Clinical Net

Technical Documentation of the Data Engineering and Modeling Pipelines

Table of Content

Nuria Coll Bastus
Clinical Net Director
nuria@clinicalnet.com

Mouafek Ayadi
Artificial Intelligence Consultant
mouafek.ayadi@esprit.tn

04-01-2022

Clinical Trials API

The following will detail the use of the Clinical Trials API to acquire the necessary data for our Data Engineering and Modeling phases. We have used these details to extract and automate the data collection.

1- Official links

ClinicalTrials.gov is a database of privately and publicly funded clinical studies conducted around the world.
https://clinicaltrials.gov/
The application programming interface (API) provides a toolbox for programmers and other technical users to use to access all posted information on ClinicalTrials.gov study records data.
https://clinicaltrials.gov/api/gui

2- API Definitions

Returns detailed definitions.The Study Structure, Study Fields List, Study Statistics, and Search Areas info URLs listed below retrieve key components of API Definitions.
https://clinicaltrials.gov/api/info/api_defs
<Struct Name="Study" NInstances="399046">
<Struct Name="OrgStudyIdInfo" NInstances="398997">
<Struct Name="StatusModule" NInstances="399046"
DEDLink="https://prsinfo.clinicaltrials.gov/definitions.html#status">
<Struct Name="EligibilityModule" NInstances="398227"
DEDLink="https://prsinfo.clinicaltrials.gov/definitions.html#Eligibility">
<List Name="LocationList" NInstances="354417" MinNChildInstances="1"
MaxNChildInstances="3511">

3- Study Structure

Returns all available data elements for a single study record.
https://clinicaltrials.gov/api/info/study_structure
<Struct Name="EligibilityModule"
DEDLink="https://prsinfo.clinicaltrials.gov/definitions.html#Eligibility">
<Struct Name="ContactsLocationsModule"
DEDLink="https://prsinfo.clinicaltrials.gov/definitions.html#Locations">

4- Study Fields List

Returns all data elements that can contain text values and are used by the Study Statistics and Search Areas info URLs listed below.
https://clinicaltrials.gov/api/info/study_fields_list

5- Search Areas

Returns groups of weighted study fields, or "search areas", that can be searched at once
https://clinicaltrials.gov/api/info/search_areas

6- Query URLs

Query URLs need the expr parameter to set a search expression.
If no search expression is provided in a Query URL, the query will retrieve all study records.

Note that queries can retrieve large numbers of study records.
Each query URL type has default and maximum numbers of studies that can be returned.

Use the minimum rank (min_rnk) and maximum rank (max_rnk) parameters to select the range of study records to process and return.

To retrieve more study records, issue multiple queries with sequential ranges for the minimum rank and maximum rank until all records have been retrieved.
Demo GUI
https://clinicaltrials.gov/api/gui/demo/simple_full_study

7- Full Studies (Lung Cancer example)

Break down of the query url fields

https://clinicaltrials.gov/api/query/full_studies?
expr=lung+cancer
&min_rnk=1
&max_rnk=2
&fmt=xml

https://clinicaltrials.gov/api/query/full_studies?expr=lung+cancer&min_rnk=1&max_rnk=2&fmt=xml

Break down of the query url response

<FullStudy Rank="1">
<Field Name="NCTId">NCT03581708

<Field Name="OrgFullName">Guangdong Provincial People's Hospital
<Field Name="OfficialTitle">Real-world Study of the Incidence and Risk Factors of Venous Thromboembolism
(VTE) in Chinese Advanced Stage Lung Cancer
<Field Name="OverallStatus">Not yet recruiting
<Field Name="Keyword">lung cancer
<Field Name="Keyword">Venous Thromboembolism
<Field Name="DetailedDescription">VTE has high incidence in lung cancer and increases the mortality. Appropriate preventive measures contribute to 50% increase of incidence. The investigators are to investigate the VTE in advanced

non-small cell lung cancer and delineate the risk factors to establish a VTE risk model system helping clinicians
to differentiate VTE high risk population and apply early prevention in order to reduce the incidence of VTE.

<Field Name="Condition">Lung Neoplasms(if there are ultiple condition the api request with format csv will seperate them with a pipe '|')

<Field Name="EligibilityCriteria">

Inclusion Criteria:

Age ≥ 18 years at the time of screening.

Eastern Cooperative Oncology Group performance status of ≤ 2.

Written informed consent obtained from the patient.

Histologically and cytologically documented Stage 3B-4 lung cancer
(according to Version 8 of the International Association for the Study of Lung Cancer Staging system). Patients with stage 1 to 3, who undergo radical therapy with disease free survival (DFS) >12 months. Willingness and ability to comply with scheduled visits and other study procedures.

Exclusion Criteria:

History of another primary malignancy except for malignancy treated with curative intent with known active disease ≥ 5 years before date of the informed consent.
Without signed informed consent.Unwillingness or inability to comply with scheduled visits or other study procedures. Previously diagnosed with VTE before signing informed consent.

<Field Name="HealthyVolunteers">No

<Field Name="Gender">All

<Field Name="MinimumAge">18 Years

<Field Name="StudyPopulation">Patients diagnosed with advanced staged lung cancer with written informed consent

<Field Name="LocationFacility">Guangdong General Hospital

<Field Name="LocationCity">Guangzhou

<Field Name="LocationState">Guagndong

<Field Name="LocationZip">510080

<Field Name="LocationCountry">China

8- Study Fields (we'll be working with this)

Returns values from selected API fields for a large set of study records. Select the fields returned using the fields parameter.
Returns 20 study records by default.
Returns in JSON or CSV format when the format parameter is set (fmt=JSON or fmt=CSV, respectively).Returns up to 1,000 study records when minimum rank and maximum rank parameters are set.

https://clinicaltrials.gov/api/query/study_fields?expr=heart+attack&fields=NCTId,Condition,BriefTitle&fmt=csv

The full list of the study fields to add to the above URL if needed
https://clinicaltrials.gov/api/info/study_fields_list

Before using Python to automate the data extraction we can test if the URL in question is returning the targeted data we can use the playground
https://clinicaltrials.gov/api/gui/demo/simple_study_fields

Data Engineering

After understanding how the API works we have automated the data collection using Python, and you can check the results here https://github.com/MWFK/NLP-Semantic-Similarity/tree/main/ClinicalTrials/Data

We have used Anaconda Jupyter Notebooks on our local laptop, which requires the use of the following software https://docs.anaconda.com/anaconda/ https://jupyter.org/

The notebooks that have produced the data in the link above are then uploaded here https://github.com/MWFK/NLP-Semantic-Similarity/tree/main/ClinicalTrials/Data%20Engineering

The Clinical Trials dataset was divided into four different parts since the maximum file size that can be uploaded to GitHub is 25MB(Megabyte) then get reassembled, and the purpose is to able to connect the data automatically to Google Colaboratory, or 'Colab' for short which allow to write and execute Python in your browser, with zero configuration required, free access to GPUs, Easy sharing, instead of or installing software, libraries, and dependencies on your machine, and benefit from faster compute https://colab.research.google.com/

For the test use cases, to be able to manipulate them with Python we have manually put them into an Excel sheet, then we have uploaded them to the same repository as the Clinical Trials Data, for the same reasons mentioned before.

We have linked the notebooks from Google Colab with the Data on GitHub, and the updated notebooks are saved into GitHub, where they can be accessed by just pressing the “Open in Colab”, once in Colab, you can choose using GPU(Graphics processing unit) which much faster than the regular Colab processing Unit or ordinary personal machines, then press Run All from the Runtime tab or Ctrl+F9 to run all the code and get the results, and these previous steps are not specific to the Data Engineering phase but for most notebooks.

The following are parts of the Data Engineering pipeline:

1. Separate the Inclusion Criteria from Exclusion Criteria
2. Detect the features Types* and Encoding*, then clean it
3. Remove Stop-Words*, Punctuation, and lowercase the text
4. Tokenize* the texts and apply Lemmatization* to the tokens
5. Filter by Age, Gender, and Travel Distance

6. Remove Accents and perform other character normalization
7. Use unigrams and bigrams for word N-Grams* boundaries
8. Limit the number of Vectorized* features to 50000
9. Apply TFIDF* on the dataset, filtered dataset, and the test set

Explication of the technical terms

1. Type: the data can have the same output visually, but it can have different types ('5' can be both an integer or a String which can break executions, so limiting the user input to choose from a specific list, that yield specific data types).

2. Encoding: the data can have the same output visually, but they're encoded in different ways, which can be interpreted by the code in different ways (â€¢ this is a badly decoded character).

3. Stop-Words: Stopwords are English words which does not add much meaning to a sentence. They can safely be ignored without sacrificing the meaning of the sentence. For example, the words like the, he, have etc.

4. Tokenization: is a common task in Natural Language Processing (NLP). It's a fundamental step in both traditional NLP methods like Count Vectorizer which we have used in the first model https://github.com/MWFK/NLP-Semantic-Similarity/blob/main/ClinicalTrials/Models/00.%20Model_0.ipynb
and Advanced Deep Learning-based architectures like Transformers like we have used in the second model https://github.com/MWFK/NLP-Semantic-Similarity/blob/main/ClinicalTrials/Models/01.%20Transformers_Cosine_GPU.ipynb

5. Lemmatization: Lemmatization is the process of converting a word to its base form. The difference between stemming and lemmatization is, lemmatization considers the context and converts the word to its meaningful base form, whereas stemming just removes the last few characters, often leading to incorrect meanings and spelling errors.

6. N-grams: are continuous sequences of words or symbols or tokens in a document. In technical terms, they can be defined as the neighboring sequences of items in a document. They come into play when we deal with text data in NLP(Natural Language Processing) tasks.

7. Vectorization: is a step in Machine Learning to perform feature extraction. The idea is to get some distinct features out of the text for the model to train on, by converting text to numerical vectors.
8. TF-IDF: Term Frequency–Inverse Document Frequency for a word in a document is calculated by multiplying two different metrics: The term frequency of a word in a document. ... The inverse document frequency of the word across a set of documents. This means, how common or rare a word is in the entire document set.

Modeling Pipeline

After Going through all the steps details that we have explained in the previous sections, we have created two Natural Language processing models. The first is completely based on statistics and the second one is based on Deep Learning. In Data Science and Artificial Intelligence projects, we usually start by trying to answer the project's challenges through classical frameworks to better understand how the data can be modeled, and we recure to AI when we have adequate data, and the analysis of that data demonstrate its modeling feasibility with Intelligent Learning methods, which was the case.

1- Statistical Model

After going through different combinations of the Data Engineering pipeline, our data was ready to compute the Semantic Similarity between the test use cases and the Lung Cancer dataset through the Cosine Similarity metric.

Semantic Similarity, or Semantic Textual Similarity, is a task in the area of Natural Language Processing (NLP) that scores the relationship between texts or documents using a defined metric. Semantic Similarity has various applications, such as information retrieval, text summarization, sentiment analysis, etc.

Cosine similarity is a metric used to determine how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. In this context, the two vectors I am talking about are arrays containing the word counts of two documents.

As a similarity metric, how does cosine similarity differ from the number of common words? When plotted on a multi-dimensional space, where each dimension corresponds to a word in the document, the cosine similarity captures the orientation (the angle) of the documents and not the magnitude. If you want the magnitude, compute the Euclidean distance instead. The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance because of the size (like, the word 'cricket' appeared 50 times in one document and 10 times in another) they could still have a smaller angle between them. Smaller the angle, the higher the similarity.

To compute the cosine similarity, you need the word count of the words in each document. The CountVectorizer or the TfidfVectorizer from scikit-learn* lets us compute this and process the data with NLTK*. The output of this comes as a sparse_matrix of the score of each use case compared to the rest of the Lung Cancer study Inclusion Criteria.

2- Deep Learning Model

Neural networks—and more specifically, artificial neural networks (ANNs)—mimic the human brain through a set of algorithms. At a basic level, a neural network is comprised of four main components: inputs, weights, a bias or threshold, and an output.

The “deep” in deep learning is referring to the depth of layers in a neural network. A neural network that consists of more than three layers—which would be inclusive of the inputs and the output—can be considered a deep learning algorithm.

There have been a lot of approaches for Semantic Similarity. The most straightforward and effective method now is to use a powerful model which is a BERT* Transformer* to encode sentences to get their Embeddings* and then use the Cosine Similarity Metric to compute the similarity score. The similarity score indicates whether two texts have similar or more different meanings.

Therefore, the second model was based on this new technology of Transformers, they’re are semi-supervised machine learning models that are primarily used with text data and have replaced recurrent neural networks in natural language processing tasks. Transformers were originally introduced by researchers at Google in the 2017 NIPS paper *Attention is All You Need*. Transformers are designed to work on sequence data and will take an input sequence and use it to generate an output sequence one element at a time.

To use Transformers we have installed on Google Colab the following libraries and frameworks: Hugging Face Transformers (formerly known as *pytorch-transformers* and *pytorch-pretrained-bert*) which provides thousands of pre-trained models and to prepare the data embedding we have used SentenceTransformers which is a Python framework for state-of-the-art text embedding, then we have fed the embeddings to the model pre-trained model ‘stsb-roberta-large’ which stands for ‘Sentence-Transformers Robustly optimized BERT Large dictionary.

3- The libraries, frameworks, and Pre-Trained models we have used

1. Scikit-learn: or Sklearn is the most **u**seful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering, and dimensionality reduction via a consistent interface in Python. https://scikit-learn.org/

2. NLTK: is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.https://www.nltk.org/

3. BERT: stands for Bidirectional Encoder Representations from Transformers. Unlike recent language representation models, BERT is designed to pre-train deep bidirectional representations from the unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications. https://arxiv.org/abs/1810.04805

4. Sentence-Transformers: SentenceTransformers is a Python framework for state-of-the-art sentence, text, and image embeddings. Sentence-BERT (SBERT), a modification of the pre-trained BERT network that uses siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine-similarity. https://www.sbert.net/

5. Hugging Face: the transformers package is an immensely popular Python library providing pre-trained models that are extraordinarily useful for a variety of natural language processing (NLP) tasks. https://huggingface.co/

6. PyTorch: An open-source machine learning framework that accelerates the path from research prototyping to production deployment. https://pytorch.org/

Suggestions and Questions

My suggestions and questions are based on the use cases that you have provided.

1. The test data should be in excel format (after the user submit the form online we'll receive it in a specific form, where all columns are predetermined). https://github.com/MWFK/NLP-Semantic-Similarity/blob/main/ClinicalTrials/Data/TestData.csv
2. Age and Gender should be separated.
3. Performance status: which field in the API has this value.
4. Willing to travel anywhere in the UK for treatment: This should be a list of choices. If they want to travel or not, and what is the distance they're able to travel?
5. Are there any other fields in the API that can hold specific pieces of information to the points highlighted in yellow?
6. Is happy to sign consent and enter in a clinical trial: Is there a chance where a user has submitted their form but they're not willing to enter a clinical trial? And there should be no adjectives in the form, only specified classes.
7. Phrases with several negations should be separated.
8. Phrases with different pieces of information should be separated.
9. All fields of the form should have a specific list.
10. Long detailed forms have better results, the more the questions are specific the more accurate the recommendations are.
11. For female patients are the specific fields that have the following pieces of information: pregnancy, breastfeeding?
12. Are there fields specific to mental illnesses?
13. For the location, the form should display a google maps option, so you can retrieve their exact location, and compute the distance where their clinical trial is located at.
14. Most fields should be mandatory to narrow down the search.
15. The input data in the form should have a specific format, the closest one to the format of the majority of the clinical trial data.

16. Ideally, the test data should be accompanied by the clinical trials that best fit them in order to have a primary source of verification.
17. We need to have the condition of the disease in a separate column.
18. The fields in the website form screen capture should be in separate columns in the form.
19. Are there any fields in the clinical trials that describe the following: The sub-type of the condition, The stage of the condition, How it was diagnosed, previous treatment, What was the treatment?
20. 19- Adding the following features to the form: HealthyVolunteers, StudyPopulation.
21. Platelet count < 50,000 units: Numerics should have a standard to write them in this case it's favorable to write it in this format 50000 units, otherwise, they can be decoded in the wrong way.
22. Data encoding: the characters we're seeing look exactly the same to us, except underneath the visual representation they're encoded in different ways, which can affect many aspects of the data processing and modeling, therefore the standard should be UTF-8, and the tabular file should be exported as Comma Separated Files UTF-8 instead of just CSV or XLSX.
23. Location names should not use abbreviations, the UK should be written the United Kingdom, otherwise during data processing and modeling it will yield less accurate results.
24. Adding the Healthy Volunteer feature will narrow down the search.
25. Comparing sentences to paragraphs and vice-versa will not give a perfect score, therefore the input data should match the Clinical Trials Inclusion Criteria in size to have a fair comparison.

P.S. The suggestions detailed above canl be implemented gradually and they're subject to continuous updates.

Conclusion

After Carefully designing an automated Data Engineering pipeline and modeling the data through statistical tools and Deep Learning models in addition to consulting research papers, the results definitely proved that the challenge is feasible, and the current outputs that we had can be improved gradually.