

[Open in app](#)

Search

Write



P

How to use GROBID to extract text from PDF



Research Graph · [Follow](#)

8 min read · 15 hours ago



6



Created using DALL.E on 13 Feb 2024

Author: Aland Astudillo <https://orcid.org/0009-0008-8672-3168>

GROBID is a powerful and useful tool based on machine learning that can extract text information from PDF files and other files to a structured format.

One of the key challenges in knowledge mining from academic articles is reading the content of PDF files. The ability to efficiently and accurately read the content of PDF documents using automated tools and transform this data into a structured model provides many opportunities for the integration of PDF collections to expert systems, and AI-assisted data pipelines such as chatbots.

In this article, we will discuss GROBID capabilities, explain how to install it, and explore how to use it in a Python script or Jupyter Notebook.

What is GROBID?

GROBID stands for GeneRation of Bibliographic Data and is a machine learning library for extracting, parsing, and re-structuring raw documents such as PDF into structured XML/TEI encoded documents. It has a particular focus on technical and scientific publications. TEI stands for Text Encoding Initiative (<https://tei-c.org/Guidelines/>). It defines and documents a markup language for representing the structural, rendition, and conceptual features of texts.

How it works?

GROBID is a machine-learning library that extracts information from PDF files. The goal of GROBID is to help the text mining process, information extraction, and semantic analysis of scientific publications by transforming

them into machine-friendly, structured, representations with predictable structure.

Despite the large amount of PDF files and their extensive use among all fields, it is still a big challenge to process them because they are encoded in a vast variety of different publisher formats, often incomplete or inconsistent. In short, PDF files are difficult to exploit and use.

In a nutshell, GROBID uses a cascade of “sequence labelling models” to parse a document. Each model from the collection of models is in charge of identifying a small amount of labels to classify areas and information in the document. The final models produce 55 labels where each piece of identified information from the document will be structured. More information about how this library has been developed and the machine learning models are in this link: <https://grobid.readthedocs.io/en/latest/Principles/>

How to install GROBID

Here we explain the installation of the GROBID tool as a service in a Linux machine (Ubuntu for example). The same steps apply to a Windows system using the Windows Subsystem for Linux tool (WSL version 2).

For Linux, in the terminal with super user (sudo) rights create a folder and once inside, proceed to download the last version of GROBID using the wget tool. The current version of GROBID is v 0.8.0. After that, unzip the file:

```
> wget https://github.com/kermitt2/grobid/archive/0.8.0.zip  
> unzip 0.8.0.zip
```

For Windows, you can use Docker containers to install and run the GROBID service locally. More documentation is available at <https://grobid.readthedocs.io/en/latest/Run-Grobid/>

Note: At the time of writing this article, GROBID requires JAVA(JDK) and it works fine with JDK version 1.11 up to version JDK 1.17. Other recent JDK versions should work correctly. To check your JAVA installation, in the terminal use the following command:

```
> java -version
```

If you do not have any JAVA installed, proceed to install JAVA using the following line in the terminal.

```
> sudo apt install default-jdk
```

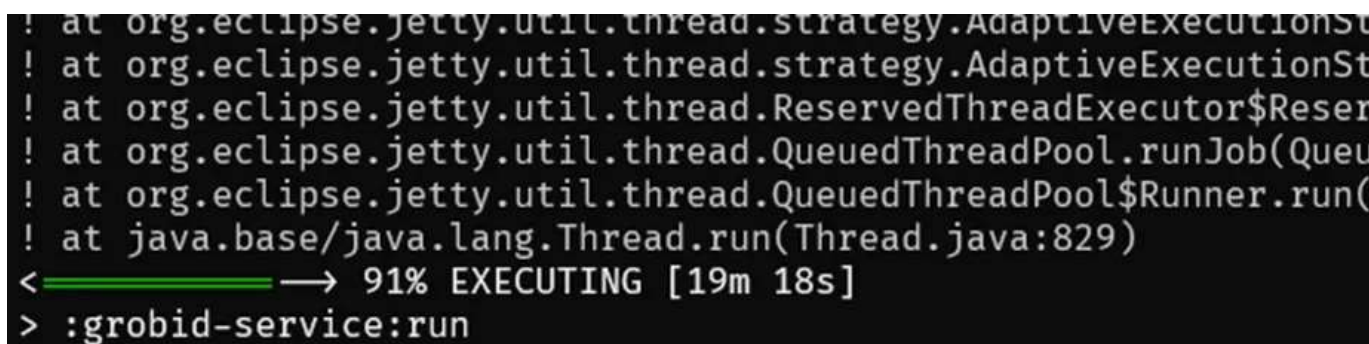
It is a good practice to check Java after installation to ensure the right version of Java is installed and ready to operate on your system.

How to run GROBID

Once you downloaded and unzipped the “*grobid*” file, proceed to the folder and run the *grewed* file. In the terminal ensure to use “.” before the command “*grewed*”.

```
> cd grobid-0.8.0  
  
> ./grewed run
```

The first time that you run the service, it will download and install all the required dependencies and then it will run the service. It will be completed by printing a message “:grobid-service:run” as presented in the following screenshot.



```
! at org.eclipse.jetty.util.thread.strategy.AdaptiveExecutionSt  
! at org.eclipse.jetty.util.thread.strategy.AdaptiveExecutionSt  
! at org.eclipse.jetty.util.thread.ReservedThreadExecutor$Reser  
! at org.eclipse.jetty.util.thread.QueuedThreadPool.runJob(Queu  
! at org.eclipse.jetty.util.thread.QueuedThreadPool$Runner.run(  
! at java.base/java.lang.Thread.run(Thread.java:829)  
<=====→ 91% EXECUTING [19m 18s]  
> :grobid-service:run
```

How to use GROBID

After running the GROBID service, you can connect to the created server in the following direction using your browser:

<http://localhost:8070/>

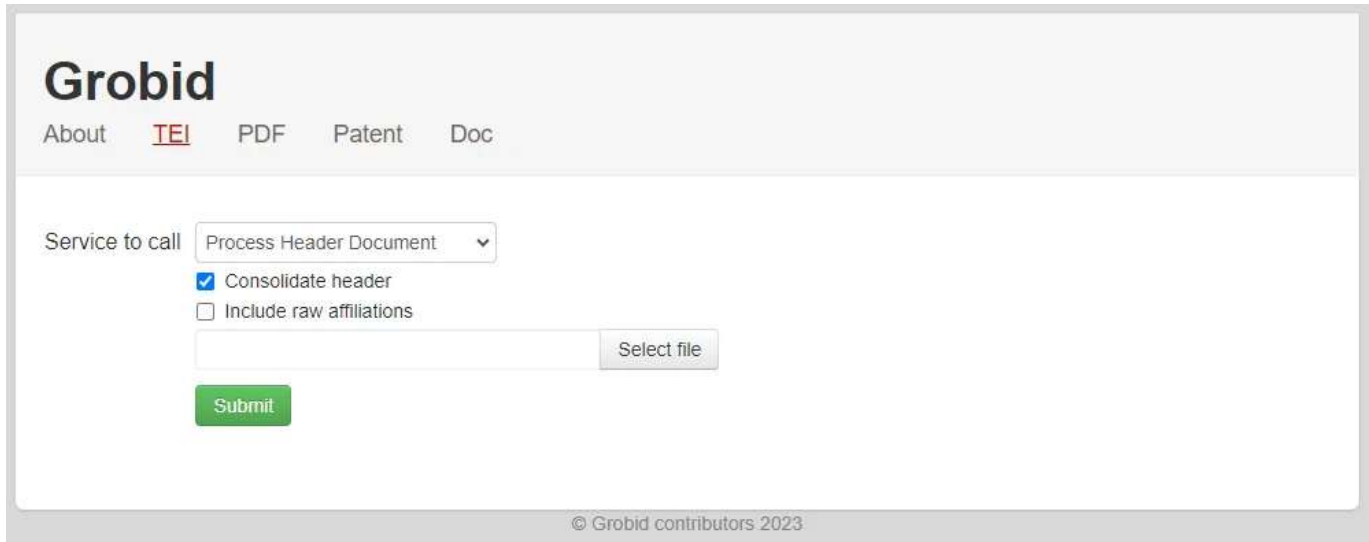
This follows the default configuration, but you can change the port in the config file. The image below shows how it looks the first time that you access the web layout.



The TEI menu will allow you to choose among the different services to call and to upload a file with the button “Select file”. The following functionalities are available as a service:

- Header extraction and parsing
- References extraction and parsing
- Citation contexts recognition and resolution
- Full-text extraction and structuring
- PDF coordinates
- Parsing of references
- Parsing of names
- Parsing of affiliation and address
- Parsing of dates
- Consolidation and resolution of the extracted bibliographical references
- Extraction and parsing of patent and non-patent references in patent publications
- Extraction of funders and funding information

Once you upload a PDF document and choose the required service, you can press the Submit button



The screenshot shows the Grobid web interface. At the top, there is a navigation bar with links: About, **TEI**, PDF, Patent, and Doc. Below this, the main content area has a section titled "Service to call" with a dropdown menu set to "Process Header Document". There are two checkboxes: "Consolidate header" (checked) and "Include raw affiliations" (unchecked). Below these is a text input field and a "Select file" button. A green "Submit" button is at the bottom left. The footer of the interface reads "© Grobid contributors 2023".

After a few seconds, the web will show you the XML structure of the output depending on what service you have selected. You can download the resulting XML structure in an XML file. The image below shows the result of processing the full-text document for a given paper. For example, in the image below we used the article *Pan et al., 2024 Unifying Large Language Models and Knowledge Graphs: A Roadmap. arXiv:2306.08302* downloaded from arXiv (<https://arxiv.org/abs/2306.08302>)

Grobid

[About](#)
[TEI](#)
[PDF](#)
[Patent](#)
[Doc](#)

Service to call Process Fulltext Document

☒ Consolidate header
 ☐ Consolidate citations
 ☐ Consolidate funders

☐ Include raw affiliations
 ☐ Include raw citations

☐ Segment sentences
 ☐ Add coordinates

Change
Remove

Submit
Download TEI Result

```

<?xml version="1.0" encoding="UTF-8"?>
<TEI
  xmlns="http://www.tei-c.org/ns/1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xlink="http://www.w3.org/1999/xlink" xml:space="preserve" xsi:schemaLocation="http://www.tei-c.org/ns/1.0 https://raw.githubusercontent.com/kermitt2/grobid/master/grobid-home/schemas/xsd/Grobid.xsd">
  <teiHeader xml:lang="en">
    <fileDesc>
      <titleStmt>
        <title level="a" type="main">Unifying Large Language Models and Knowledge Graphs: A Roadmap</title>
        <funder ref="#_xKYcEgt #_AGduva4">
          <orgName type="full">Australian Research Council</orgName>
          <orgName type="abbreviated">ARC</orgName>
        </funder>
        <funder ref="#_f5M9hr5">
          <orgName type="full">National Natural Science Foundation of China</orgName>
          <orgName type="abbreviated">NSFC</orgName>
        </funder>
      </titleStmt>
      <publicationStmt>
        <publisher/>
      </publicationStmt>
    </fileDesc>
  </teiHeader>
  <text>
    <p>This paper presents a roadmap for the future of large language models and knowledge graphs. It discusses the current state of the art and the challenges that need to be addressed in order to achieve the full potential of these technologies. The roadmap is divided into three main areas: research, development, and deployment. Each area contains a list of specific tasks and milestones that need to be completed in order to achieve the goals of the roadmap. The roadmap is intended to provide a clear and concise guide for researchers, developers, and deployers alike. It is hoped that this roadmap will help to accelerate the progress of research and development in this field and to ensure that the benefits of these technologies are realized for all.
  </text>
</TEI>

```

How to use GROBID Python Client

The GROBID Python client is a handy tool that can process efficiently and concurrently a set of PDF files in a given folder using GROBID as a service without requiring the web option. It includes a command line for processing PDFs and writing the results in a given folder. It allows us to use this tool inside Python scripts. To use this client, It assumes that you are running GROBID service, as we saw in the previous steps.

The repository of this client is here

https://github.com/kermitt2/grobid_client_python/blob/master/grobid_client/grobid_client.py

To install the GROBID Python client to use in your scripts or Jupyter Notebook, in your Python environment, you can use the pip manager in a given environment:

```
!pip install grobid_client_python
```

If you want to use this client in a script or a Jupyter Notebook, you must import the GROBID client class. Remember that it assumes that the GROBID service is already running.

```
from grobid_client.grobid_client import GrobidClient  
  
client = GrobidClient(config_path="./config.json")
```

Once the client connects to the service it will show you the following message.

```
"GROBID server is up and running"
```

How to analyse a bunch of PDF documents

After checking the connection, you can call the process method to analyse a bunch of documents in a folder. The process method has 3 required parameters and other optional parameters. In the following example we are using the service “**processFulltextDocument**” for processing all the pdf files in the folder “**resources/test_pdf**” and get the xml files output on the folder “**resources/test_out/**”. Additionally, we set the consolidate citation option and the TEI coordinates to “true”.

```
client.process("processFulltextDocument",  
              "./resources/test_pdf",  
              output="./resources/test_out/",  
              consolidate_citations=True,  
              tei_coordinates=True,  
              force=True)
```

In the defined folder, the output will be the XML files for each PDF file with the re-structured information.

How to analyse a single PDF file

For analysing one PDF file at a time, you can use the method **process_pdf**, with similar parameters as the **process** method:

```
service_name = "processFulltextDocument"  
  
pdf_file = "./pdf_examples/input/0046d83a-edd6-4631-b57c-755cdcce8b7f.pdf"  
  
rsp = client.process_pdf(service_name, pdf_file,  
                          generateIDs=True,  
                          consolidate_header=True,  
                          consolidate_citations=True,  
                          include_raw_citations=True,  
                          include_raw_affiliations=True,
```

```
tei_coordinates=True,
segment_sentences=True)
```

The following text snippet shows the top section of the “rsp” variable that holds the results from the PDF processing.

```
( './pdf_examples/input/0046d83a-edd6-4631-b57c-755cdcce8b7f.pdf',
  200,
  '<?xml version="1.0" encoding="UTF-8"?>
<TEI xml:space="preserve" xmlns="http://www.tei-c.org/ns/1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.tei-c.org/ns/1.0 https://raw.githubusercontent.com
xmlns:xlink="http://www.w3.org/1999/xlink">
  <teiHeader xml:lang="en">
    <fileDesc>
      <titleStmt>
        <title level="a" type="main" xml:id="_nrS3kCw">Multi-contact functional
        <funder ref="#_ZTFTHCu">
          <orgName type="full">German Federal Ministry for Education and Research
        </funder>
        <funder ref="#_xTkqs2w">
          <orgName type="full">German Research Council</orgName>
        </funder>
        <funder ref="#_dV4wQd4 #_dVJJnCd">
          <orgName type="full">Federal Ministry for Education and Research</orgN
        </funder>
      </titleStmt>
      <publicationStmt>
        <publisher>Springer Science and Business Media LLC</publisher>
        <availability status="unknown">
<p>Copyright Springer Science and Business Media LLC</p>
        </availability>
        <date type="published" when="2016-03-08">2016-03-08</date>
      </publicationStmt>
      <sourceDesc>
        <biblStruct>
          <analytic>
            <author>
              <persName coords="1,56.69,234.15,85.88,9.71"><forename type="first
              <surname>De Marchis</surname></persName>
              <email>cristiano.demarchis@uniroma3.it</email>
              <affiliation key="aff0">
                <note type="raw_affiliation"><label>1</label> Division of Functi
```

```

<orgName type="department" key="dep1">Division of Functional and
<orgName type="department" key="dep2">Department of Neurosurgery
<orgName type="institution">Eberhard Karls University</orgName>
<address>
  <addrLine>Otfried-Mueller-Str.45</addrLine>
  <postCode>72076</postCode>
  <settlement>Tübingen</settlement>
  <country key="DE">Germany</country>
</address>
</affiliation>
</author>

```

...

(Note: The result was truncated)

As you can see, the information is organised hierarchically, using standardised tags <tag> to identify each type of information of the publication. For instance, the tag <titleStmt> contains sub-tags holding the title of the article.

```

<title level="a" type="main" xml:id="_nrS3kCw">
Multi-contact functional electrical stimulation for hand opening: electrophysiol
</title>

```

Other tags contain information about authors, affiliations, and so on. For example, the identification of the organisation is in the tag <orgName>

```

<orgName type="full">
German Federal Ministry for Education and Research [BMBF

```

```
</orgName>
```

Information about the authors is inside the tags `<author>` containing details of each author, complete name (`<forename>`), last name (`<surname>`), email (`<email>`), affiliations (`<affiliation key="aff0">`), and all other information like the address of the institution to which the author is affiliated (`<address>`), and all the information contained in the article about the author.

```
<author>
  <persName coords="1,56.69,234.15,85.88,9.71">
    <forename type="first">Cristiano</forename>
    <surname>De Marchis</surname>
  </persName>
  <email>cristiano.demarchis@uniroma3.it</email>
  <affiliation key="aff0">
    <note type="raw_affiliation"><label>1</label> Division of Functional and Resto
    <orgName type="department" key="dep1">Division of Functional and Restorative
    <orgName type="department" key="dep2">Department of Neurosurgery</orgName>
    <orgName type="institution">Eberhard Karls University</orgName>
    <address>
      <addrLine>Otfried-Mueller-Str.45</addrLine>
      <postCode>72076</postCode>
      <settlement>Tübingen</settlement>
      <country key="DE">Germany</country>
    </address>
  </affiliation>
</author>
```

In the same way, we can retrieve information on the citations of the article in the tag for references. For example, an article in the list of references has the following structure

```

<biblStruct coords="8,319.75,636.26,214.89,6.62;8,319.75,645.27,210.17,6.62;8,31
  <analytic>
    <title level="a" type="main">The influence of functional electrical stimulat
    <author>
      <persName xmlns="http://www.tei-c.org/ns/1.0">
        <forename type="first">Fanny</forename>
        <surname>Quandt</surname>
      </persName>
    </author>
    <author>
      <persName xmlns="http://www.tei-c.org/ns/1.0">
        <forename type="first">Friedhelm</forename>
        <forename type="middle">C</forename>
        <surname>Hummel</surname>
      </persName>
    </author>
    <idno type="DOI">10.1186/2040-7378-6-9</idno>
  </analytic>
  <monogr>
    <title level="j">Experimental & Translational Stroke Medicine</title>
    <title level="j" type="abbrev">Exp & Trans Stroke Med</title>
    <idno type="ISSNe">2040-7378</idno>
    <imprint>
      <biblScope unit="volume">6</biblScope>
      <biblScope unit="issue">1</biblScope>
      <biblScope unit="page">9</biblScope>
      <date type="published" when="2014-08-21"/>
      <publisher>Springer Science and Business Media LLC</publisher>
    </imprint>
  </monogr>
</biblStruct>

```

In which we can identify the title tag (`<title level="a" type="main">`), authors information (`<author>`), DOI (`<idno type="DOI">`), and the journal details (`<monogr>`) of the referenced article.

Finally, to get each piece of information from this structure for further analysis, the next steps involves extracting the required information based on the tags of the XML structure, which can be done with different tools to

match these tags in the text. That said, the text is ready for mining all the information.

Conclusion

We have explored the important aspects of GROBID. With its simple installation process user interface, and scripts, we can put GROBID to work easier than ever. Whether you are facing mountains of PDFs or struggling to tame unruly bibliographic data, GROBID is your trusty companion on the path to research efficiency. Download GROBID today and unlock the door to a world of streamlined workflows, saving time, and deeper insights from your research endeavours.

Materials for further reading:

- <https://grobid.readthedocs.io/en/latest/>
- https://github.com/kermitt2/grobid_client_python


[Data Science](#)[Artificial Intelligence](#)[Pdf](#)[Machine Learning](#)[Knowledge](#)

Written by Research Graph

[Follow](#)

26 Followers


More from Research Graph

 Research Graph


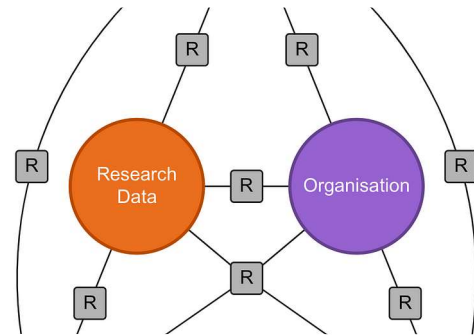
Unveiling the Synergy: Retrieval Augmented Generation (RAG)...

Tools and Platform for Integration of Knowledge Graph with RAG pipelines.

11 min read · Jan 18, 2024

 Research Graph


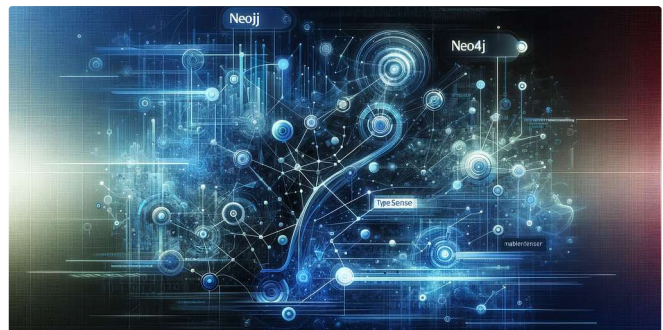
How to use GPT API to export a research graph from PDF...

 Research Graph

Mini RAG using Neo4j

Connecting OpenAI GPT to Neo4j

4 min read · Jan 25, 2024

 Research Graph

Typesense and Neo4j in a hybrid information retrieval solution

Using OpenAI API and the new Assistant functionality of GPT (Currently in Beta) to...

8 min read · Feb 6, 2024

👏 85 💬 1

🔖 ⋮

A novel approach to improving the efficiency of text search in graph databases utilizing...

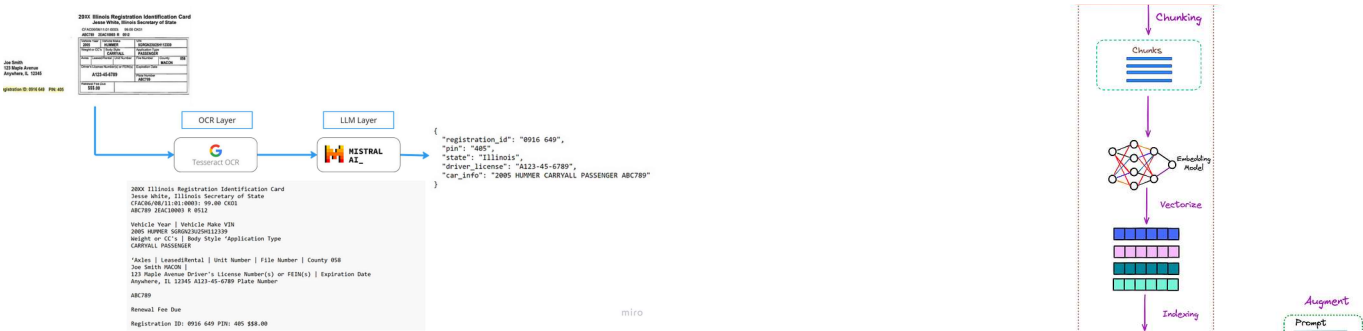
7 min read · Jan 18, 2024

👏 3 💬

🔖 ⋮

See all from Research Graph

Recommended from Medium



 Júlio Almeida in GoPenAI


Open-Source LLM Document Extraction Using Mistral 7B

Introduction

6 min read · Feb 2, 2024

👏 151 💬 1

🔖 ⋮

 Florian June in Towards AI

Advanced RAG 02: Unveiling PDF Parsing

Including key points, diagrams, and code

🌟 · 13 min read · Feb 2, 2024

👏 604 💬 8

🔖 ⋮

Lists



Predictive Modeling w/ Python

20 stories · 894 saves



Natural Language Processing

1189 stories · 665 saves



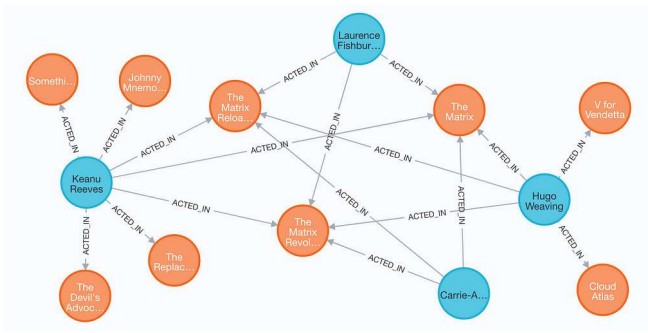
Practical Guides to Machine Learning

10 stories · 1046 saves



ChatGPT prompts

38 stories · 1115 saves



Mahimai Raja J

Build Knowledge Graph From TextData using LangChain | Unde...

Brain stores knowledge in the form of information graph.

5 min read · Feb 4, 2024



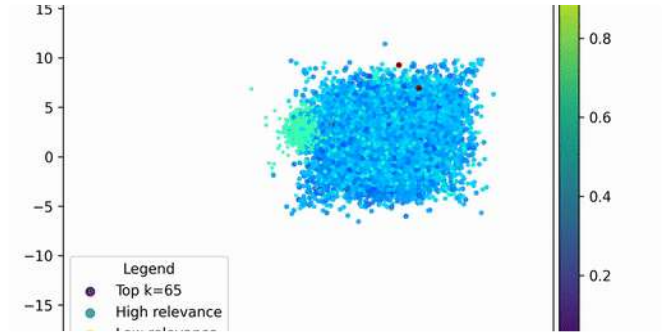
83



1



...



Markus Stoll in ITNEXT

Visualize your RAG Data—EDA for Retrieval-Augmented Generation

How to use UMAP dimensionality reduction for Embeddings to show Questions, Answer...

9 min read · 5 days ago



531



2



...



George Stavrakis in Towards Data Science

Extracting text from PDF files with Python: A comprehensive guide

A complete process to extract textual information from tables, images, and plain...

🌟 · 17 min read · Sep 22, 2023



1.2K



23



Senthil E in Level Up Coding

Unlocking LLM's Potential with RAG: A Complete Guide from...

Using OpenAI, Google Gemini Pro, and Open Source Models

47 min read · Feb 4, 2024



927



8



See more recommendations