Avaliação objectiva.
 - 4.1.1. Para cada uma das 4 queries, obter o ranking das 20 músicas recomendadas com base na correspondência com os **metadados** seguintes: artista, género, quadrante e emoção (colunas *Artist*, *Quadrant*, *MoodsStrSplit* e *GenresStr*). Por cada item coincidente, adicionar um ponto à qualidade da música alvo, e.g., se tanto a música de referência como o alvo tiverem género = jazz e emoção = happy, a qualidade do alvo será 2.
 - 4.1.2. Criar e gravar a matriz de similaridade baseada em contexto (i.e., nos metadados).
 - 4.1.3. Para cada um dos rankings determinados em 3.3, calcular a métrica *precision*, assumindo como relevantes as músicas devolvidas em 4.1.1 (metadados).

- 4.1.4. 📄 **Apresentar, comparar e discutir os resultados.**
- 4.2. Avaliação subjectiva.
 - 4.2.1. Para cada uma das 4 queries, conjunto de 100 features e distância do coseno, avaliar a qualidade de cada uma das 20 recomendações, com base na seguinte escala de Likert ": 1 – Muito Má; 2 – Má; 3 – Aceitável; 4 – Boa; 5 – Muito Boa (para as 4 queries). Cada elemento do grupo deverá efectuar individualmente a avaliação da recomendação.
 - a) Calcular a média e o desvio-padrão de todos os membros por query, assim como a média e o desvio-padrão global para as 4 queries.
 - b) Definindo um score mínimo de 2.5 para “recomendação relevante”, calcular a *precision* resultante.
 - 4.2.2. Para cada uma das 4 queries e similaridade com base nos metadados, avaliar a qualidade de cada uma das 20 recomendações, com base na escala de Likert anterior. Cada elemento do grupo deverá efectuar individualmente a avaliação da recomendação.
 - a) Calcular a média e o desvio-padrão de todos os membros por query, assim como a média e o desvio-padrão global para as 4 queries.
 - b) Definindo um score mínimo de 2.5 para “recomendação relevante”, calcular a *precision* resultante.
 - 4.2.3. 📄 **Apresentar, comparar e discutir os resultados.**