PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO


**Arnour Sabino de Souza Junior**

# ML Text Assistent

## INF2102 - PROJETO FINAL DE PROGRAMAÇÃO

**DEPARTAMENTO DE INFORMÁTICA**
Mestrado em Informática

Rio de Janeiro, 2021

## Lista de ilustrações

## Lista de tabelas

**Sumário**

## 1 Objetivo

A biblioteca de software ***ML Text Assistant*** tem por objetivo auxiliar a extração, o tratamento e a modelagem de tópicos textos de arquivos PDF.

# 2 Especificações

Esta seção especifica o escopo e os requisitos levantados na construção desta biblioteca.

## 2.1 Escopo

Este projeto limita-se a oferecer funções utilitárias para extração de texto de PDF's, limpeza e preparação dos textos extraídos, além da avaliação de coerência de modelos de extração de tópicos.

O que esse software deve fazer:

- Extrair textos de PDF's;
- Aplicar tratamentos ordinários a técnicas de modelagem de tópicos;
- Determinar a quantidade de tópicos ideal com base no modelo de coerência;
- Determinar um resumo dos principais tópicos aplicados a cada documento considerado.

O que esse software não deve fazer:

- Determinar quais métodos de machine learning devem ser aplicados;
- Selecionar hiperparâmetros de modelos;
- Exibir dados em formatos visuais como gráficos ou tabelas.

## 2.2 Requisitos

### 2.2.1 Requisitos Funcionais

**Tabela 1 – Lista de requisitos funcionais**

| ID | Nome | Descrição |
|---|---|---|
| RF1 | Extrair textos | Dado um caminho para diretório contendo arquivos em formato PDF ou o caminho de um arquivo específico, o conteúdo textual deste(s) arquivo(s) devem ser extraídos para arquivo(s) em formato de texto no diretório destino indicado. |
| RF2 | Preparar dados textuais | Dado o caminho para um ou mais arquivos de texto, o conteúdo de cada arquivo deve ser tratado conforme configuração especificada. Tratamentos considerados padrões para a extração de tópicos tais como: normalização de caixa, remoção de pontuação, remoção de caracteres não alfabéticos, remoção de palavras comuns, extração de radicais e de repetições devem ser oferecidos. |
| RF3 | Calcular coeficiente de coerência | Determinar a melhor quantidade de tópicos a serem extraídos pelo modelo com base no coeficiente de coerência. |
| RF4 | Resumir principais tópicos | Indicar qual o tópico principal de cada documento presente no *corpus* |

### 2.2.2 Requisitos não funcionais

**Tabela 2 – Lista de requisitos não funcionais**

| ID | Nome | Descrição |
|---|---|---|
| RNF1 | Biblioteca de software | O projeto deve ser utilizado na instrumentação de experimentos de machine learning apoiando o pesquisador nas tarefas mais comuns deste tipo de análise. |
| RNF2 | Ser agnóstico de modelo de extração de tópicos | Para ser possível um maior reaproveitamento de suas funções, este projeto não especifica o tipo de modelo usado, apenas define um contrato para modelos disponíveis no pacote gensim. |

# 3 Projeto

## 3.1 Linguagem e dependências

Este projeto é implementado na linguagem Python, versão 3.8. Para extração de textos de arquivos PDF o pacote **pdfplumber**. Além dessas dependências os pacotes **nltk** e **gensim** foram escolhidos para preparação de textos e avaliação de modelos de extração de tópicos, respectivamente.

Para o ambiente de desenvolvimento foram incluídos pacotes para avaliação da cobertura de testes e estilo de código.

## 3.2 Módulos

O projeto conta com três módulos: **extraction**, **preparation** e **exploration**. Estes módulos estão internalizados na biblioteca **ml_text_assistant**.

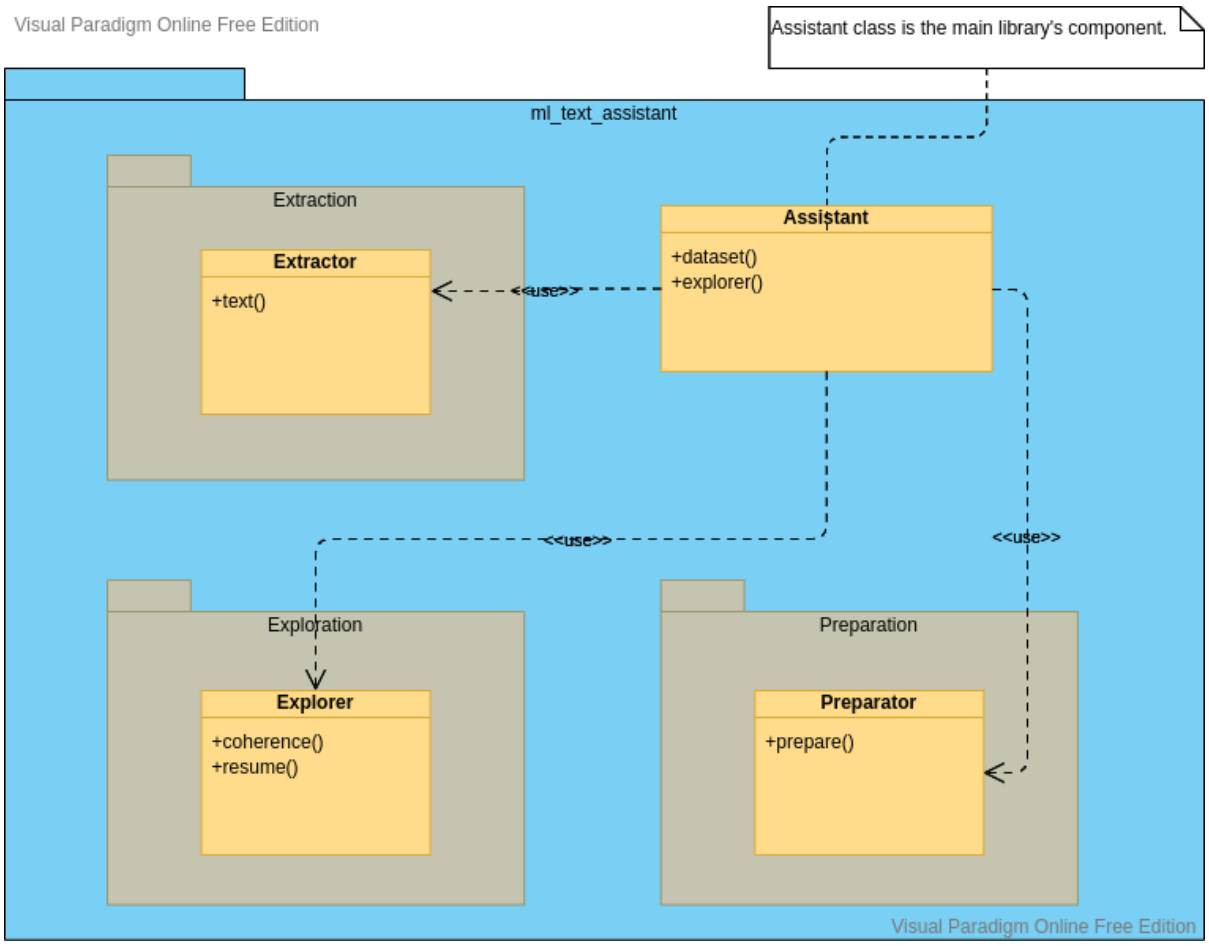
O módulo **extraction** agrupa as funcionalidades de extração de textos de arquivos PDF.

No módulo **preparation** oferece opções de tratamento dos dados de texto, agrupando-as em uma função utilitária que gera o conjunto de tokens que representa cada texto.

Já o módulo **exploration** fornece as operações para avaliar a coerência dos modelos e resumí-los.

## 3.3 Arquitetura

**Figura 1 – Diagrama de pacotes simplificado**




Na figura acima o diagrama representa a estrutura simplificada dos componentes internos da biblioteca. Cada módulo expõe uma classe responsável pelas operações as quais os módulos são destinados.

A classe **Assistant** é o principal componente da biblioteca expondo em seu contrato as principais funcionalidades: extração de dados, preparação de conjunto de dados e exploração da análise de modelos.

## 4 Código

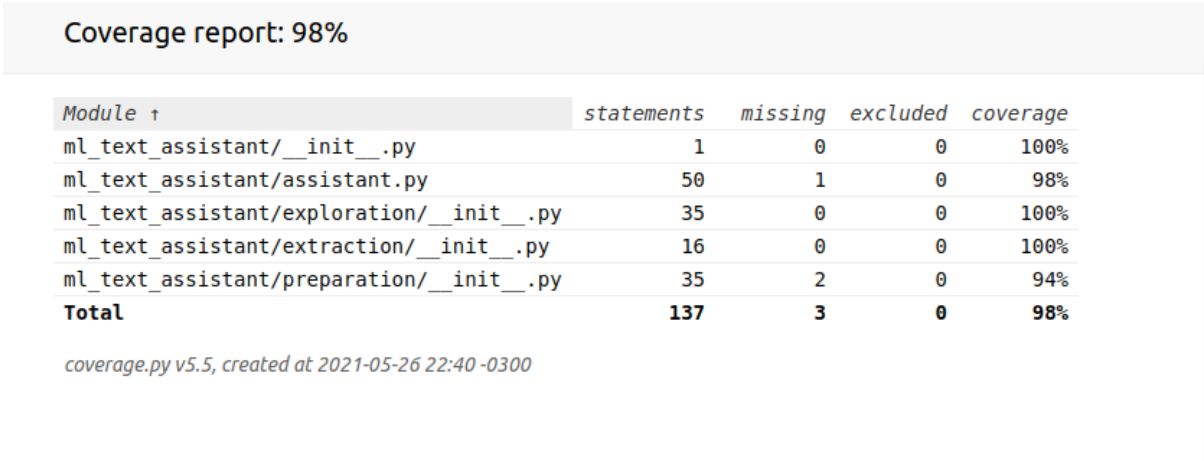
O código fonte está disponível em repositório público através do link *https://github.com/arnour/ml-text-assistant*.

## 5 Testes

Neste projeto foi utilizado unittest como framework padrão. No Makefile é possível encontrar funções para executar os testes e apresentar relatório de cobertura.

Na imagem a seguir temos um exemplo de relatório de cobertura para este projeto.

**Figura 2 – Relatório de cobertura de testes**



Coverage report: 98%

| Module ↑ | statements | missing | excluded | coverage |
|---|---|---|---|---|
| ml_text_assistant/__init__.py | 1 | 0 | 0 | 100% |
| ml_text_assistant/assistant.py | 50 | 1 | 0 | 98% |
| ml_text_assistant/exploration/__init__.py | 35 | 0 | 0 | 100% |
| ml_text_assistant/extraction/__init__.py | 16 | 0 | 0 | 100% |
| ml_text_assistant/preparation/__init__.py | 35 | 2 | 0 | 94% |
| **Total** | **137** | **3** | **0** | **98%** |

*coverage.py v5.5, created at 2021-05-26 22:40 -0300*