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:
10 de Maio, sexta-feira, 23h59

Esforço extra-aulas previsto: 18h/aluno

Formato de Entrega:

1) Entrega final (código completo + relatório): InforEstudante
2) Notas: Gerar o ficheiro zip, contendo o pdf do relatório, os ficheiros com o código e outros ficheiros que considere relevantes;

Período de execução: 6 aulas práticas laboratoriais

Ritmo de execução esperado para avaliação contínua:

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

Trabalho Prático

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

1. Preparação.
   1.1. Descarregar a base de dados a utilizar (4Q Audio Emotion Dataset, com 900 músicas) do seguinte URL: ***http://mir.dei.uc.pt/downloads.html*.**
   1.2. Analisar a base de dados.
      1.2.1. Excertos áudio: ficheiros mp3 (em particular a query fornecida).
      1.2.2. Metadados: ficheiro ***panda_dataset_taffc_metadata.csv***, colunas Song, Artist, MoodsStrSplit e GenresStr.
   1.3. Estudar a framework de processamento áudio *librosa*
      1.3.1. Instalar a framework a partir do seguinte URL: ***https://librosa.org/*** (pip install librosa).
      Será também necessário instalar a framework ffmpeg para leitura de ficheiros .mp3: ***https://ffmpeg.org/*** (em linux: sudo apt-get install -y ffmpeg)
      1.3.2. Executar e analisar o código fonte de base fornecido (ficheiro ***mrs.py***).
      Para reprodução de ficheiros áudio, será necessário instalar a biblioteca **sounddevice** (conda install conda-forge::python-sounddevice)
      1.3.3. Estudar a documentação da framework librosa, em particular as funções referentes à **extracção de features**: ***https://librosa.org/doc/latest/feature.html***.

   Sugestão de funções a utilizar nas alíneas seguintes:
   - *os.path.isfile*
   - *os.listdir*
   - *numpy.genfromtxt*
   - *numpy.savetxt*
   - *numpy.sort*
   - *numpy.argsort*
   - *numpy.corrcoef*
   - *list.split*

2. **Extracção de Features**.
   2.1. Extrair features da framework *librosa*.
      2.1.1. 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*.
      Nota: para validação, as features deverão ser extraídas e guardadas pela ordem indicada.

- Utilize os parâmetros por omissão do librosa (*sr* = 22050 Hz, mono, *window length* = *frame length* = 92.88 ms e *hop length* = 23.22 ms)

Nota: são parâmetros por omissão, pelo que não será necessário especificá-los na chamada das funções de extracção de features).

2.1.2. **Calcular as 7 estatísticas** típicas sobre as features anteriores: média, desvio padrão, assimetria (skewness), curtose (kurtosis), mediana, máximo e mínimo (para efeitos de debug, sugere-se a utilização desta ordem). 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**

2.1.3. **Normalizar** as features no intervalo [0, 1].

Sugestão: a função a desenvolver poderá receber tanto os valores mínimo e máximo a utilizar na normalização como calculá-los como determiná-los pela gama de valores da variável em causa.

2.1.4. **Criar e gravar em ficheiro** o array numpy com as features extraídas.

Nota: a primeira linha do ficheiro deverá conter o mínimo de cada feature; a segunda linha deverá conter o máximo de cada feature; as linhas seguintes devem conter os feature vectors de cada música.

2.1.5. **Validar os resultados** obtidos com base nos exemplos fornecidos juntamente com este enunciado.

2.2. Implementar a feature **spectral centroid** de raiz. Neste ponto, não é permitido utilizar o librosa nem qualquer outra biblioteca, à excepção do scipy e numpy, e.g., scipy.fft.rfft.

2.2.1. Deverá desenvolver o código Python/numpy para extrair a feature spectral centroid (SC), usando a mesma parametrização usada na alínea 2.1. Para tal, deverá basear-se nos **materiais teóricos da disciplina**, em particular os slides do capítuo de Multimedia Information Retrieval.

2.2.2. **Comparar os resultados obtidos com os resultados do librosa**, usando as seguintes métricas de erro: **coeficiente de correlação de Pearson** e **RMSE**. Sugestão de debug: visualizar os dois arrays de SC (do librosa e o resultado desta alínea); de notar que a extracção de features do librosa tem um **atraso de 2 janelas**.

2.2.3. **Guardar em ficheiro** os resultados das métricas de erro (ficheiro csv com 900 linhas e 2 colunas, uma para cada métrica).

2.2.4. **Validar os resultados** obtidos com base nos exemplos fornecidos juntamente com este enunciado.

3. Implementação de **métricas de similaridade**.

3.1. Desenvolver o código Python/numpy para **calcular as seguintes métricas de similaridade** (poderá também utilizar funções pré-existentes):

3.1.1. Distância Euclidiana

3.1.2. Distância de Manhattan

3.1.3. Distância do Coseno

3.2. Para a query:

3.2.1. **Extrair, calcular as 7 estatísticas e normalizar as features correspondentes**.

3.2.2. **Criar e gravar em ficheiro** 3 matrizes de similaridade, um para cada métrica de distância utilizada.

3.3. Para a query, **criar os 3 rankings de similaridade**, um para cada métrica de distância utilizada. Considere apenas recomendações de **10 músicas**.
3.4. **Validar os resultados** obtidos com base nos exemplos fornecidos juntamente com este enunciado.
3.5. ✍ **Apresentar, comparar e discutir os resultados.**

4. Avaliação.

4.1. Avaliação objectiva.

4.1.1. **Obter o ranking** das 10 músicas recomendadas com base na correspondência com os **metadados** seguintes: artista, género e emoção (colunas Artist, MoodsStrSplit e GenresStr dos ficheiros de metadados ***panda_dataset_taffc_metadata.csv*** e ***query_metadata.csv***). Por cada item coincidente, adicionar um ponto à qualidade da música alvo, e.g., se tanto a query como o alvo tiverem género = jazz e emoção = happy, a qualidade do alvo será 2. Para o efeito, criar e gravar a matriz de similaridade baseada em contexto (i.e., nos metadados).

4.1.2. Para cada um dos rankings determinados em 3.3, calcular a métrica ***precision***, assumindo como relevantes as músicas devolvidas em 4.1.1 (metadados).

4.1.3. **Validar os resultados** obtidos com base nos exemplos fornecidos juntamente com este enunciado.

4.1.4. ✍ **Apresentar, comparar e discutir os resultados.**

4.2. Avaliação subjectiva.

4.2.1. Para cada um dos r**ankings construídos usando as 3 métricas de distâncias**, **avaliar a qualidade de cada uma das 10 recomendações**, com base na seguinte escala de Likert ": 1 – Muito Má; 2 – Má; 3 – Aceitável; 4 – Boa; 5 – Muito Boa. Cada elemento do grupo deverá efectuar individualmente a avaliação da recomendação.

a) Calcular a **média e o desvio-padrão** de cada música recomendada, com base nas avaliações de todos os membro do grupo de trabalho, assim como a média e o desvio-padrão global (para cada ranking).

b) Definindo um **score mínimo de 2.5** para "recomendação relevante", calcular a *precision* resultante.

4.2.2. Para o **ranking de similaridade com base nos metadados**, **avaliar a qualidade de cada uma das 10 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 cada música recomendada, com base nas avaliações de todos os membro do grupo de trabalho, assim como a média e o desvio-padrão global.

b) Definindo um **score mínimo de 2.5** para "recomendação relevante", calcular a ***precision*** resultante.

4.2.3. ✍ **Apresentar, comparar e discutir os resultados.**