Universidade Federal do Rio Grande do Sul
Instituto de Informática
INF01124 - Classificação e Pesquisa de Dados - Turma B
Prof. Renan Maffei

TRABALHO FINAL

Nesse trabalho você irá desenvolver um algoritmo para classificar avaliações de comentários sobre filmes de acordo com o sentimento que eles expressam. Esse problema é derivado de uma competição de programação do site Kaggle[1] na área de análise de sentimentos.

Análise de sentimentos *é o processo de identificação computacional e categorização de opiniões expressas em um texto, especialmente para determinar se a atitude do escritor em relação a um tópico específico, produto, etc., é positiva, negativa ou neutra.*

Veja um exemplo em que a aplicação que classifica um texto em sentimento negativo ou positivo em [2].

1. Funcionalidade básica (70 % da nota)

O algoritmo deve funcionar em duas fases. Na primeira fase, é necessário **calcular o escore das palavras**. Na segunda fase, a partir dos escores das palavras, o sistema poderá calcular os escores para novos comentários. Os escores das palavras serão calculados a partir de um arquivo de comentários de filmes, anotados com sentimento em uma escala de 0 a 4.

Os rótulos de sentimento são:

- 0 = negativo
- 1 = um pouco negativo
- 2 = neutro
- 3 = um pouco positivo
- 4 = positivo

O arquivo de entrada contém comentário extraídos do site **Rotten Tomatoes**[3] e anotados manualmente com o sentimento geral que expressam. Neste arquivo, os comentários de cada filme estão organizados separadamente por linha. Cada comentário é precedido pelo rótulo de sentimento, como ilustrado no segmento abaixo:

Trecho do dataset

```
1 A combination of escapades demonstrating the adage that what is good for the
4 This quiet, introspective and entertaining independent is worth seeking.
1 Even fans of Ismail Merchant's work, I suspect, would have a hard time sitt
3 A positively thrilling combination of ethnography and all the intrigue, betra
1 Aggressive self-glorification and a manipulative whitewash.
4 A comedy-drama combination of nearly epic proportions rooted in a sincere
1 Narratively, Trouble Every Day is a plodding mess.
3 The Importance of Being Earnest, so thick with wit it plays like a reading fr
1 But it doesn't leave you with much.
1 You could hate it for the same reason.
```

[1] https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews
[2] https://transcranial.github.io/keras-js/#/imdb-bidirectional-lstm
[3] https://www.rottentomatoes.com/

A seguir, descrevemos a funcionalidade básica a ser implementada.

1.1. Cálculo do escore das palavras (40%)

Cada comentário possui um sentimento associado, e é composto por várias palavras. Nesta etapa você deve associar o sentimento do comentário a todas estas palavras. Portanto, você deve implementar um **arquivo invertido**, ou seja, projetar uma estrutura de dados para acessar cada uma das palavras contidas nos documentos e coletar informações referentes a elas. Por exemplo, para cada palavra, pode-se inserir numa **tabela hash** (ou **B-Tree**, **TRIE**, etc) um registro contendo *a palavra, o escore, e o número de ocorrências da mesma*. Caso a palavra já exista na tabela, é necessário atualizar o escore e o número de ocorrências do registro.

Para calcular o *escore* de uma palavra usamos a **média dos escores** dos comentários nos quais ela aparece. Por exemplo, a palavra `combination` no exemplo acima teria número de ocorrências 3, e o escore (4+3+4)/3 = 3,67.

1.2. Classificação de novos comentários de filmes (20%)

Uma vez calculados o escore médio das palavras, usaremos a estrutura de dados descrita na etapa anterior para classificar o sentimento em novos comentários de filmes. Para isso, escreva um programa que receba da console (ou interface gráfica) um novo texto de comentário sobre um filme e gere como saída o escore calculado. Este valor corresponde a média dos escores de cada uma de suas palavras.

Um exemplo de entrada é apresentado a seguir:

```
Escreva uma avaliação (pressione ESC para sair):
>> A weak script that ends with a quick and boring finale
O comentário tem um escore médio de 1.79128
Sentimento negativo

Escreva uma avaliação (pressione ESC para sair):
>> Loved every minute of it
O comentário tem um escore médio de 3.39219
Sentimento positivo
```

1.3. Identificar Extremos e Ocorrências (10%)

Identificar e visualizar as palavras mais negativas e mais positivas. Identificar as palavras de maior frequência.

Por exemplo, o programa deve permitir visualizar as top-K palavras (K mais positivas, K mais negativas ou K de maior ocorrência). Permita que o valor de K seja um parâmetro fornecido pelo usuário do programa (por exemplo, através de um input na console).

2. Adicionais (mais 50% da nota)

A seguir são descritas funcionalidades adicionais que permitirão compreender melhor a relação das palavras e sua relação com o sentimento. Escolha algumas funcionalidades e implemente-as **utilizando estruturas de dados de apoio apropriadas**, diferente daquela utilizada para a funcionalidade básica.

2.1. Teste a partir de um arquivo de comentários (5%)

Para você avaliar o desempenho do seu classificador é importante definir uma metodologia de testes consistente. Com esse intuito, na fase de avaliação de novos comentários, permita que o programa leia um arquivo de teste com novos comentários de filmes e retorne um arquivo de saída com o sentimento associado a cada comentário.

Obs: O arquivo de entrada para teste contém um comentário por linha.

2.2. Melhorar a classificação dos comentários (15 %)

Identificar casos em que há erros de classificação, isto é, comentários positivos que são classificados como negativos ou vice-versa (também chamados de falso negativos ou falso positivos). Propor e implementar uma forma melhor de fazer a classificação.

Exemplos:

- Tratar palavras que tiveram uma variação alta nos seus escores.
- Converter todas as letras para minúsculo.
- Remover palavras *stopwords* (palavras que não possuem significado tais como artigos, preposições e conjunções).

Mostrar exemplos que comparem o seu novo algoritmo de classificação com o algoritmo básico. **Apresente exemplos que evidenciem a melhora na classificação com o seu novo algoritmo.**

2.3. Buscar comentários associados a palavras (10%)

Dada uma palavra, gerar um arquivo com todos os comentários que a contém, e o respectivo escore de sentimento. O critério de pesquisa para esta busca são: palavra, e opcionalmente a polaridade do comentário buscado.

Usando o comentário do exemplo anterior, pode-se buscar todos os comentários que contenham a palavra `combination`, ou somente os comentários com avaliação positiva que contenham a palavra `combination`.

2.4. Buscar palavras usando radicais (20 %)

Dado um radical de duas ou mais letras, encontrar todas as suas variações encontradas nos comentários. Por exemplo, com o radical `com`, poderíamos encontrar diversas palavras (`combination`, `comedy`, ...).

3. Material inicial fornecido

Em anexo é fornecido um código inicial no qual vocês pode se basear para desenvolver o seu trabalho. A **escolha da linguagem de implementação é livre**. Porém as estruturas de dados e algoritmos principais **devem ser de sua autoria**. Por exemplo, algumas estruturas básicas de C++ STL (`vector, pair, list, stack, queue`) podem ser usadas, mas estruturas e funcionalidades mais avançadas (`map, unordered_map, sort,` ...) que implementem a funcionalidade exigida não podem ser usadas (pergunte em caso de dúvida).

- Código exemplo em C++ para leitura do arquivo.
- Dataset com avaliações de filmes.

4. Critérios de avaliação

O trabalho deve ser realizado em duplas (excepcionalmente pode ser individual, mas em prévio acordo com o professor). Todo o grupo deve estar presente na avaliação do trabalho. Caso algum colega esteja ausente, será interpretado que ele/ela não realizou o trabalho. Todos os membros do grupo devem ser capazes de responder sobre qualquer coisa do trabalho. O trabalho irá ser julgado usando os seguintes critérios:

- Uso correto das estruturas de dados.
- Relatório sobre as decisões de implementação.
- Organização e documentação do código.

5. Material a ser entregue

- **Relatório**: Explicação geral do programa e quantidade de pontos pretendidos (de acordo com as funcionalidades implementadas).
- **Código fonte**: com instruções de compilação.