PyCPI: A package for capturing, NLP preprocessing, and Solr indexing of mandatory reports in the Brazilian Capital Market

C. N da Silva, T.*, Sbruzzi, Elton, Leles, Michel, Curtis, V. Vitor

Instituto Tecnológico de Aeronáutica, São José dos Campos - SP / Brazil

Abstract

The collection and preprocessing of unstructured data, such as financial reports, often require significant effort and time. Dealing with heterogeneous formats and a large volume of available data, due to the modernization of financial information systems, adds to the complexity. The goal of the PyCPI package is to simplify this process by providing a pipeline that captures financial reports provided by the Brazilian Securities and Exchange Commission (CVM, 2023), performs preprocessing, including PDF-to-text conversion, tokenization, and tensor generation using the Transformer (Beltagy et al., 2020) architecture. It then indexes the metadata using Apache Solr (Apache, 2021) for efficient and fast information retrieval. Another functionality offered by the PyCPI package is text comparison, which measures the similarity between two tensors. The package also includes features such text summarization, highlighting the most significant sentences in the text, among others. PyCPI aims to contribute to the advancement of finance studies through the integration of unstructured data.

Keywords: Financial text analysis, Text mining in finance, Financial event extraction

Code Metadata

*Corresponding author

Email addresses: `thiagotcns@ita.br` (C. N da Silva, T.), `elton@ita.br` (Sbruzzi, Elton), `michel.leles@gp.ita.br` (Leles, Michel), `vitor.curtis@gp.ita.br` (Curtis, V. Vitor)

C1	Current code version	v1.0.0
C2	Permanent link to code/repository used for this code version	`https://github.com/fico-ita/po_245_2023_T4.git`
C3	Legal Code License	Apache 2.0
C4	Code versioning system used	Git
C5	Software code languages, tools, and services used	Python
C6	Compilation requirements, operating environments & dependencies	*python* ≥3.11, *pandas* ≥1.3.0, *requests* ≥2.26.0, *unidecode* ≥1.3.6, $transformers$ ≥4.30.2, *torch* ≥2.0.1, *pypdf*2 ≥3.0.1, *spacy* ≥3.5.3, *ipykernel* ≥6.23.3, $scikit - learn$ ≥1.2.2, *sentencepiece* ≥0.1.99
C7	If available Link to developer documentation/manual	`https://github.com/fico-ita/po_245_2023_T4/blob/main/docs/`
C8	Support email for questions	thiagotcns@ita.br

Table 1: Code metadata

1. Motivation and significance

The growing number of listed companies in the stock market and the advancements in information systems have led to a significant increase in the volume of available data for company analysis. However, a large portion of this data is unstructured text, which often contains valuable insights. Previous studies have shown that financial reports not only reflect historical information but also provide indications of future performance (Zhang et al., 2018).

Harnessing textual data to enhance the modeling of financial market dynamics has become a common practice in the industry. In this work, we propose a comprehensive package specifically tailored for the efficient availability of financial reports submitted by companies to the Brazilian Securities and Exchange Commission (CVM).

The first stage of PyCPI involves the collection of raw data directly from the CVM. Subsequently, the captured data undergoes a comprehensive preprocessing and transformation process to ensure it is in a suitable format for further analysis. We also organize and categorize the reports based on the financial institution and publication year, enabling efficient retrieval and analysis.

After the capture stage, PyCPI employs advanced NLP techniques to generate tokens and tensors, enabling the extraction of meaningful information from the reports. This step facilitates the identification of patterns and trends that can contribute to more accurate financial analysis and decision-making.

Finally, PyCPI leverages Solr, a powerful indexing tool, to enable fast and precise search capabilities for users. The metadata indexing includes essential information such as company names, report descriptions, reference dates, publication years, and the availability of raw and preprocessed reports. This ensures that users can easily navigate and retrieve the relevant information they need for their analysis.

1.1. Literature Review

Availability and analysis of unstructured data have become increasingly relevant for organizations, particularly in the context of financial reports. With advancements in technology, it is now possible to obtain and process this data in an automated manner. Studies indicate that 71% of financial institutions use big data analytics to gain a competitive advantage (Pejić et al., 2019). Recent research shows that relying solely on historical data has become more challenging. According to Zhang et al. (2018), using the Hurst exponent, the correlation between daily returns of the Dow Jones and its historical data has declined since the 1990s. In this context, the analysis of unstructured data such as financial reports has become a competitive advantage.

1.1.1. Financial Reports

Financial reports are the most representative documents for analyzing a company's communications about its performance, financial health, and future prospects. These reports are mandatory and are produced annually to share information with the market. Sometimes, they contain clues about future performance and valuable information that companies may try to hide from investors (Masson et al., 2020). Signals of changes in a company's financial position may appear in reports before they can be identified in the financial numbers.

In Brazil, the Brazilian Securities and Exchange Commission (CVM) plays a crucial role in regulating and supervising these reports, ensuring transparency and reliability of the disclosed information. One of its responsibilities is to establish rules and guidelines for the disclosure of financial reports by companies (CVM, 2023). Studies conducted with companies listed on B3 indicate that reports related to assemblies, traded and retained securities, and market communications are the most commonly disclosed in terms of quantity. The frequency of disclosure varies depending on the company's segment on B3, with companies classified as Level 1 of corporate governance having more frequent disclosure due to regulatory requirements (Checon et al., 2021).

1.1.2. Text Preprocessing

Preprocessing stages using various methods have a substantial impact on text mining techniques (Pejić et al., 2019). It involves techniques and procedures applied to prepare raw textual data for analysis and extraction of meaningful information. There are three main approaches to a text preprocessing pipeline (Ferrario et al., 2020):

1° Classical approach: utilizes traditional natural language processing techniques such as "bag-of-words" and "bag-of-part-of-speech."

2° Modern approach: employs word embedding techniques such as Word2Vec, GloVe, ELMo, and FastText, which capture distributed representations of words.

3° Contemporary approach: introduces the use of recurrent neural networks (RNNs), convolutional neural networks (CNNs), and Transformer-based models like Longformer.

The contemporary approach aims to automate the discovery of representations required for the classification task, reducing the need for manual preprocessing steps (Ferrario et al., 2020). However, token generation and vectorization tasks remain essential for feeding text processing models.

Token generation is a critical step in text preprocessing, involving the segmentation of text into smaller semantic units. Tokens can represent words, phrases, sentences, or even individual characters, depending on the desired level of granularity (Borggreve, 2022).

This step is crucial to preserve the structure and context of the original text during subsequent analysis and modeling (Beltagy, I et al., 2020). In Longformer during token generation, the text is divided into smaller units, such as words or subwords, depending on the tokenizer used. Longformer employs a WordPiece-based approach to break words into subwords based on a predefined vocabulary. This technique enables capturing flexibility and generalization of words, particularly in the case of compound words, neologisms, or languages with complex morphological structures.

An essential feature of Longformer is its ability to handle long documents that may exceed the fixed token limit used in conventional Transformer models. To address this, Longformer employs a strategy called "sliding window." In this approach, the text is divided into smaller overlapping segments, called windows, which are sequentially fed into the model. This way, Longformer can process long sequences without losing contextual connections between parts of the text.

After token generation, the texts are transformed into dense vectors in a high-dimensional space. Each token is assigned a unique index and represented by a binary vector with all dimensions set to zero except for the dimension corresponding to its index, which is set to one. This binary representation allows the model to identify and differentiate each token during processing.

1.1.3. Data Catalog and Metadata Indexing

Metadata indexing is a critical step in enabling efficient organization and retrieval, particularly in environments with large volumes of data for reuse (Nakandala et al., 2015). Metadata refers to descriptive information that provides context and structure to the data, facilitating its organization and retrieval. It describes characteristics such as data type, format, author, creation date, and other relevant attributes.

Apache Solr is a widely used tool for metadata indexing and information retrieval. It is an open-source search and text indexing platform based on the Lucene library, offering powerful capabilities for efficient management of large volumes of data (Apache, 2021).

By utilizing Apache Solr for metadata indexing, data is structured and organized into indexes that enable efficient and fast information retrieval. After metadata indexing, advanced queries can be performed in Apache Solr to retrieve documents that match specific search criteria.

Solr provides powerful search features, including advanced text search, result filtering, relevance ranking, and pagination, among other capabilities. For example, it is possible to filter search results by author, creation date, or any other relevant field present in the indexed metadata.

The use of Apache Solr for metadata indexing brings significant advantages for information retrieval in large datasets. Its scalability, efficiency, and advanced search capabilities make it a popular choice for implementing search and data retrieval systems.

Several studies and scientific works have explored the use of Apache Solr for metadata indexing in different contexts and application domains. These researches have demonstrated the effectiveness and usefulness of Solr as a robust and flexible solution for indexing and efficient retrieval of metadata (K. Guntupally et al., 2020).

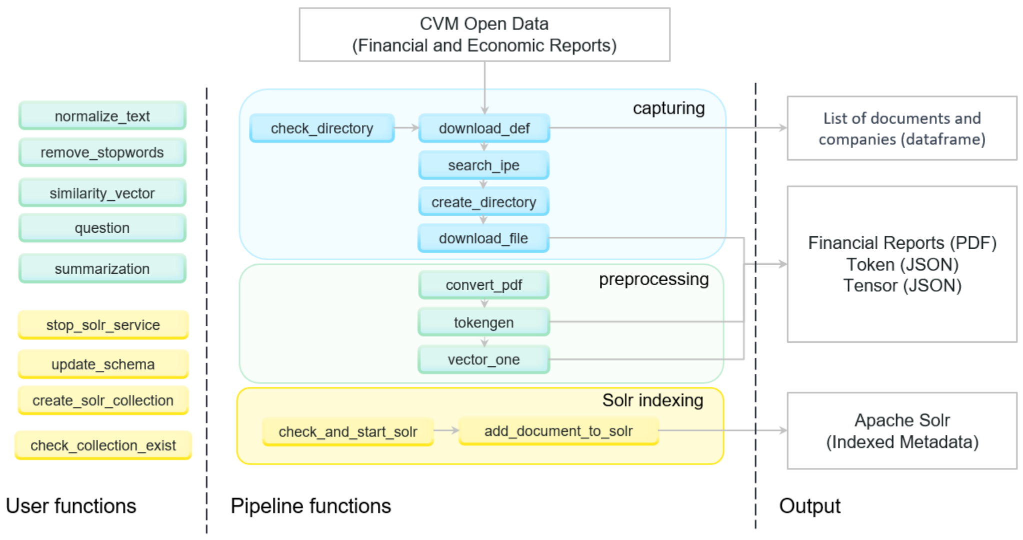

Figure 1: Software Architecture PyCPI

2. Software description

2.1. Software Architecture

PyCPI package is implemented in Python 3 and provides a set of functions to facilitate common steps within a data pipeline, including data capture, preprocessing, and metadata indexing. Moreover, it encompasses functions that enable users to manipulate pre-defined collections in Apache Solr and implements classical text preprocessing techniques, such as normalization and stop word removal. The package offers functions for text summarization and comparisons.

PyCPI adopts a procedural programming style, and the provided functions are organized as illustrated in Fig. 1. These functions are designed at a high level, allowing users to take advantage of a wide array of functionalities offered.

The code is developed using well-established Python libraries, such as sklearn (Pedregosa et al., 2011), as well as modern specialized libraries for deep learning, such as PyTorch (Paszke et al., 2019), transformers (Beltagy et al., 2020), among others.

2.2. Software functionalities

The PyCPI package offers a comprehensive set of functionalities for the automation and efficient management of economic and financial documents. Focused on capture, preprocessing, and indexing pipelines, PyCPI simplifies the process of collecting relevant information and enables detailed analysis of these documents. The following list describes some of the most important functions:

- `pipeline_cpi()`: This function serves as the centerpiece of the PyCPI package, as it coordinates and automates the entire process of capturing, preprocessing, and indexing economic and financial documents.

The function `pipeline_cpi(company, year)` orchestrates the following functions:

- `check_directory()`:Verifying whether the directory provided by the user is valid, returning the complete path for document storage.
- `search_ipe(year)`: Obtaining the Information Periódicas (IPE) data for a specific company and year.
- `download_def(company, year, root)`: Downloading specific files based on the provided company and year.
- `convert_pdf(doc)`: Converting a PDF file into text.
- `tokengen(text)`: This function is orchestrated by the `pipeline_cpi` and utilizes the LongformerTokenizer to generate a list of tokens from a given input text, including special tokens such as [CLS] (start) and [SEP] (end) to mark the beginning and end of the text.
- `vector_one(text)`: Also orchestrated by the `pipeline_cpi`, this function generates a representation vector for a text using the pre-trained Longformer model. This vector representation is obtained by tokenizing the text, dividing it into smaller parts if necessary, and subsequently using the model to generate embedding vectors for each part of the text. Finally, these vectors are concatenated, creating a single vector that represents the input text. This functionality is essential for the efficient and advanced processing of text in capture, preprocessing, and indexing pipelines for economic and financial documents, leading to improved performance in machine learning and natural language processing applications.
- `add_document_to_solr(collection_name, metadatas)`: This function indexes the metadata of the financial reports to be stored. These metadata contain information such as name (the name of the economic and financial report), type (document type), cod_cvm (company code in the CVM), content (the content description), year (publication year),

date (reference date of that report), Keywords (keywords of that report), size (document size), and fields tensor, tokens, file containing the paths where the documents with their respective contents are stored.

The package also includes other functions and libraries that allow users to perform more specific tasks, such as question answering and automatic summarization, as well as create collections in Apache Solr, among others:

- `remove_stopwords(tokens)`: Removes stopwords from a list of tokens using the spaCy model for the Portuguese language.
- `normalize_text(text)`: Normalizes the provided text by converting it to lowercase and removing special characters and accents.
- `similarity_vector(a, b)`: Calculates the similarity between two vectors using the cosine similarity metric.
- `question(text, question)`: Answers questions based on an input text using the pre-trained BERT model for question-answering.
- `summarization(text)`: Generates a summary of the provided text using the pre-trained T5 (Text-to-Text Transfer Transformer) model.
- `stop_solr_service()`: Shuts down the running Solr service.
- `check_collection_exists(col_name)`: Verifies if a collection exists on the Solr server.
- `create_solr_collection(col_name)`: Creates a new collection on the Solr server and updates the associated schema.
- `update_schema(data, collection_name)`: Updates the schema of the Solr collection with the provided fields and configurations.

3. Illustrative examples

The PyCPI package can be used with a single call to extract all reports related to that company and year. Before making the call, it is important to install Apache Solr. Note: It's essential to install Apache Solr before using PyCPI. After the installation, update line 35 in the code, modifying the "slrb" variable to set the correct Solr repository path.

```
# Path to the Solr bin directory
slrb = Path("C:\local\solr\bin\")
```

Before running the pipeline for the first time, it is necessary to create a "collection," a kind of library that will store the metadata:

Check and start the Solr service if it is inactive

```
import PyCPI
PyCPI.check_and_start_solr()
Checking the status of Solr...
Starting Solr...
Solr started successfully.
(0, 'Solr␣started␣successfully.')
```

Create the collection with the name "relatorio-financeiro"

```
PyCPI.create_solr_collection("relatorio_financeiro")
Schema updated successfully.
```

Once you have the collection, simply execute the pipeline, providing the company name and the publication year of the report:

When executing the pipeline, you will be prompted to provide the storage directory for the files.

```
PyCPI.pipeline("Magazine␣Luiza", 2023)
Root directory for document storage <path>
Pipeline execution started *********
Company found: MAGAZINE LUIZA SA | CVM code: 22470
Year: 2023
File name: 22470-magazine-luiza-sa
File path: <path>\22470-magazine-luiza-sa
Download completed. File saved at:
<path>\...\2470-MAGAZINE LUIZA SA-2023-03-10.pdf
...
Document added successfully!
```

However, this work is not without challenges and limitations. The heterogeneity of data formats and the presence of incomplete or inconsistent information were obstacles faced during the process of capturing and processing the reports. Therefore, future work can focus on improving the robustness and accuracy of the pipeline to deal more efficiently with these issues.

Furthermore, a promising area for future studies is the application of natural language processing (NLP) techniques and machine learning (transformer-based models) for semantic analysis of financial reports. This could allow for a deeper understanding of the report's content, identifying sentiments, trends, and insights that go beyond purely quantitative analysis.

4. Impact

The proposed software has a significant impact on the field of financial data analysis. With the increasing relevance of accessing and analyzing unstructured data, particularly in the context of financial reports, the software provides an automated solution to obtain and process such data. Studies indicate that 71% of financial institutions use big data analysis to gain a competitive advantage (Peji´c et al. 2019).

Furthermore, the software significantly enhances the exploration of existing research questions. As mentioned by Zhang et al. (2018), the analysis of historical data has become more challenging, and the software addresses this difficulty by enabling the analysis of unstructured financial reports. These reports are crucial documents for analyzing companies' communications regarding their financial performance, health, and future prospects (Masson et al. 2020). They often contain valuable information that companies try to hide from investors but can provide clues about future performance.

The proposed software has a significant impact on financial data analysis. It provides an automated solution for obtaining and analyzing unstructured financial reports, improving the exploration of existing research questions, and transforming users' daily practices.

5. Conclusions

In this work, we have presented a pipeline proposal for the capture, processing, and provisioning of financial reports in the Brazilian capital market. By integrating advanced machine learning and data mining techniques, we aim to enhance analysis and informed decision-making in the financial market. Throughout this work, we collected unstructured data provided by the Securities and Exchange Commission (CVM) and developed a preprocessing and transformation process to adapt the financial reports to a suitable format for analysis.

Additionally, we organized and grouped the reports by financial institution and publication year, while indexing the metadata to catalog the documents and facilitate data access and understanding. However, this work is not without challenges and limitations. The heterogeneity of data formats and the presence of incomplete or inconsistent information posed obstacles during the report capture and processing process. Therefore, future work can focus on improving the robustness and accuracy of the pipeline to more efficiently address these issues.

Furthermore, a promising area for future studies is the application of natural language processing (NLP) techniques and machine learning (transformer-based models) for the semantic analysis of financial reports. This could enable a deeper understanding of the report content, identifying sentiments, trends, and insights that go beyond purely quantitative analysis. Another aspect to explore is the integration of external data, such as market information and financial news, to further enrich the analysis of financial reports. This could provide a broader context for investment decisions and contribute to a comprehensive view of the capital market.

References

[1] Apache Software Foundation, "Solr Reference Guide - Getting Started", Solr Apache, 2021. [Online]. Available: https://solr.apache.org/guide/solr/latest/getting-started/introduction.html. [Accessed: July 2, 2023].

[2] Beltagy, Iz, Peters, Matthew E., Cohan, Arman. (2020). Longformer: The Long-Document Transformer. arXiv preprint arXiv:2004.05150. Available at: https://arxiv.org/abs/2004.05150

[3] Borggreve, L.A. (2022) “Effects of Annual Report Sentiment on Stock Returns.“ UNIVERSITY OF TWENTE STUDENT THESES

[4] Checon, Bianca Q., Santana, Verônica de F., “Relatórios Obrigatórios e Padrôes de Divulgação no Mercado de Capitais.“ Artigo Cíclico n°8. FGV EASP Instituto de finanças

[5] CVM (2023). "O papel da CVM". Available at: https://www.gov.br/investidor/pt-br/investir/cuidados-ao-investir/o-papel-da-cvm

[6] Masson, Corentin and Patrick Paroubek. "NLP Analytics in Finance with DoRe: A French 250M Tokens Corpus of Corporate Annual Reports." International Conference on Language Resources and Evaluation (2020).

[7] Nakandala, Supun Withana, Sachith Kumarasiri, Dinu Jayawardena, Hirantha Bandara, Dilum Perera, Srinath Marru, Suresh Pamidighantam, Sudhakar. (2015). Schema-independent scientific data cataloging framework. MERCon 2015 - Moratuwa Engineering Research Conference. 289-294. 10.1109/MERCon.2015.7112361.

[8] Pedregosa F, Varoquaux G, GramfortA, MichelV, Thirion B, GriselO, etal.Scikit-learn: Machine learning in Python. J Mach Learn Res 2011;12:2825–30.

[9] Pejić Bach M., Ž. Krstić, S. Seljan, and L. Turulja, "Text Mining for Big Data Analysis in Financial Sector: A Literature Review", Sustainability, vol. 11, no. 5, p. 1277, Feb. 2019, doi: 10.3390/su11051277.

[10] Paszke A, Gross S, Massa F,Lerer A, Bradbury J, Chanan G, et al.PyTorch: An imperative style, high-performance deep learning library. In: WallachH, Larochelle H, Beygelzimer A, d'Alché Buc F,Fox E, Garnett R, editors. Advances in neural information processing systems 32. Curran Associates, Inc.;2019,p.8024–35, URL http://papers.neurips.cc/paper/9015-pytorchan-imperative-style-high-performance-deep-learning-library.pdf.

[11] Zhang, X., Tan, Y. (2018). "Deep Stock Ranker: A LSTM Neural Network Model for Stock Selection." In: Tan, Y., Shi, Y., Tang, Q. (eds) Data Mining and Big Data. DMBD 2018. Lecture Notes in Computer Science, vol. 10943. Springer, Cham.