Projeto 1

Introdução

Este projeto tem como objetivo trabalhar os conceitos de orientação a objetos, e estruturas de dados sequenciais. Especificamente, abordaremos os assuntos de modelagem computacional de dados através de classes em Python, e o armazenamento dos dados em listas. Os conceitos serão trabalhados tendo por base uma aplicação real, que será adaptada para fins didáticos.

A aplicação que trabalharemos neste e nos próximos projetos é a construção de um sistema de detecção de plágio. Nosso objetivo é construir um sistema que verifique se um documento, ou trechos dele, foi copiado de outro(s). Tais sistemas são essenciais nos dias de hoje, visto que o volume de conteúdo publicado na internet cresce a cada dia, tornando, assim, impossível a verificação manual.

O projeto será ancorado no trabalho de Barrón-Cedeño e Rosso (2009). Os autores propõem nesse trabalho uma abordagem baseada em repetições de sequências de palavras de tamanho fixo. Estas sequências são conhecidas na área de Processamento de Linguagens Naturais como n-gramas. A proposta de Barrón-Cedeño e Rosso é comparar a proporção de n-gramas compartilhados por dois documentos para indicar um possível plágio entre os documentos. Especificamente, sendo p um documento que se deseja verificar e d um documento de referência (pertencente a um conjunto de documentos que se sabe não terem sido plagiados), p foi possivelmente plagiado de d se a medida de contenção, definida abaixo, é maior que um limiar.

$$C(p \mid d) = \frac{|N(p) \cap N(d)|}{|N(p)|}$$

onde N(.) é o conjunto de n-gramas de um documento. Assim, lê-se C(p | d) como a contenção de p em d, sendo ela a proporção de n-gramas de p contidos em d.

O que fazer?

Como dito, o objetivo é implementar um sistema de detecção de plágio. Nessa etapa, você deverá criar as estruturas de dados básicas do sistema. Você deverá criar representações de:

- Um documento. Sua representação de documento deve armazenar a lista de palavras de um documento salvo em um arquivo. Essa lista de palavras, após ser carregada, deve ser armazenada em um vetor de Numpy. A classe também deverá ter uma lista encadeada (implementada por você) para guardar os n-gramas desse documento. A representação de um n-grama é dada a seguir.

- Um n-grama. Essa classe deve guardar as informações referentes a um n-grama, isto é, uma subsequência de tamanho fixo do documento acima. Você é livre para implementá-la como quiser. Isto se refere somente às estruturas de dados internas que serão usadas. Você não pode usar uma representação copiada da internet ou alguma biblioteca.
- Corpus. Você deve criar uma classe que armazene um conjunto de documentos. Esses documentos são arquivos-texto de um diretório (documentos de referência). A classe é responsável por carregá-los e armazená-los em memória principal. Os documentos devem ser representados como acima.

Todos os atributos das classes devem ser privados. Você deverá implementar métodos de acesso para esses atributos, caso seja necessário (podem existir atributos que são estritamente privados). Utilize property para isso (http://blog.aprendapython.com.br/articles/pq-properties-sao-melhores-que-getters-and-setters-1fr72/)

Você deverá implementar também os métodos especiais que se fizerem necessários. Em particular, a classe n-grama deve ter o método __str__ e __repr__ implementados. O resultado desses métodos deve ser a sequência de palavras do documento que compõem o n-grama.

A classe documento deverá conter um método *gerarNGramas* que, caso ainda não tenham sido gerados, deverá criar a lista de n-gramas do respectivo documento. Como dito, cada n-grama é uma subsequência de n palavras do documento original. Os n-gramas são gerados através de uma técnica chamada de *janela deslizante*. Na primeira iteração, os n-primeiros símbolos são considerados um n-grama (janela tem tamanho n). Depois, a janela desliza uma posição à direita e os símbolos de 1 até n+1 formam o segundo n-grama. O procedimento é repetido até não poderem mais ser gerados n-gramas.

A classe documento também deverá implementar o método *contencao*. Esse método deve obrigatoriamente receber outro documento e retornar a contenção do objeto no documento recebido como parâmetro, conforme a definição acima.

A classe Corpus deve conter um método *verificarPlagio*, que recebe um documento e um limiar de contenção como parâmetros e retorna uma lista ordenada dos documentos mais prováveis de terem servido de base para o plágio. Somente documentos com valor de contenção acima do limiar devem ser retornados.

Você também deverá implementar qualquer outro método/função que se fizer necessário. Lembre-se de criar código para testar suas funções e métodos.

Por se tratar de um projeto maior, sugiro o uso de uma IDE com mais recursos que IDLE. Em particular, recomendo o uso de Eclipse (https://www.eclipse.org/) com PyDev. Outra questão importante, mas que está fora do escopo do projeto é o uso de ferramentas de controle de versão de software. Isso é exigido de qualquer profissional da área, então é uma

ótima oportunidade para aprender. A ferramenta mais usada para controle de versão atualmente é o git (https://git-scm.com/book/pt-br/v2).

Por fim, será disponibilizado um conjunto de documentos que serão usados nos testes.

O que entregar?

Você deve entregar todas as classes implementadas. **Não comprima os arquivos de forma alguma**. O uso de qualquer ferramenta de compressão acarretará em perdas de pontos, em particular, se a ferramenta for **rar**.

Não coloque no classroom os dados. Somente o código deve ser entregue.

Como será a avaliação?

Os seguintes pontos serão observados:

1. Completude (foram implementadas todas as funcionalidades especificadas)
2. Correção (as funcionalidade foram implementadas corretamente)
3. Organização (o código está bem organizado -- evite colocar todas as funções em um único arquivo, organize seu código em módulos https://docs.python.org/3/tutorial/modules.html)

Trabalhos plagiados da internet ou colegas serão prontamente anulados. Como não é possível saber quem fez e quem copiou, todos os envolvidos serão penalizados. Isso não significa que você não possa discutir uma solução com o colega, contudo, existe uma diferença bem grande entre pedir ajuda/discutir e copiar trechos ou o todo do outro.

Datas importantes

- Divulgação da proposta: 24/09/18
- Entrega da classe NGrama: 03/10/18
- Entrega da classe Documento: 10/10/18
- Entrega da versão final: 14/10/18

A entrega do projeto será dividida em etapas. A razão é fazer com que haja um desenvolvimento constante do projeto. A intenção é evitar que você deixe para construir o projeto nos últimos dias e, com isso, não consiga executá-lo. Dessa forma, atrasos parciais contarão negativamente para a nota final. Haverá penalização de 10% por dia para as etapas iniciais. Não serão aceitas versões finais entregues com atraso.

Referência

Barrón-Cedeño A., Rosso P. (2009) On Automatic Plagiarism Detection Based on n-Grams Comparison. In: Boughanem M., Berrut C., Mothe J., Soule-Dupuy C. (eds) Advances in Information Retrieval. ECIR 2009. Lecture Notes in Computer Science, vol 5478. Springer, Berlin, Heidelberg