

Estudo investigativo sobre o desempenho de atributos de Recuperação de Informação em tarefas de Mineração de Textos

Ruan de Medeiros Bahia
ruanmed@live.com

Orientador: Prof. Dr. Rosalvo Ferreira de Oliveira Neto

Universidade Federal do Vale do São Francisco (UNIVASF)
Curso de Engenharia de Computação (CECOMP)

31 de janeiro de 2020



MINISTÉRIO DA
EDUCAÇÃO

- 1 Introdução
- 2 Fundamentação Teórica
 - Recuperação de Informação
 - Mineração de Texto
- 3 Materiais e Métodos
- 4 Resultados
- 5 Considerações finais e trabalhos futuros
- 6 Referências

- 1 Introdução
- 2 Fundamentação Teórica
 - Recuperação de Informação
 - Mineração de Texto
- 3 Materiais e Métodos
- 4 Resultados
- 5 Considerações finais e trabalhos futuros
- 6 Referências

Evolução dos equipamentos computacionais permite a geração e coleta de grandes volumes de dados diariamente.

(HAN; KAMBER; PEI, 2011, p. 1) afirmam que é comum dizer que vivemos na era na informação, no entanto, segundo eles, vivemos na era dos dados.

How Much Information? 2003

Estudo feito pela Universidade da Califórnia em Berkeley por (LYMAN; VARIAN, 2003). Anualmente, em armazenamento digital, já eram gerados:

- 13,5 terabytes de notícias de jornal;
- 5,5 terabytes de livros;
- 440 exabytes de e-mails.

A Mineração de Textos (MT) aborda o problema da coleta e análise de dados textuais. Deriva técnicas das seguintes áreas:

- Mineração de dados;
- Aprendizados de máquina;
- Processamento de linguagem natural;
- Recuperação de Informação (RI); e
- Gerenciamento do conhecimento.

Aplicações da MT:

- **Classificação** de documentos;
- Clusterização de documentos;
- Sumarização de opiniões;
- Acesso de dados dados biomédicos; e
- Auxílio em investigações forenses.

Classificação de texto

Segundo (JO, 2018, p. 7) e (ZHAI; MASSUNG, 2016, p. 299):

- Classificação (ou categorização) é definida como o processo de designar uma, ou mais, categorias a cada objeto de texto, dentre categorias predefinidas, sendo que predominantemente é utilizado um conjunto de textos já classificados para treinamento.

Figura 1 – Tarefa de categorização de texto (com exemplos de treinamento disponíveis).



Fonte: Figura adaptada de (ZHAI; MASSUNG, 2016, p. 300).

Processo de classificação de texto:

- Derivação de atributos;
- Manejo de atributos (*feature engineering*);
- Diferentes conjuntos de atributo impactam no desempenho dos classificadores de MT;
- Criação de atributos pode melhorar a acurácia.

Recuperação de Informação na Classificação

- Técnicas de RI são utilizadas intensivamente no pré-processamento dos textos para as tarefas de MT;
- Utilização de funções de ranqueamento de RI diretamente na criação de atributos é rara, somente foi encontrada as pesquisas de (WEREN, 2014).

Segundo (BAEZA-YATES; RIBEIRO-NETO, 2011, p. 5–8), a área de Recuperação de Informação abarca os seguintes processos aplicados sobre coleções de documentos:

- Armazenamento;
- Indexação;
- Recuperação; e
- Ranqueamento de consultas.

Foco dos sistemas de RI

Segundo (KOWALSKI, 2010, p. 2, tradução nossa):

- O objetivo principal de um sistema de Recuperação de Informação é minimizar a sobrecarga do usuário em localizar informação de valor. Na perspectiva do usuário, sobrecarga pode ser definido como o tempo que decorre para localizar a informação necessária. O tempo inicia quando um usuário começa a interagir com o sistema e termina quando encontra os itens de interesse.

Recuperação de Informação:

- Área de estudo madura e bem desenvolvida;
- Complexa.

Benefícios da utilização de ferramentas de RI já existentes

- Implementar sistemas de RI que atendam os objetivos da área se torna dificultoso pela necessidade de otimização desses sistemas para atingir o estado da arte.
- É vantajoso usar ferramentas que subsidiem as tarefas de armazenamento, indexação, recuperação e ranqueamento da Recuperação de Informação.
- Facilidade no cálculo das funções de ranqueamento para criação de atributos derivados de RI em tarefas de MT.

Objetivo geral

Avaliar o desempenho de atributos oriundos de Recuperação de Informação para tarefas de Mineração de Textos.

Objetivos específicos

- Avaliar o ganho de desempenho de classificadores de Mineração de Texto com adição de atributos derivados da função de ranqueamento BM25 da Recuperação de Informação, em pelo menos 2 corpus de competições diferentes, utilizando medidas consolidadas na literatura;
- Reproduzir soluções disponíveis *online* para os corpus selecionados, comprovando as medidas dos resultados das competições;
- Elencar em qual dos corpus selecionados os atributos criados proporcionam maior ganho de desempenho de classificador;
- Comparar o desempenho computacional de ferramentas de armazenamento e indexação de textos:
 - ▶ na questão de indexação;
 - ▶ na questão de consulta utilizando as implementações do BM25 nativas das ferramentas;

1 Introdução

2 Fundamentação Teórica

- Recuperação de Informação
- Mineração de Texto

3 Materiais e Métodos

4 Resultados

5 Considerações finais e trabalhos futuros

6 Referências

Campo científico de Recuperação de Informação

Segundo (MANNING; RAGHAVAN; SCHÜTZE, 2008, p. 1), Recuperação de Informação (RI, do inglês *Information Retrieval*) consiste de encontrar material (geralmente documentos) de natureza desestruturada (geralmente texto) que satisfaça uma necessidade de informação dentro de grandes acervos (geralmente armazenados em computadores).

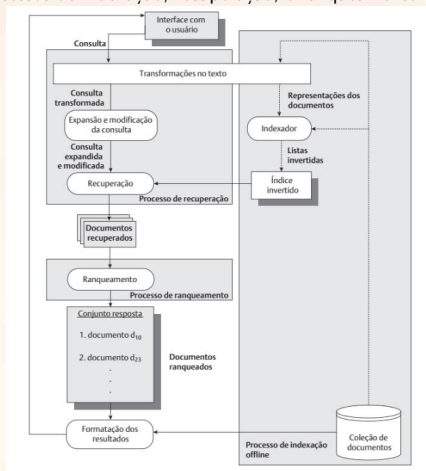
Preocupações iniciais da área, segundo (SANDERSON; CROFT, 2012, p. 3):

- ❶ *como indexar documentos; e*
 - ❷ *como recuperá-los.*
- Início baseado nos sistemas de indexação manuais;
 - Migração para sistema baseado em palavras, Uniterm;
 - Prevalência da indexação por palavras.

Recuperação de Informação

Funcionamento de sistema de RI moderno

Figura 2 – Processos de indexação, recuperação, e ranqueamento dos documentos.



Fonte: Figura extraída de (BAEZA-YATES; RIBEIRO-NETO, 2011, p. 8).

Recuperação ranqueada:

- Estabelece uma pontuação para cada resultado;
- Retorno dos resultados de forma ordenada.

Contabilização do número de aparições de cada um dos termos no documento.

- Número de ocorrências do termo t em um documento d :

$$tf_{t,d} \quad (1)$$

- Relevância dos termos na coleção de documentos:

$$idf_t = \log \frac{N}{df_t} \quad (2)$$

$$tf-idf_{t,d} = tf_{t,d} \times idf_t. \quad (3)$$

Função de ranqueamento Okapi BM25:

- Evolução das implementações do BIM;
- Integra conceitos aplicados no modelo vetorial:
 - ▶ Frequência dos termos;
 - ▶ Normalização de tamanho; e
 - ▶ Correspondência parcial.
- Alguns autores, como (ZHAI; MASSUNG, 2016, p. 111), apresentam a função BM25 junto às dos modelos vetoriais devido à sua similaridade com estes.

Função de ranqueamento BM25

$$\text{Pontuação_BM25}(d_j, q) = \sum_{t \in q} \text{idf}_t \cdot \frac{(k_1 + 1) \text{tf}_{t,d}}{k_1((1 - b) + b \times (\frac{L_d}{L_{\text{avg}}})) + \text{tf}_{t,d}} \quad (4)$$

Os termos $\text{tf}_{t,d}$ e idf_t tem o mesmo significado já apresentado nos modelos anteriores. Os termos b e k_1 são parâmetros de refinamento.

Função de ranqueamento BM25

$$\text{Pontuação_BM25}(d_j, q) = \sum_{t \in q} \text{idf}_t \cdot \frac{(k_1 + 1) \text{tf}_{t,d}}{k_1 ((1 - b) + b \times (\frac{L_d}{L_{\text{avg}}})) + \text{tf}_{t,d}} \quad (5)$$

- $\text{tf}_{t,d}$: contagem do número de ocorrências do termo t no documento d ;
- idf_t : peso de Robertson/Spark Jones dado pela seguinte fórmula:

$$\text{idf}_t = \log \frac{N - \text{df}_t + \frac{1}{2}}{\text{df}_t + \frac{1}{2}}. \quad (6)$$

Mas pode ser simplificado para o valor já apresentado nos modelos anteriores;

- L_{avg} : tamanho médio dos documentos na coleção inteira;
- L_d : tamanho do documento;
- b ($0 \leq b \leq 1$): controle do grau de normalização por tamanho de documento. Usa como referência os valores de L_{avg} e L_d ;
- k_1 ($0 \leq k_1 \leq \infty$): efeito da correção de frequência dos termos presente na fórmula.

Consultas com muitos termos, e repetição destes, podem ser consideradas também pela função Okapi BM25 com a adição de um fator de ajuste k_3 .

Função de ranqueamento BM25 adaptada para termos da consulta

$$\text{Pontuação_BM25}(d_j, q) = \sum_{t \in q} \left[\log \frac{N}{df_t} \right] \cdot \frac{(k_1 + 1)tf_{t,d}}{k_1((1 - b) + b \times (\frac{L_d}{L_{avg}})) + tf_{t,d}} \cdot \frac{(k_3 + 1)tf_{t,q}}{k_3 + tf_{t,q}} \quad (7)$$

- Parâmetros de refinamento b , k_1 e k_3 são definidos para otimizar o desempenho na recuperação em uma coleção de teste.
- Valores experimentais:
 - ▶ k_1 e k_3 : valores entre 1.2 e 2;
 - ▶ $b = 0.75$ ou valores entre 0.5 e 0.8.

A Mineração de Textos (MT) é definida como o processo de extrair conhecimento implícito de dados textuais (JO, 2018; FELDMAN; SANGER, 2006).

- Tratada como *knowledge discovery in text* por alguns autores, (KODRATOFF, 1999) e (FELDMAN; DAGAN, 1995).
- Recebe suporte direto da Mineração de Dados.
- Mineração de Dados é somente uma parte do processo de descoberta de conhecimento (HAN; KAMBER; PEI, 2011, p. 6).

Processo de descoberta de conhecimento em dados

Composto pelas seguintes etapas, segundo (HAN; KAMBER; PEI, 2011, p. 6–7):

- 1 **Limpeza dos dados;**
- 2 **Integração dos dados;**
- 3 **Seleção dos dados;**
- 4 **Transformação dos dados;**
- 5 **Mineração dos dados;**
- 6 **Avaliação de padrões; e**
- 7 **Apresentação do conhecimento.**

Mineração de Texto

O processo de descoberto do conhecimento (KDD)

Figura 3 – Mineração de dados como uma fase do processo de descoberta do conhecimento (KDD).

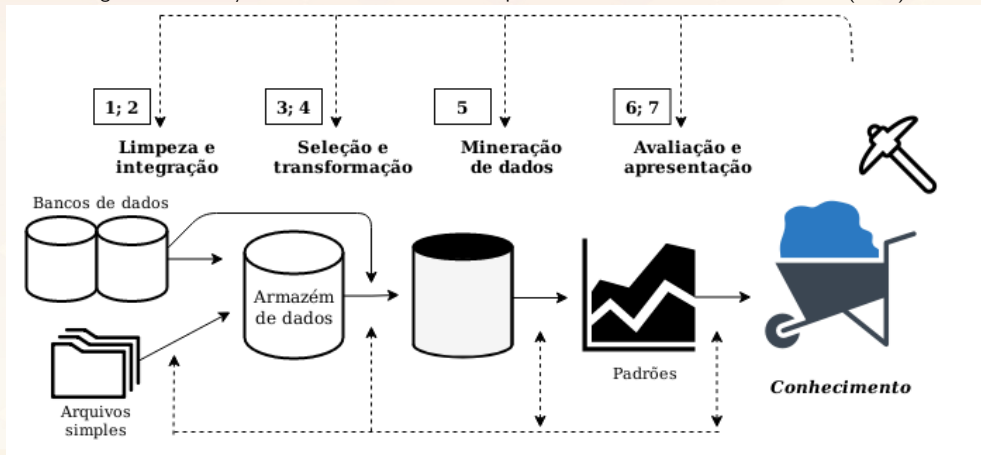


Figura 4 – **Fonte:** Figura baseada na original de (HAN; KAMBER; PEI, 2011, p. 7).

- Mineração de Dados assume que os dados já estão processados.
 - ▶ Pré-processamento do KDD direcionado à limpeza e integração dos dados, etapas 1 e 2.
- Mineração de Texto trabalha com dados desestruturados.
 - ▶ Pré-processamento focado em mais extração de atributos, etapas 3 e 4.
- O pré-processamento da MT utiliza de técnicas da:
 - ▶ Recuperação de Informação;
 - ▶ Extração de informação; e
 - ▶ Linguística computacional.

para transformar as coleções de documentos desestruturados em dados intermediários cuidadosamente estruturados (FELDMAN; SANGER, 2006, p. 2–3).

- Estrutura intermediária definida por um modelo representacional dos documentos de texto composto por um conjunto de atributos.

Apesar da Mineração de Texto utilizar de técnicas da Recuperação de Informação, ambos são campos independentes com objetivos diferentes.

Diferença entre MT e RI

(JO, 2018, p. 4, tradução nossa) ressalta as diferenças:

- A saída da mineração de dados é o conhecimento implícito que é necessário diretamente para a tomada de decisões, enquanto a saída da recuperação é composta por alguns dos itens de dados que são relevantes para a consulta dada. Por exemplo, no domínio de preços de ações, a previsão dos preços futuros de ações é uma tarefa típica da mineração de dados, enquanto que obter alguns dos preços de ações passadas e atuais é tarefa da recuperação de informação. Observe que a certeza perfeita nunca existe na mineração de dados, em comparação com a recuperação. A computação mais avançada para obter conhecimento dos dados brutos, chamada de síntese, é necessária para executar as tarefas de mineração de dados.

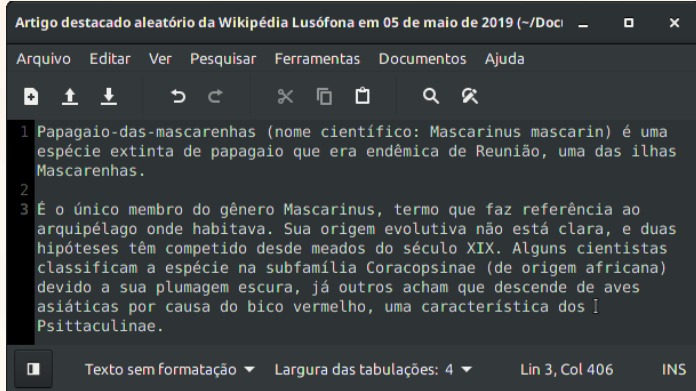
Tabela 1 – Mineração de Dados versus Recuperação de Informação (em específico para objetos de texto a comparação vale para Mineração de Texto versus Recuperação de Texto).

	Mineração	Recuperação
Saída	Conhecimento	Itens relevantes
Exemplo	Valores previstos	Valores anteriores ou atuais
Certeza	Probabilística	Nítida
Síntese	Necessária	Opcional

Fonte: (JO, 2018, p. 4).

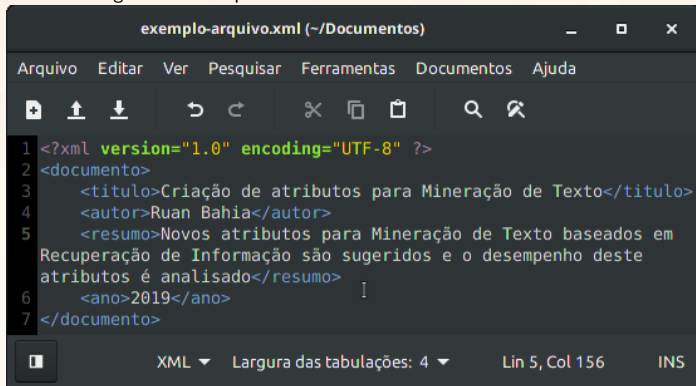
- Vários formatos de texto são utilizados para o processamento computacional de texto:
 - ▶ Formatos proprietários da Microsoft:
 - ★ MS Word com extensão “doc”;
 - ★ MS PowerPoint com extensão “ppt”;
 - ★ MS Excel com o “xls”;
 - ▶ Formatos abertos ODF:
 - ★ Documentos de texto com extensão “odt”;
 - ★ Apresentações com extensão “odp”;
 - ★ Folhas de cálculo com extensão “ods”;
 - ▶ PDF para transferência entre computadores.
- Texto simples (sem formatação) é o formato mais elementar de texto que é feito por um editor.
- Corpus é uma coleção de textos simples, referenciada pelo diretório que contém os arquivos de texto (JO, 2018, p. 6).
- (KWARTLER, 2017, p. 9) considera como corpus qualquer corpo, ou conjunto, grande de texto organizado.

Figura 5 – Arquivo de texto simples, texto não formatado, aberto para edição no xed.



Fonte: O autor, conteúdo do texto obtido de (WIKIPÉDIA, 2019).

Figura 6 – Exemplo de documento XML aberto no editor xed.



```
1 <?xml version="1.0" encoding="UTF-8" ?>
2 <documento>
3   <titulo>Criação de atributos para Mineração de Texto</titulo>
4   <autor>Ruan Bahia</autor>
5   <resumo>Novos atributos para Mineração de Texto baseados em
   Recuperação de Informação são sugeridos e o desempenho deste
   atributos é analisado</resumo>
6   <ano>2019</ano>
7 </documento>
```

Fonte: O autor.

Mineração de Texto

Tarefas da Mineração de Dados

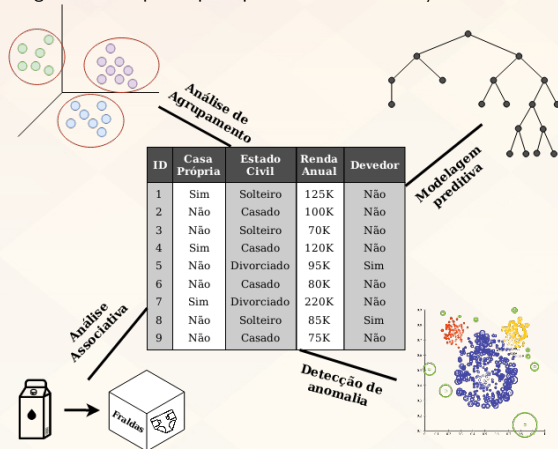
Segundo (TAN; STEINBACH; KUMAR, 2014, p. 7) e (HAN; KAMBER; PEI, 2011, p. 15) são separadas em:

- Descritivas; e
- Preditivas.

As tarefas descritivas são de:

- Agrupamento;
- Associação;
- Descrição; e
- Detecção de anomalias.

Figura 7 – As quatro principais tarefas da mineração de dados.



Fonte: Figura baseada na original de (TAN; STEINBACH; KUMAR, 2014, p. 7).

- O problema da classificação consiste na aprendizagem da estrutura dos exemplos do conjunto de dados, os quais estão classificados em grupos chamados de categorias ou classes (AGGARWAL, 2015, p. 285).
- (TAN; STEINBACH; KUMAR, 2014, p. 146, tradução nossa) dizem que “classificação é a tarefa de aprender uma função objetivo f que mapeia cada conjunto de atributos x a um rótulo de classe predefinido y ”.
- Segundo (AGGARWAL, 2015, p. 286), a maior parte dos algoritmos de classificação possui duas fases:
 - 1 Fase de treinamento;
 - 2 Fase de teste;
- A classificação binária é o caso mais simples das tarefas de classificação, pois nele só existem duas possibilidades de rótulo de classe e cada objeto de dados pertence, exclusivamente, a uma das classes.

- (ZHENG; CASARI, 2018) chamam de *manejo de atributos* o processo para extrair atributos passíveis de serem utilizados por classificadores da mineração de dados.
- (DONG; LIU, 2018, p. 3, tradução nossa) definem *feature engineering* como uma área que abrange “os tópicos de transformação de atributos, geração de atributos, extração de atributos, seleção de atributos, análise e avaliação de atributos, metodologias de manejo generalista e automatizado de atributos, e aplicações do manejo de atributos”.

Os tópicos da engenharia de atributos são:

- ▶ **Transformação de atributos;**
- ▶ **Geração de atributos;**
- ▶ **Seleção de atributos;**
- ▶ **Metodologias de manejo generalista e automatizado de atributos;**
- ▶ **Aplicações do manejo de atributos.**

Os tipos de atributos mais utilizados para representar documentos, segundo (FELDMAN; SANGER, 2006, p. 5–7):

- **Caracteres:** são as letras, números, e caracteres especiais presentes nos documentos utilizados para construir a semântica do mesmo;
- **Palavras:** são símbolos linguísticos nativos do espaço de atributos de um documento. Atributos a nível de palavra geralmente são palavras únicas selecionadas de um documento nativo, e também é possível que sejam utilizados todas as palavras de um documento para representá-lo;
- **Termos:** são palavras únicas e frases com mais de uma palavra selecionadas de um corpus de documentos nativos por meio de técnicas específicas para extração de termos;
- **Conceitos:** são os atributos gerados de modo manual, baseado em regras, ou via categorização híbrida feita no pré-processamento. Por exemplo, um documento sobre carros esportivos pode não incluir a palavra “automóvel”, mas este conceito pode ser utilizado para identificar e representar esse documento.

A criação de atributos consiste em derivar novos conjuntos de atributos que capturem, de forma mais efetiva, a informação carregada pelos dados (TAN; STEINBACH; KUMAR, 2014, p. 55).

Metodologias de criação de atributos apresentadas por (TAN; STEINBACH; KUMAR, 2014, p. 55–57):

- **Extração de atributos:** a criação de um novo conjunto de atributos a partir dos dados brutos.
- **Mapeamento dos dados para um novo espaço:** mudar completamente a visualização dos dados.
- **Construção de atributos:** os atributos presentes nos dados possuem a informação necessária para o processo de mineração, mas não estão na forma adequada, assim novos atributos podem ser construídos na forma adequada.

- Verdadeiros positivos (TP);
- Verdadeiros negativos (TN);
- Falsos positivos (FP);
- Falsos negativos (FN).

Tabela 2 – Matriz de confusão para uma tarefa de classificação binária, exibida com os totais para exemplos positivos e negativos.

Classe real	Classe prevista		Total
	<i>sim</i>	<i>não</i>	
	<i>sim</i>	<i>FP</i>	<i>N</i>
<i>não</i>	<i>FN</i>	<i>TN</i>	<i>P</i>
Total	P'	N'	$P + N$

Fonte: Tabela disponível em (HAN; KAMBER; PEI, 2011, p. 366).

- Precisão (p);
- Revocação (r);
- F -score;
- Acurácia (acc);
- F_{β} .

$$p = \frac{TP}{TP + FP} = \frac{TP}{P'} \quad (8)$$

$$r = \frac{TP}{TP + FN} = \frac{TP}{P} \quad (9)$$

$$F = \frac{2 \times p \times r}{p + r} \quad (10)$$

$$acc = \frac{TP + TN}{P + N} \quad (11)$$

$$F_{\beta} = \frac{(1 + \beta^2) \times p \times r}{\beta^2 \times p + r} \quad (12)$$

1 Introdução

2 Fundamentação Teórica

- Recuperação de Informação
- Mineração de Texto

3 Materiais e Métodos

4 Resultados

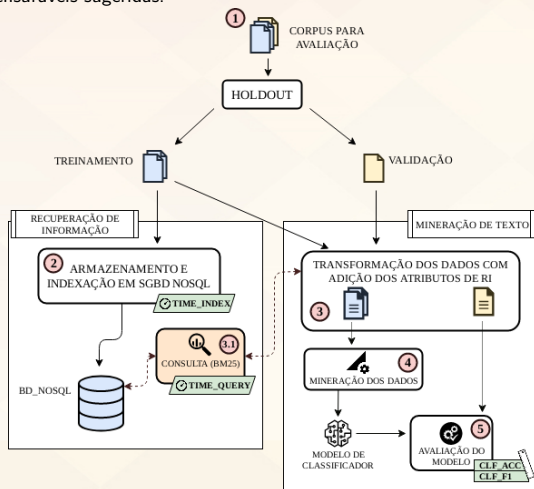
5 Considerações finais e trabalhos futuros

6 Referências

A metodologia proposta para avaliação do desempenho dos atributos criados consiste dos seguinte passos:

- 1 **Corpus para Avaliação**
- 2 **Armazenamento e Indexação em SGBD NoSQL**
- 3 **Transformação dos dados com adição dos atributos de RI**
 - 1 Consultas
- 4 **Avaliação do Modelo**
- 5 **Mineração dos Dados**

Figura 8 – Metodologia proposta para avaliação de desempenho, em verde estão as variáveis mensuráveis sugeridas.



Definidos dois corpus de competições promovidas pela PAN¹:

- **DB_AUTHORPROF - Author Profiling - PAN @ CLEF 2018:** Uma tarefa da competição *CLEF 2018* promovida pela PAN na classe de análise de autoria, a qual foca na identificação de gênero no Twitter em três linguagens distintas, inglês, espanhol, e árabe (PAN, 2018).
- **DB_HYPARTISAN - Hyperpartisan News Detection - PAN @ SemEval 2019 Task 4:** Esta tarefa da competição *SemEval 2019* promovida pela PAN consiste em, dada uma notícia, avaliar se esta segue uma argumentação hiperpartidária, que significa verificar se ela possui fidelidade cega, preconceituosa, ou irracional a um partido, grupo, causa, ou pessoa (PAN, 2019a).

Ambos se tratam de problemas de classificação binária, a classe real é a presença ou ausência de hiperpartidarismo em cada exemplo, ou masculino e feminino no primeiro corpus.

Os dois corpus possuem soluções de participantes nas competições da PAN que tem seu código fonte aberto e disponível em repositórios online.

¹Sigla da organização que se originou do *International Workshop on Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection* em 2007 (PAN'07 WORKSHOP, 2007).

Tabela 3 – Soluções encontradas de participantes da competição DB_AUTHORPROF.

Posição	Equipe	Repositório de código no site https://github.com/
2	daneshvar18	/SamanDaneshvar/pan18ap/
4	laporte18	/arthur-sultan-info/PAN2018-AP/
12	gouravdas18	/brajagopalce/PAN2018/
16	schaetti18	/nschaetti/PAN18-Author-Profiling/
21	raiyni18	/kraiyni/author-profiling-pan-clef-2018
23	karlgreen18	/jussikarlgreen/pan18/

Fonte: Classificações obtidas de (PAN, 2018), e repositórios encontrados pelo autor.

Tabela 4 – Soluções encontradas de participantes da competição DB_HYPERPARTISAN.

Posição	Equipe	Repositório de código no site https://github.com/
1	bertha-von-suttner	/GateNLP/semEval2019-hyperpartisan-bertha-von-suttner/
4	tom-jumbo-grumbo	/chialun-yeh/SemEval2019/
10	clint-buchanan	/hmc-cs159-fall2018/final-project-team-mvp-10000/
13	paparazzo	/ngannlt/semEval2019-hyperpartisan-paparazzo/
17	spider-jerusalem	/amal994/hyperpartisan-detection-task/
19	doris-martin	/ixa-ehu/ixa-pipe-doc/

Fonte: (PAN, 2019b).

Selecionadas as seguintes ferramentas de armazenamento e indexação:

- **TOOL_ELASTIC**: Elasticsearch 7.2 é o mecanismo distribuído de análise e busca baseado no Apache Lucene², desenvolvido em Java, e possui código aberto sob diversas licenças sendo a principal a Licença Apache³ ([ELASTICSEARCH:...](#), 2019; [ELASTICSEARCH...](#), 2019).
- **TOOL_ARANGO**: ArangoDB v3.4.6 é um banco de dados multi-modelo nativo, desenvolvido principalmente em C++ com extensões em JavaScript, que possui código-fonte aberto e possibilita modelos de dados flexíveis, tanto para documentos, gráficos, e valores-chave ([ARANGODB...](#), 2019; [ARANGODB...](#), 2019).
- **TOOL_ZETTAIR**: Zettair v0.9.3 é um mecanismo de busca de código-fonte aberto escrito na linguagem C, projetado para ser compacto e pesquisar rapidamente em texto, desenvolvido pelo Grupo de Mecanismos de Busca do Instituto Real de Tecnologia de Melbourne em 2009 ([RMIT UNIVERSITY](#), 2009).

Todas as três ferramentas implementam a função de ranqueamento BM25.

²Biblioteca de software livre e de código aberto para ferramentas de buscas em texto, escrita originalmente em Java ([APACHE...](#), 2019).

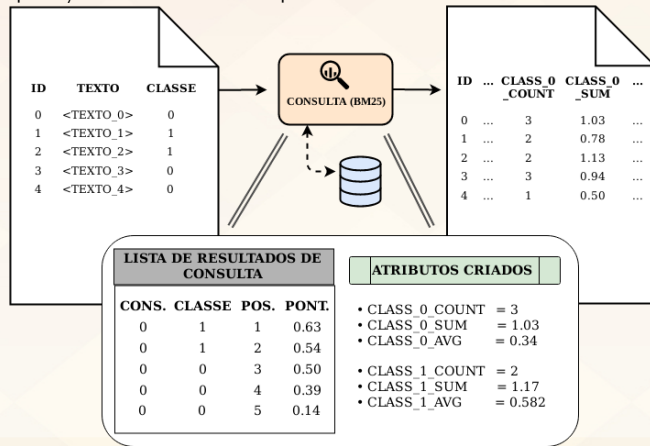
³Licença de software livre permissiva de autoria da Apache Software Foundation (ASF) ([OPEN...](#), 2015).

Os atributos são fundamentados na investigação de (WEREN, 2014).

Pressuposto:

- Os autores de um mesmo grupo de gênero ou idade tendem a usar termos semelhantes, e que a distribuição desses termos difere entre os grupos;
- Generalizado para outras classes de identificação de autoria, como por exemplo que autores de artigos hiperpartidários tendem a usar termos semelhantes, e a distribuição desses termos difere de autores não hiperpartidários.

Figura 9 – Metodologia de consulta aos BD para geração dos atributos sugeridos, exemplificação da lista de resultados para uma única consulta.



Fonte: O autor.

Tabela 5 – Atributos derivados de RI sugeridos.

Exemplo	Não faz parte da classe da tarefa	Faz parte da classe da tarefa
Agregação		
Média aritmética das pontuações	CLASS_0_BM25_AVG	CLASS_1_BM25_AVG
Contagem do número de resultados	CLASS_0_BM25_COUNT	CLASS_1_BM25_COUNT
Soma das pontuações	CLASS_0_BM25_SUM	CLASS_1_BM25_SUM

Fonte: O autor.

Medidas de desempenho computacional das ferramentas de armazenamento e indexação:

- **TIME_INDEX**: Tempo de execução para indexar o conjunto de treinamento de cada um dos corpus para avaliação elencados;
- **TIME_QUERY**: Tempo para consulta de cada exemplo do conjunto de teste e geração dos atributos sugeridos para o item específico.

Medidas de desempenho de classificador:

- **CLF_ACC**: Acurácia do classificador no conjunto de validação.
- **CLF_F1**: F_1 -score do classificador no conjunto de validação.

- 1 Introdução
- 2 Fundamentação Teórica
 - Recuperação de Informação
 - Mineração de Texto
- 3 Materiais e Métodos
- 4 Resultados
- 5 Considerações finais e trabalhos futuros
- 6 Referências

Tabela 6 – Configuração do computador de mesa utilizado neste estudo.

Processador	Intel(R) Core(TM) i7-4770 CPU @ 3.40GHz
Memória RAM	32370 MB
Sistema Operacional	Linux Mint 19.2 Tina
Placa-mãe	Gigabyte Z97-D3H
Gráficos	Mesa DRI Intel(R) Haswell Desktop
Disco	HP SSD EX920 512GB

Fonte: O autor.

Todo código fonte utilizado está disponível em:
<https://github.com/ruanmed/tcc-ii-ir-features-text-mining/>.

- Geração dos atributos de RI utilizando as ferramentas com os seguintes parâmetros:
 - ▶ $k_1 = 1.2$;
 - ▶ $k_3 = 0$;
 - ▶ $b = 0.75$; e
 - ▶ $\text{top-}k = 100$.
- Os scripts para o estudo foram feitos na linguagem de programação Python 3.7.5 com o ambiente VSCodium.
- RNG^a fixo para permitir reprodutibilidade.
- As dependências das soluções selecionadas foram instaladas: bibliotecas adicionais do Python e download de arquivos específicos para essas bibliotecas.

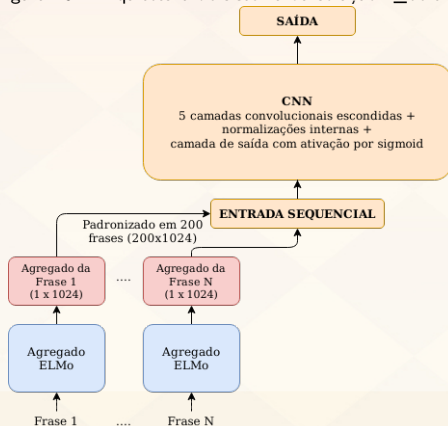
^agerador de números aleatórios

Foram selecionadas as seguintes soluções para reprodução e posterior adição dos atributos de RI:

- DB_HYPERPARTISAN:
 - ▶ 1_bertha (JIANG et al., 2019)
 - ▶ 4_tom (YEH; LONI; SCHUTH, 2019)
- DB_AUTHORPROF:
 - ▶ 2_daneshvar18 (DANESHVAR; INKPEN, 2018)

Solução do corpus DB_HYPERPARTISAN.

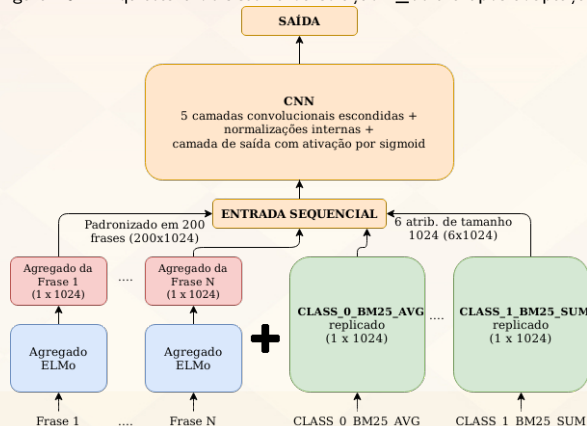
Figura 10 – Arquitetura de sistema da solução 1_bertha após adaptação.



◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡

Solução do corpus DB_HYPERPARTISAN.

Figura 10 – Arquitetura de sistema da solução 1_bertha após adaptação.



Fonte: O autor.

Solução do corpus DB_HYPERPARTISAN.

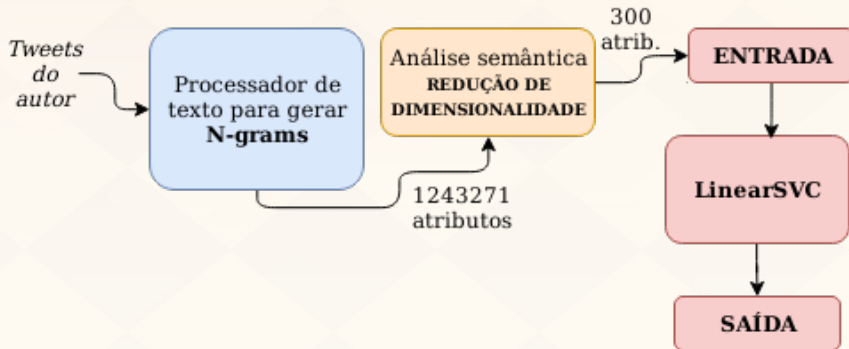
Figura 11 – Arquitetura de sistema da solução 4_tom após adaptação.



Fonte: O autor.

Solução do corpus DB_AUTHORPROF.

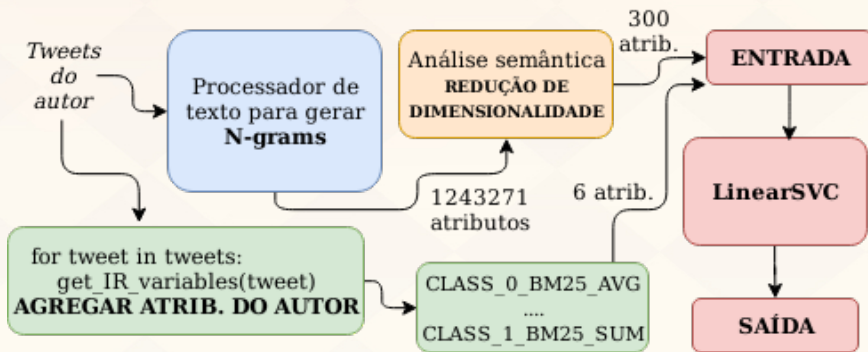
Figura 12 – Arquitetura de sistema da solução 2_daneshvar18 após adaptação.



Fonte: O autor.

Solução do corpus DB_AUTHORPROF.

Figura 12 – Arquitetura de sistema da solução 2_daneshvar18 após adaptação.



Fonte: O autor.

As principais características das soluções selecionadas estão na Tabela 7.

Tabela 7 – Resumo dos detalhes das soluções selecionadas.

Solução	Pré-processamento	Núm. atrib.	Redução dim.	Núm. atrib. após	Classificador
1_bertha	ELMo	200 x 1024	Não	200 x 1024	CNN (5 cam. esc.)
4_tom	GloVe	300	Não	300	SVC
2_daneshvar18	N-gram palavras+caracteres	1243271	Sim	300	LinearSVC

Fonte: O autor.

Resultados

Desempenho das ferramentas de armazenamento e indexação

O cálculo das medidas
TIME_INDEX e
TIME_QUERY foi auxiliado
com a implementação de uma
classe chamada
IndexToolManager que abstrai
a indexação e o cálculo das
variáveis de RI com as
ferramentas.

A utilização dela concentrou as
funções para acesso e
manipulação, quando
disponíveis, aos dados das
ferramentas de armazenamento
e indexação, centralização das
diferentes bibliotecas do
Python já disponíveis para

```
from arango import ArangoClient
from elasticsearch import Elasticsearch
class IndexToolManager:
    def __init__(self, indexName='default_index',
                 bm25_b=0.75, bm25_k1=1.2, bm25_k3=0.0, top_k=100):
        # Inicializa classes das ferramentas e conecta com os bancos de
    def log_result(self, itemKey, itemBody):
        # Insere um documento no banco de dados do Elasticsearch
    def get_documents(self, db='authorprof',
    ↪ documents_xml_folder='db_authorprof/en/',
        truth_txt='db_authorprof/truth.txt',
        append_class_to_id=False):
        # Generates a list with all documents from db formatted files.
    def calc_IR(self, result_df, positive_class='true'):
        # Calcula os atributos de RI sugeridos para a pesquisa
        # e retorna esses atributos em um dicionário do Python
    def insertArango(self, itemKey, itemBody):
        # Insere um documento na coleção 'indexName' do ArangoDB.
    def arango_query(self, query, ignore_first_result=False):
        # Consulta uma 'view' do ArangoDB e retorna um DataFrame do
        # Pandas com os resultados.
    def arango_get_IR_variables(self, query, positive_class='true',
    ↪ ignore_first_result=False):
        # Consulta uma 'view' do ArangoDB e retorna um 'dict' com os
        # atributos de RI.
```

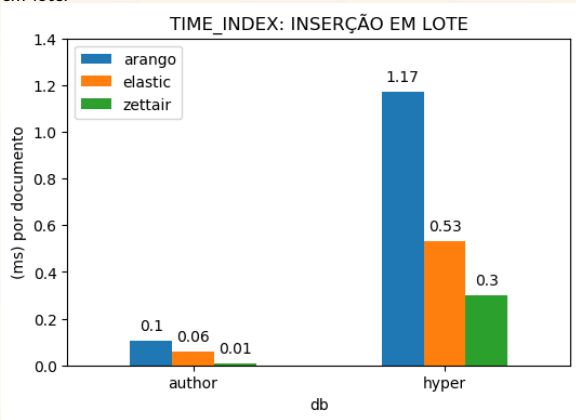
O cálculo da medida TIME_INDEX foi feito com o script `time_index.py`, o qual utilizou da classe *IndexToolManager* em duas funções feitas para executar a indexação dos corpus DB_AUTHORPROF e DB_HYPERPARTISAN nas 3 ferramentas, ArangoDB, Elasticsearch e Zettair. A função principal do script `time_index.py` possui nome *measure_TIME_INDEX*, e um trecho dela pode ser visto ao lado.

```
def measure_TIME_INDEX(normal=False, clean=False):
    mylogger.info('START OF TIME_INDEX MEASUREMENTS')
    exp_id = str(datetime.datetime.now())
    mylogger.info(exp_id)
    initial = time.time()
    if (clean):
        mylogger.info('CLEANING DATABASES')
        testTool = IndexToolManager(indexName=authorprof_db_name)
        testTool.clean_default()
        final = time.time()
        mylogger.info(f'CLEANING FINISHED: {final - initial}')
    tools = ['arango', 'elastic', 'zettair']
    dbs = ['authorprof', 'hyperpartisan', 'hyperpartisan_split_42']
    for db in dbs:
        mylogger.info('DB_' + db)
        for tool in tools:
            if (normal and tool != 'zettair'):
                index(idx_type='normal', db=db, tool=tool,
                      db_name=db, exp_id=exp_id)
            index(idx_type='bulk', db=db, tool=tool,
                  db_name=str(db+'_bulk'), exp_id=exp_id)
        mylogger.info(str(datetime.datetime.now()))
    mylogger.info('END OF TIME_INDEX MEASUREMENTS')
```

Resultados

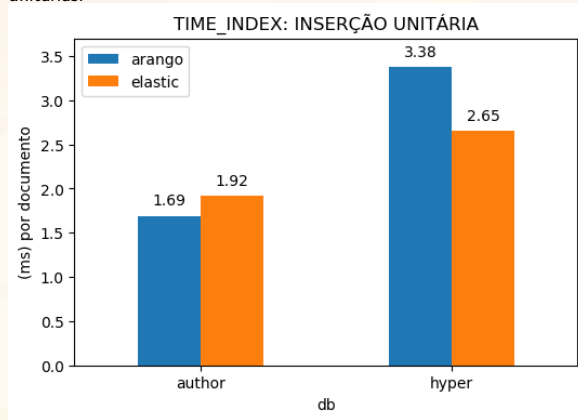
Tempo de indexação

Figura 13 – Medidas de desempenho TIME_INDEX mensuradas para as ferramentas de armazenamento e indexação, com inserções feitas em lote.



Fonte: O autor.

Figura 14 – Medidas de desempenho TIME_INDEX mensuradas para as ferramentas de armazenamento e indexação, com inserções unitárias.



Fonte: O autor.

Para medir o TIME_QUERY utilizando cada ferramenta durante a execução das soluções foi necessário adaptar os códigos das soluções para fazer o registro do tempo que cada consulta, e posterior geração dos atributos de RI, levou.

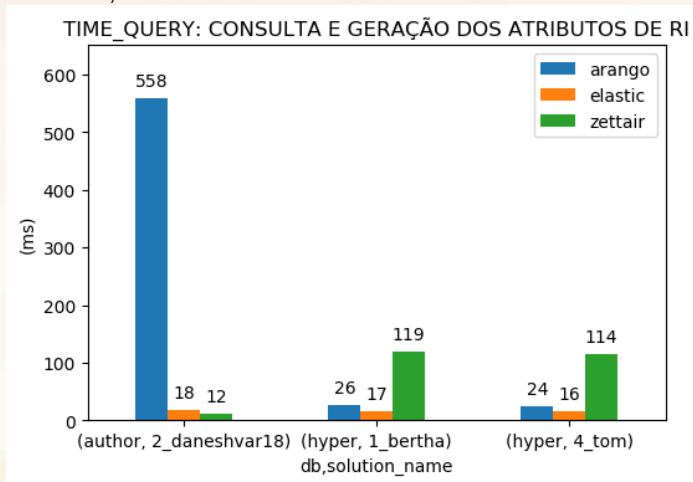
A utilização dos métodos implementados na classe *IndexToolManager* podem ser vistos no trecho disposto a seguir do script Python *feat_GloVe.ipynb* da solução 4_tom adaptada.

```
ir_top_k = 100
testTool = IndexToolManager(indexName=str(hyper_db_name), top_k=ir_top_k)
def gloveVectorize(glove, text):
    ...
    for text_id, t in enumerate(text):
        ...
        if (exp_dict['add_ir_variables']):
            initial = time.time()
            ign_first = ignore_first_result and (text_id in id1)
            if (exp_dict['tool'] == 'arango'):
                ir_variables = testTool.arango_get_IR_variables(
                    t, 'true', ignore_first_result=ign_first)
            elif (exp_dict['tool'] == 'elastic'):
                ir_variables = testTool.elastic_get_IR_variables(
                    t, 'true', ignore_first_result=ign_first)
            elif (exp_dict['tool'] == 'zettair'):
                ir_variables = testTool.zettair_get_IR_variables(
                    t, 'true', interactive=False,
                    ignore_first_result=ign_first)
            final = time.time()
            time_query_list.append(float(final-initial))
            ir_vars_dict = [
                ir_variables['CLASS_0_BM25_AVG'], ir_variables['CLASS_0_BM25_COUNT'],
                ir_variables['CLASS_0_BM25_SUM'], ir_variables['CLASS_1_BM25_AVG'],
                ir_variables['CLASS_1_BM25_COUNT'], ir_variables['CLASS_1_BM25_SUM']]
```

Resultados

Tempo de consulta

Figura 15 – Medidas de desempenho TIME_QUERY mensuradas para consulta e criação dos 6 atributos de RI sugeridos, utilizando as ferramentas de armazenamento e indexação.



Fonte: O autor.

Desempenho dos classificadores com atributos de RI

O desempenho dos classificadores foi mensurado conforme as medidas CLF_ACC e CLF_F1 estabelecidas.

As soluções originais foram reproduzidas, comparadas com os resultados divulgados nas páginas das competições, e então as soluções foram reproduzidas com as adaptações para incluíros atributos de RI gerados por cada uma das ferramentas.

DB_HYPERPARTISAN: reprodução das soluções originais

O conjunto de validação da competição não foi disponibilizado ao público. Foi realizado o *holdout* no conjunto de treinamento, 430 artigos foram separados para o conjunto de treinamento e 215 para validação.

As medidas das soluções reproduzidas se encontram na Tabela 8.

Tabela 8 – Comparação das medidas das soluções do corpus DB_HYPERPARTISAN divulgadas pela competição e das reproduções.

Solução	Acurácia		F_1 -Score	
	Competição	Reprodução	Competição	Reprodução
1_bertha	0,822	0,814	0,809	0,762
4_tom	0,806	0,809	0,790	0,707

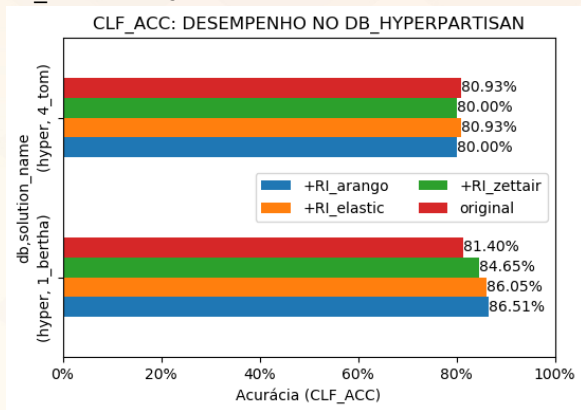
Fonte: O autor.

Relação de superioridade das soluções na competição se manteve nas reproduções.

Resultados

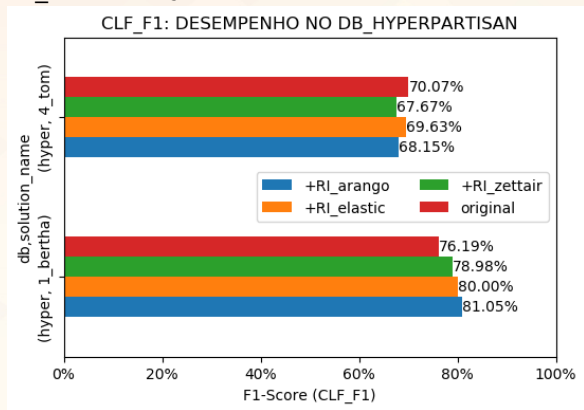
DB_HYPERPARTISAN: desempenho das soluções com atributos de RI

Figura 16 – Desempenho CLF_ACC das soluções do corpus DB_HYPERPARTISAN.



Fonte: O autor.

Figura 17 – Desempenho CLF_F1 das soluções do corpus DB_HYPERPARTISAN.

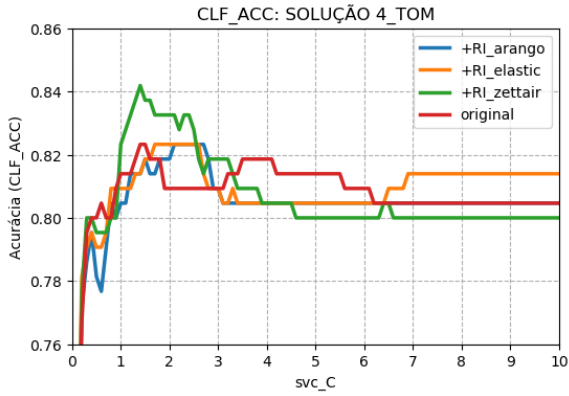


Fonte: O autor.

Resultados

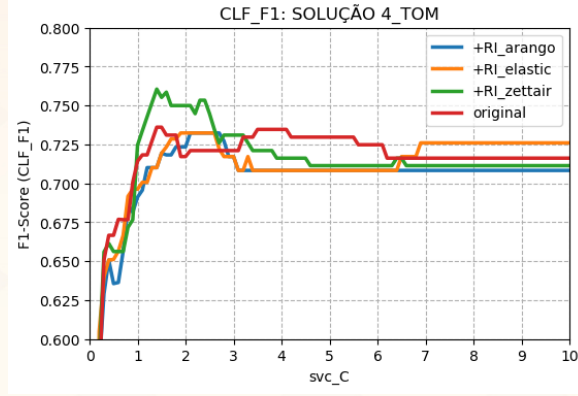
DB_HYPERPARTISAN: reprodução da solução 4_tom com diferentes C

Figura 18 – Desempenho CLF_ACC da solução 4_tom para diferentes valores de refinamento da função custo, parâmetro C do classificador SVC.



Fonte: O autor.

Figura 19 – Desempenho CLF_F1 da solução 4_tom para diferentes valores de refinamento da função custo, parâmetro C do classificador SVC.



Fonte: O autor.

DB_HYPERPARTISAN: reprodução da solução 4_tom com diferentes C

Nota-se que, mesmo com a adição dos atributos de RI, utilizando tanto o ArangoDB quanto o Elasticsearch, o classificador em nenhum momento consegue superar o melhor valor de C do classificador original, porém, com adição dos atributos de RI gerados pelo Zettair é atingido o melhor valor. Os valores máximos atingidos com os respectivos valores de C estão dispostos na Tabela 9.

Tabela 9 – Melhores valores de CLF_ACC e CLF_F1 do classificador SVC da solução 4_tom após reproduzida com diferentes valores de C.

Solução	C	Acurácia	F_1 -Score
original	1,4	0,8233	0,7361
+RI_arango	2,1	0,8233	0,7324
+RI_elastic	1,9	0,8233	0,7324
+RI_zettair	1,4	0,8419	0,7606

Fonte: O autor.

DB_AUTHORPROF: reprodução da solução original

A solução 2_daneshvar18 foi reproduzida sem nenhuma adaptação para comparação com os valores que se encontram na página da competição.

Na Tabela 10 está a medida de acurácia divulgada na página da competição (PAN, 2018) junto com as medidas da reprodução feitas do mesmo modo que a submissão original, treinamento com os tweets de 3000 autores de língua inglesa e validação com os tweets de 1900 autores de língua inglesa.

Tabela 10 – Comparação das medidas da solução 2_daneshvar18 do corpus DB_AUTHORPROF divulgadas pela competição e da reprodução da solução.

Solução	Acurácia		F_1 -Score	
	Competição	Reprodução	Competição	Reprodução
2_daneshvar18	0,8221	0,822105	–	0,820785

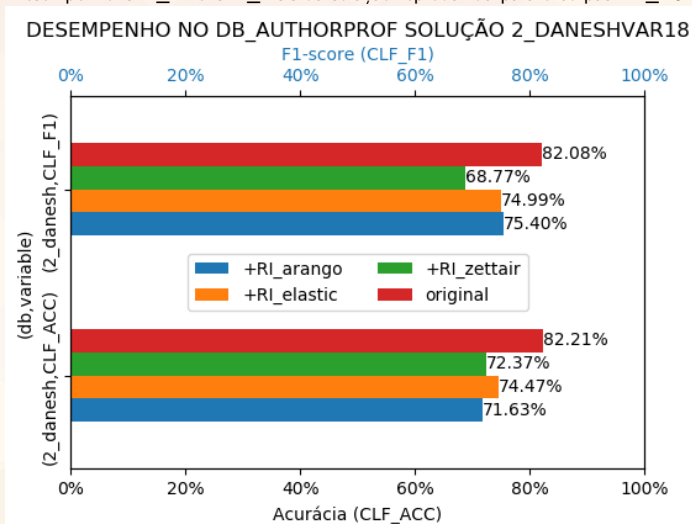
Fonte: O autor.

O resultado de acurácia da solução reproduzida truncado em 4 dígitos é o mesmo valor divulgado na página da competição.s

Resultados

DB_AUTHORPROF: desempenho da solução com atributos de RI

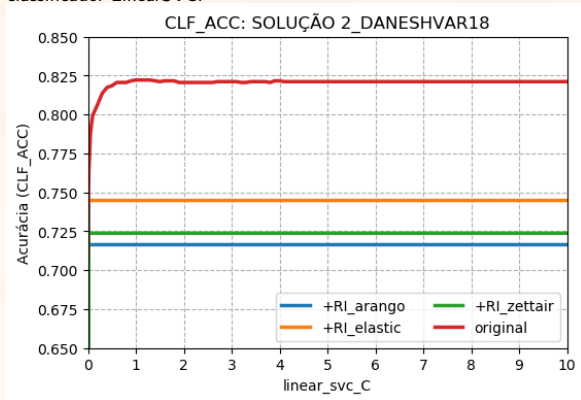
Figura 20 – Desempenho CLF_F1 e CLF_ACC da solução reproduzida para o corpus DB_AUTHORPROF.



Resultados

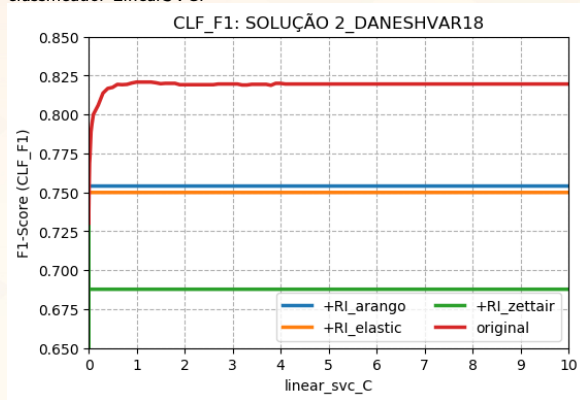
DB_AUTHORPROF: reprodução da solução 2_daneshvar18 com diferentes C

Figura 21 – Desempenho CLF_ACC da solução 2_daneshvar18 para diferentes valores de refinamento da função custo, parâmetro C do classificador LinearSVC.



Fonte: O autor.

Figura 22 – Desempenho CLF_F1 da solução 2_daneshvar18 para diferentes valores de refinamento da função custo, parâmetro C do classificador LinearSVC.



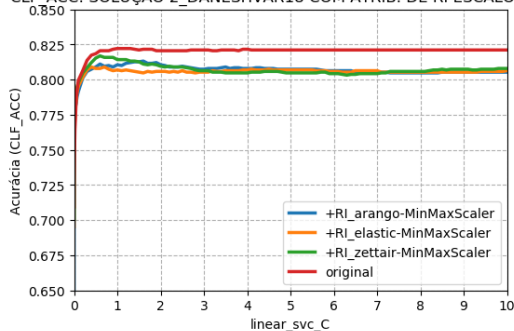
Fonte: O autor.

Resultados

DB_AUTHORPROF: reprodução da solução 2_daneshvar18 com atributos de RI escalonados

Figura 23 – Desempenho CLF_ACC da solução 2_daneshvar18 para diferentes valores de refinamento da função custo, parâmetro C do classificador LinearSVC com atributos de RI escalonados pelo MinMaxScaler.

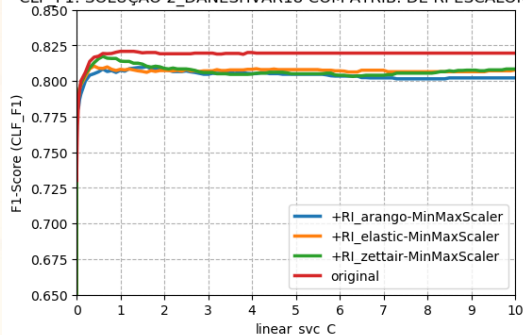
CLF_ACC: SOLUÇÃO 2_DANESHVAR18 COM ATRIB. DE RI ESCALONADOS



Fonte: O autor.

Figura 24 – Desempenho CLF_F1 da solução 2_daneshvar18 para diferentes valores de refinamento da função custo, parâmetro C do classificador LinearSVC com atributos de RI escalonados pelo MinMaxScaler.

CLF_F1: SOLUÇÃO 2_DANESHVAR18 COM ATRIB. DE RI ESCALONADOS



Fonte: O autor.

DB_AUTHORPROF: reprodução da solução 2_daneshvar18 com atributos de RI escalonados

Ainda assim o desempenho continuou abaixo da reprodução da solução original.

Na Tabela 11 podem ser vistos os valores máximos atingidos com os respectivos valores de C.

Tabela 11 – Melhores valores de CLF_ACC e CLF_F1 do classificador LinearSVC da solução 2_daneshvar18 após reproduzida com diferentes valores de C, com atributos de RI escalonados pelo MinMaxScaler.

Solução	C	Acurácia	F_1 -Score
original	1,0	0,822105	0,820785
+RI_arango	1,6	0,813158	0,810059
+RI_elastic	0,4	0,808947	0,810444
+RI_zettair	0,6	0,816842	0,817227

Fonte: O autor.

O melhor desempenho com os atributos de RI é quando gerado pelo Zettair, no entanto ainda assim fica abaixo da solução original.

- 1 Introdução
- 2 Fundamentação Teórica
 - Recuperação de Informação
 - Mineração de Texto
- 3 Materiais e Métodos
- 4 Resultados
- 5 Considerações finais e trabalhos futuros
- 6 Referências

- Avaliação do desempenho de atributos oriundos da área de Recuperação de Informação em tarefas de Mineração de Texto, com enfoque em atributos gerados pela função BM25;
- Zettair foi a ferramenta testada mais rápida, porém com limitações;
- Elasticsearch foi mais rápido no geral;
- ArangoDB apresenta lentidão para calcular BM25 para corpus grandes;
- Comprovação das medidas do ranking final da competição do corpus DB_AUTHORPROF para a solução 2_daneshvar18;
- Falta do conjunto de validação final da competição impossibilitou comprovar as medidas do ranking final do corpus DB_HYPERPARTISAN;
- Desempenho de classificador com atributos de RI:
 - ▶ Solução 1_bertha do DB_HYPERPARTISAN apresentou ganho de desempenho ao adicionar os atributos, das 3 ferramentas;
 - ▶ Não houve ganho inicial de desempenho com a solução 4_tom utilizando os atributos de RI;
 - ▶ Adicionar os atributos gerados pelo Zettair produz um ganho de desempenho quando o melhor valor do parâmetro C, do SVC da solução 4_tom, é selecionado;
 - ▶ O classificador LinearSVC da solução 2_daneshvar18 não obteve nenhum ganho de desempenho, nem mesmo ao ser feita a análise e seleção do parâmetro C do classificador;
 - ▶ Mesmo com o reparo de magnitude por meio do escalonador MinMaxScaler, e nova análise e seleção do parâmetro C do classificador, não foi obtido ganho de desempenho na solução 2_daneshvar18.








Considerações finais e trabalhos futuros

Desempenho dos atributos de RI









- Os 6 atributos de RI sugeridos nesse estudo proporcionaram **ganho de desempenho somente** para classificações do corpus DB_HYPERPARTISAN, um **corpus pequeno com documentos extensos**. Os atributos de RI funcionaram melhor com o classificador CNN, e para o classificador SVC somente o Zettair proporcionou ganho de desempenho;
- No corpus DB_AUTHORPROF, um **corpus grande com documentos curtos**, com o classificador LinearSVC, **os atributos de RI causaram perda de desempenho**;
- São necessários mais estudos para corroborar os resultados obtidos, com outros corpus e classificadores.


- Reproduzir a solução 2_daneshvar18 com outras variações nos parâmetros do classificador LinearSVC;
- Reproduzir mais uma das soluções do corpus DB_AUTHORPROF disponíveis online;
- Analise de diferentes valores que influenciem somente nos atributos de RI gerados, o top- k e os parâmetros k_1 , k_3 e b reproduzindo-os para os mesmos corpus.
- Separação dos corpus em 3 pedaços para trabalhar com um índice isolado para geração dos atributos de RI. O primeiro pedaço seria utilizado para para treinamento do classificador, o segundo pedaço seria utilizado como o índice para geração dos atributos derivados da função BM25 para o primeiro pedaço, e o terceiro pedaço ficaria para teste/validação do classificador.




- 1 Introdução
- 2 Fundamentação Teórica
 - Recuperação de Informação
 - Mineração de Texto
- 3 Materiais e Métodos
- 4 Resultados
- 5 Considerações finais e trabalhos futuros
- 6 Referências

-  AGGARWAL, C. C. **Data Mining: The Textbook**. [S.l.]: Springer, 2015. p. 734. ISBN 9783319141411. DOI: <https://doi.org/10.1007/978-3-319-14142-8>. Disponível em: <<https://www.springer.com/gp/book/9783319141411>>. Acesso em: 5 ago. 2019.
-  ARANGODB - GitHub - arangodb/arangodb at 3.4.2. ArangoDB Inc. 2019. Disponível em: <<https://web.archive.org/web/20190712211053/https://github.com/arangodb/arangodb/tree/3.4.2>>. Acesso em: 12 jul. 2019.
-  ARANGODB v3.4.6 Documentation: Introduction to ArangoDB Documentation. ArangoDB Inc. 2019. Disponível em: <<https://web.archive.org/web/20190712211033/https://www.arangodb.com/docs/3.4/index.html>>. Acesso em: 12 jul. 2019.
-  BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval: The Concepts and Technology Behind Search**. 2. ed. Essex, England: Addison-Wesley, 2011. p. 913. ISBN 978-0-321-41691-9.
-  DANESHVAR, S.; INKPEN, D. Gender Identification in Twitter using N-grams and LSA—Notebook for PAN at CLEF 2018. In: _____. **CLEF 2018 Evaluation Labs and Workshop – Working Notes Papers, 10-14 September, Avignon, France**. [S.l.]: CEUR-WS.org, set. 2018. Disponível em: <<http://ceur-ws.org/Vol-2125/>>.
-  DONG, G.; LIU, H. **Feature Engineering for Machine Learning and Data Analytics**. Edição: Guozhu Dong e Huan Liu. 1. ed. Boca Raton, FL, EUA: CRC Press, abril 2018. p. 400. ISBN 9781315181080. DOI: <https://doi.org/10.1201/9781315181080>. Disponível em: <<https://www.taylorfrancis.com/books/e/9781315181080>>. Acesso em: 30 jul. 2019.
-  ELASTICSEARCH Reference [7.2] » Elasticsearch introduction. Elastic. 2019. Disponível em: <<https://web.archive.org/web/20190712125432/https://www.elastic.co/guide/en/elasticsearch/reference/7.2/elasticsearch-intro.html>>. Acesso em: 12 jul. 2019.

-  ELASTICSEARCH: A Distributed RESTful Search Engine - GitHub - elastic/elasticsearch at 7.2. Elastic. 2019. Disponível em: <<https://web.archive.org/web/20190712125715/https://github.com/elastic/elasticsearch/tree/7.2>>. Acesso em: 12 jul. 2019.
-  FELDMAN, R.; DAGAN, I. Knowledge Discovery in Textual Databases (KDT). In: _____. 20–21 ago. 1995, Montréal, Québec, Canada. **Proceedings of the First International Conference on Knowledge Discovery and Data Mining**. Montréal, Québec, Canada: AAAI Press, 1995. Disponível em: <<http://dl.acm.org/citation.cfm?id=3001335.3001354>>.
-  FELDMAN, R.; SANGER, J. **Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data**. 1. ed. [S.l.]: Cambridge University Press, 2006. p. 410. ISBN 0521836573, 9780521836579.
-  HAN, J.; KAMBER, M.; PEI, J. **Data Mining: Concepts and Techniques**. 3rd. [S.l.]: Morgan Kaufmann Publishers Inc., 2011. p. 703. ISBN 0123814790, 9780123814791.
-  JIANG, Y. et al. Team Bertha von Suttner at SemEval-2019 Task 4: Hyperpartisan News Detection using ELMo Sentence Representation Convolutional Network. In: PROCEEDINGS of the 13th International Workshop on Semantic Evaluation. Minneapolis, Minnesota, USA: Association for Computational Linguistics, jun. 2019. p. 840–844. DOI: [10.18653/v1/S19-2146](https://doi.org/10.18653/v1/S19-2146). Disponível em: <<https://www.aclweb.org/anthology/S19-2146>>.
-  JO, T. **Text Mining: Concepts, Implementation, and Big Data Challenge**. 1. ed. [S.l.]: Springer International Publishing, 2018. p. 373. (Studies in Big Data, 45). ISBN 9783319918143. Disponível em: <<https://doi.org/10.1007/978-3-319-91815-0>>. Acesso em: 25 jul. 2019.
-  KODRATOFF, Y. Knowledge Discovery in Texts: A Definition, and Applications. In: PROCEEDINGS of the 11th International Symposium on Foundations of Intelligent Systems. [S.l.: s.n.], 1999. ISBN 3-540-65965-X. Disponível em: <<http://dl.acm.org/citation.cfm?id=646358.689959>>.

-  KOWALSKI, G. **Information Retrieval Architecture and Algorithms**. 1. ed. Ashburn, VA, USA: Springer, dezembro 2010. p. 305. ISBN 9781441977151.
-  KWARTLER, T. **Text Mining in Practice With R**. 1. ed. Chichester, Inglaterra: John Wiley & Sons, jul. 2017. p. 307. ISBN 9781119282013. Disponível em: <<https://www.wiley.com/en-us/Text+Mining+in+Practice+with+R-p-9781119282013>>. Acesso em: 4 ago. 2019.
-  APACHE LuceneTM 8.1.1 Documentation. Lucene. 2019. Disponível em: <https://web.archive.org/web/20190712130658/https://lucene.apache.org/core/8_1_1/index.html>. Acesso em: 12 jul. 2019.
-  LYMAN, P.; VARIAN, H. R. How Much Information 2003? <http://www.sims.berkeley.edu/research/projects/how-much-info-2003/>, 2003. Disponível em: <<http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/>>. Acesso em: 25 jul. 2019.
-  MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. **Introduction to Information Retrieval**. 1. ed. Cambridge, Reino Unido: Cambridge University Press, 2008. p. 482. ISBN 9780511414053.
-  OPEN Source Licensing Guide. New Media Rights. 2015. Disponível em: <https://web.archive.org/web/20190712132826/https://www.newmediarights.org/open_source/new_media_rights_open_source_licensing_guide>. Acesso em: 12 jul. 2019.
-  PAN. **Author Profiling PAN @ CLEF 2018**. 2018. Disponível em: <<https://web.archive.org/web/20190712001219/https://pan.webis.de/clef18/pan18-web/author-profiling.html>>. Acesso em: 11 jul. 2019.
-  _____. **Hyperpartisan News Detection PAN @ SemEval 2019**. 2019. Disponível em: <<https://web.archive.org/web/20190711231940/https://pan.webis.de/semeval19/semeval19-web/index.html>>. Acesso em: 11 jul. 2019.

-  PAN. **Hyperpartisan News Detection: Leaderboard - PAN @ SemEval 2019**. 2019. Disponível em: <<https://web.archive.org/web/20190803162634/https://pan.webis.de/semeval19/semeval19-web/leaderboard.html>>. Acesso em: 3 ago. 2019.
-  PAN'07 WORKSHOP. **International Workshop on Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection (PAN)**. 2007. Disponível em: <<https://web.archive.org/web/20190711212207/https://www.uni-weimar.de/medien/webis/events/pan-07/pan07-web/>>. Acesso em: 11 jul. 2019.
-  RMIT UNIVERSITY. **Zettair Homepage: Introduction**. 2009. Disponível em: <<https://web.archive.org/web/20190713000005/http://www.seg.rmit.edu.au/zettair/index.html>>. Acesso em: 12 jul. 2019.
-  SANDERSON, M.; CROFT, W. B. The History of Information Retrieval Research. **Proceedings of the IEEE**, v. 100, Special Centennial Issue, p. 1444–1451, maio 2012. ISSN 0018-9219. DOI: 10.1109/JPROC.2012.2189916.
-  TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining**. Harlow, Essex, Inglaterra: Pearson, 2014. p. 732. ISBN 9781292026152. Disponível em: <<https://www.pearsonelt.ch/HigherEducation/Pearson/EAN/9781292026152/Introduction-to-Data-Mining-Pearson-New-International-Edition>>.
-  WEREN, E. R. D. **Atribuição de Perfis de Autoria**. Nov. 2014. Diss. (Mestrado) – Universidade Federal do Rio Grande do Sul, Porto Alegre, BR-RS. Disponível em: <<http://hdl.handle.net/10183/108592>>. Acesso em: 18 jan. 2019.
-  WIKIPÉDIA. **Portal:Conteúdo destacado**. 2019. Disponível em: <https://web.archive.org/web/20190505054741/https://pt.wikipedia.org/wiki/Portal:Conte%C3%BAdo_destacado>. Acesso em: 5 mai. 2019.

-  YEh, C.-L.; LONI, B.; SCHUTH, A. Tom Jumbo-Grumbo at SemEval-2019 Task 4: Hyperpartisan News Detection with GloVe vectors and SVM. In: PROCEEDINGS of the 13th International Workshop on Semantic Evaluation. Minneapolis, Minnesota, USA: Association for Computational Linguistics, jun. 2019. p. 1067–1071. DOI: [10.18653/v1/S19-2187](https://doi.org/10.18653/v1/S19-2187). Disponível em: <https://www.aclweb.org/anthology/S19-2187>.
-  ZHAI, C.; MASSUNG, S. **Text Data Management and Analysis: A Practical Introduction to Information Retrieval and Text Mining**. 1. ed. [S.l.]: ACM Books, 2016. p. 510. ISBN 9781970001174.
-  ZHENG, A.; CASARI, A. **Feature Engineering for Machine Learning**. Edição: Rachel Roumeliotis e Jeff Bleiel. 1. ed. Sebastopol, CA, EUA: O'Reilly, 2018. p. 200. ISBN 9781491953242. Disponível em: <http://oreilly.com/catalog/errata.csp?isbn=9781491953242>. Acesso em: 8 ago. 2019.

Por sua atenção!