SEMANTIC SEARCH ON CODEBASES

Presenters :	Roll No.:
Maj Ashish Ahluwalia	21111073
Binay Kumar Suna	21111021
Chabil Kansal	21111022
Shubham Sinha	21111409

Supervised By : Prof. Arnab Bhattacharya

BRIEF OVERVIEW:

- Finding our desired snippet of code from large repository in a jiffy. This would be unlike the tedious traditional experience of finding code sections through word search using Ctrl+F.

- We could give a short description of what we intend to find and let the machine using its understanding of code find the code snippet matching for the given description.

Approach 1

S.No	TOPICS
1.	Data Implementation
2.	Translating Function to its English Description (Docstring) using Transformer
3.	Searching Via Semantic Similarity
4.	Building the Search engine Web Application

Data Implementation

Data Source

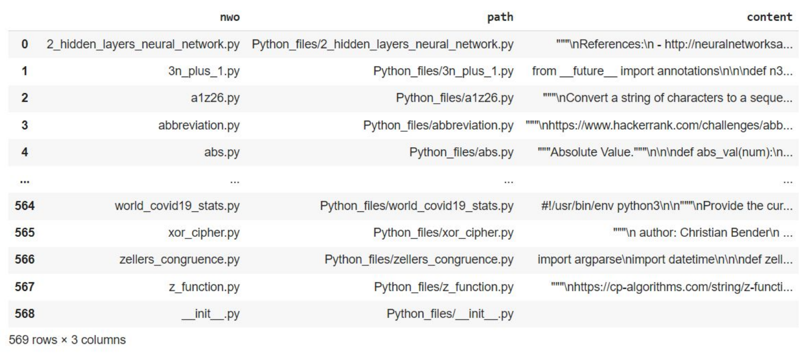
These dataset contain hundred of files containing python codes. Each python file contains various functions/classes and their descriptions in the form of comments.

Data Collection

For analysing the data and better pre-processing it, the entire data is loaded into a csv format containing various columns .(total - 1,50,000+ files)

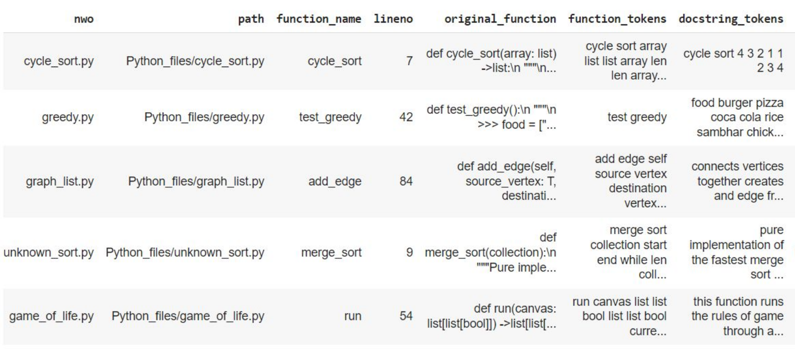
Data Preprocessing

After we extracted the function definition and its docstring we tokenized each of them to remove punctuation, decorators and convert all the tokens to lower case. Once we have extracted our function-docstring pairs and their tokens which are free from decorators and other unwanted elements,we stack our findings in a data-frame with every row containing details about a function and its corresponding docstring.



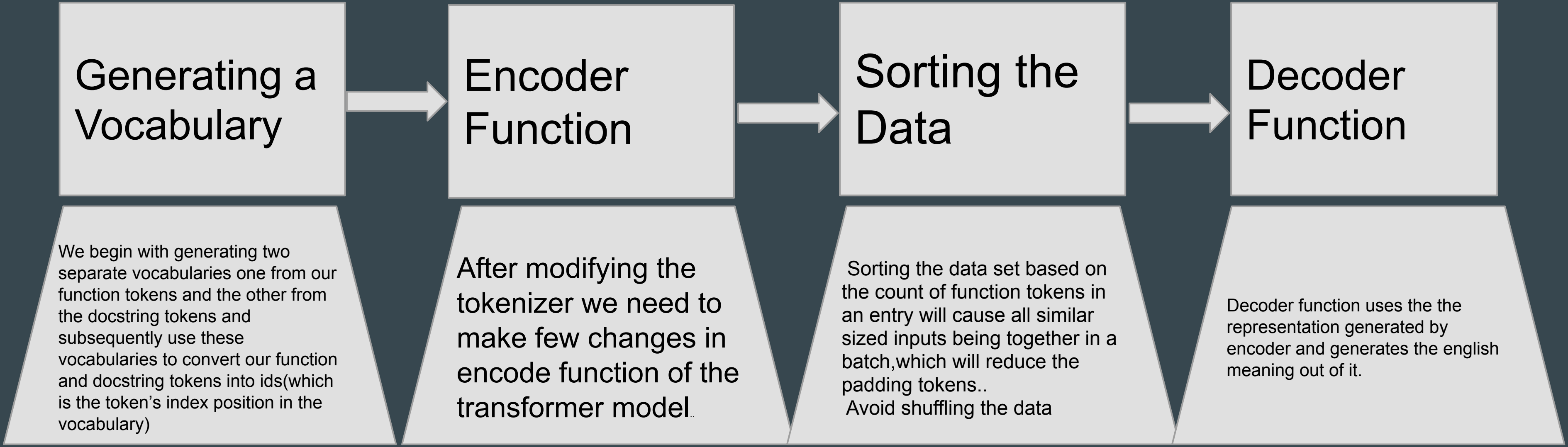
	nwo	path	content
0	2_hidden_layers_neural_network.py	Python_files/2_hidden_layers_neural_network.py	"""\nReferences:\n - http://neuralnetworksa...
1	3n_plus_1.py	Python_files/3n_plus_1.py	from __future__ import annotations\n\n\ndef n3...
2	a1z26.py	Python_files/a1z26.py	"""\nConvert a string of characters to a seque...
3	abbreviation.py	Python_files/abbreviation.py	"""\nhttps://www.hackerrank.com/challenges/abb...
4	abs.py	Python_files/abs.py	"""Absolute Value."""\n\n\ndef abs_val(num):\n...
...	...	...	...
564	world_covid19_stats.py	Python_files/world_covid19_stats.py	#!/usr/bin/env python3\n\n"""\nProvide the cur...
565	xor_cipher.py	Python_files/xor_cipher.py	"""\n author: Christian Bender\n ...
566	zellers_congruence.py	Python_files/zellers_congruence.py	import argparse\nimport datetime\n\n\ndef zell...
567	z_function.py	Python_files/z_function.py	"""\nhttps://cp-algorithms.com/string/z-functi...
568	__init__.py	Python_files/__init__.py	

569 rows × 3 columns



nwo	path	function_name	lineno	original_function	function_tokens	docstring_tokens
cycle_sort.py	Python_files/cycle_sort.py	cycle_sort	7	def cycle_sort(array: list) ->list:\n """\n...	cycle sort array list list array len len array...	cycle sort 4 3 2 1 1 2 3 4
greedy.py	Python_files/greedy.py	test_greedy	42	def test_greedy():\n """\n >>> food = ["...	test greedy	food burger pizza coca cola rice sambhar chick...
graph_list.py	Python_files/graph_list.py	add_edge	84	def add_edge(self, source_vertex: T, destinati...	add edge self source vertex destination vertex...	connects vertices together creates and edge fr...
unknown_sort.py	Python_files/unknown_sort.py	merge_sort	9	def merge_sort(collection):\n """Pure imple...	merge sort collection start end while len coll...	pure implementation of the fastest merge sort ...
game_of_life.py	Python_files/game_of_life.py	run	54	def run(canvas: list[list[bool]]) ->list[list[...	run canvas list list bool list list bool curre...	this function runs the rules of game through a...

Translating Function to its English Description (Docstring) using Transformer



Generating a Vocabulary

We begin with generating two separate vocabularies one from our function tokens and the other from the docstring tokens and subsequently use these vocabularies to convert our function and docstring tokens into ids(which is the token's index position in the vocabulary)

Encoder Function

After modifying the tokenizer we need to make few changes in encode function of the transformer model.

Sorting the Data

Sorting the data set based on the count of function tokens in an entry will cause all similar sized inputs being together in a batch,which will reduce the padding tokens..

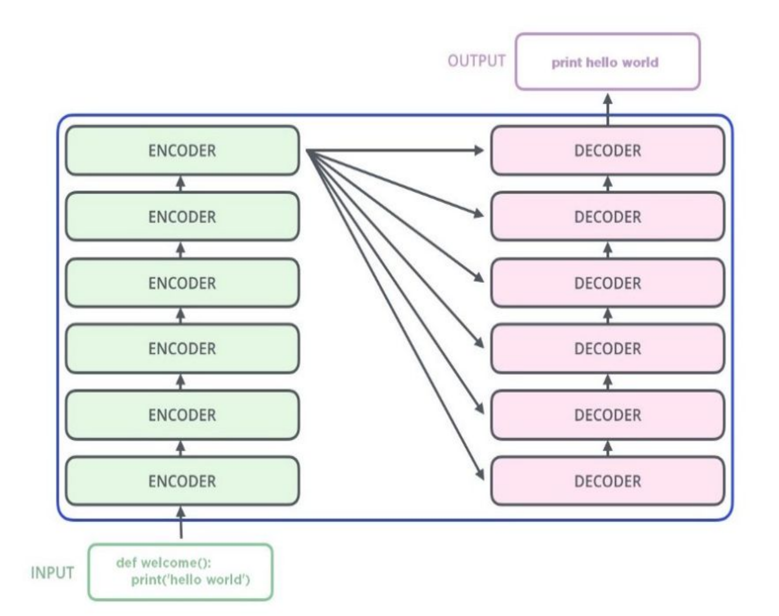
Avoid shuffling the data

Decoder Function

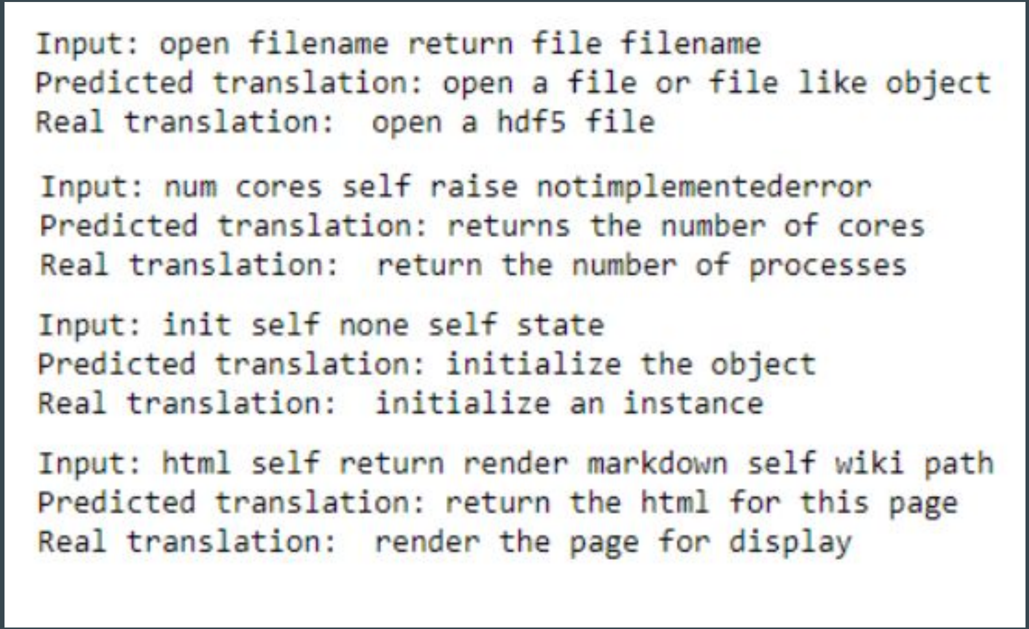
Decoder function uses the the representation generated by encoder and generates the english meaning out of it.

AFTER TRAINING THE MODEL WE GET A FUNCTION IN A TEXT FORM

Translating Function to its English Description (Docstring) using Transformer



Transformer Translation Function in English

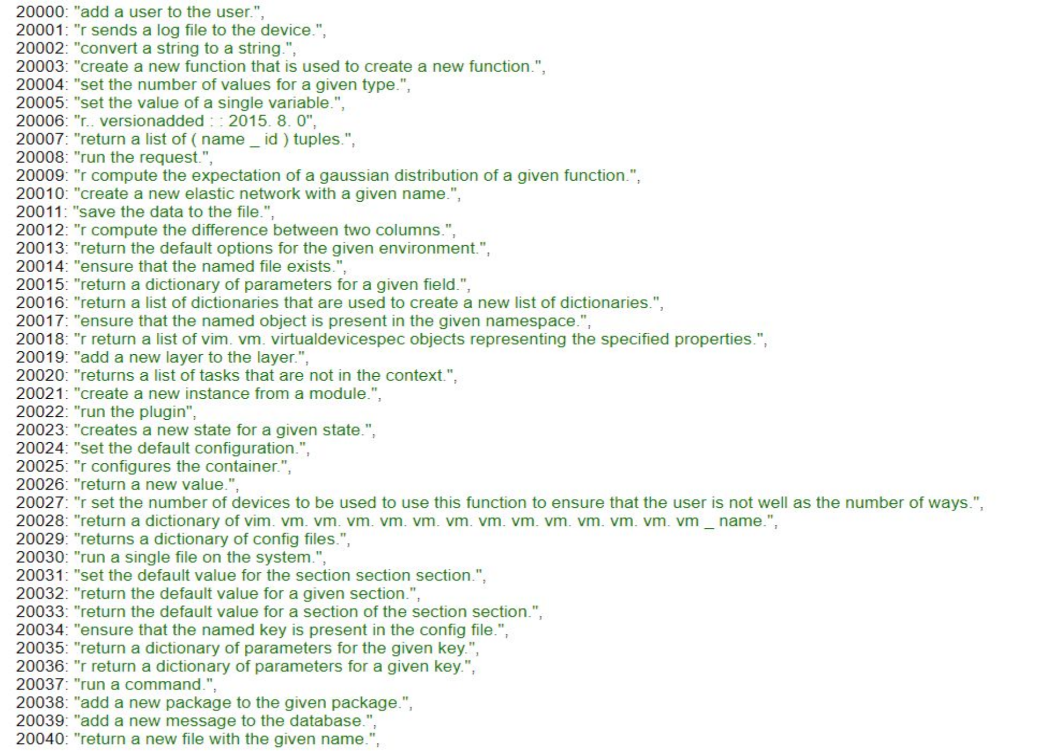


```
Input: open filename return file filename
Predicted translation: open a file or file like object
Real translation:  open a hdf5 file

Input: num cores self raise notimplementederror
Predicted translation: returns the number of cores
Real translation:  return the number of processes

Input: init self none self state
Predicted translation: initialize the object
Real translation:  initialize an instance

Input: html self return render markdown self wiki path
Predicted translation: return the html for this page
Real translation:  render the page for display
```

Translation Made by the Model

Translating Function to its English Description (Docstring) using Transformer

Translation File



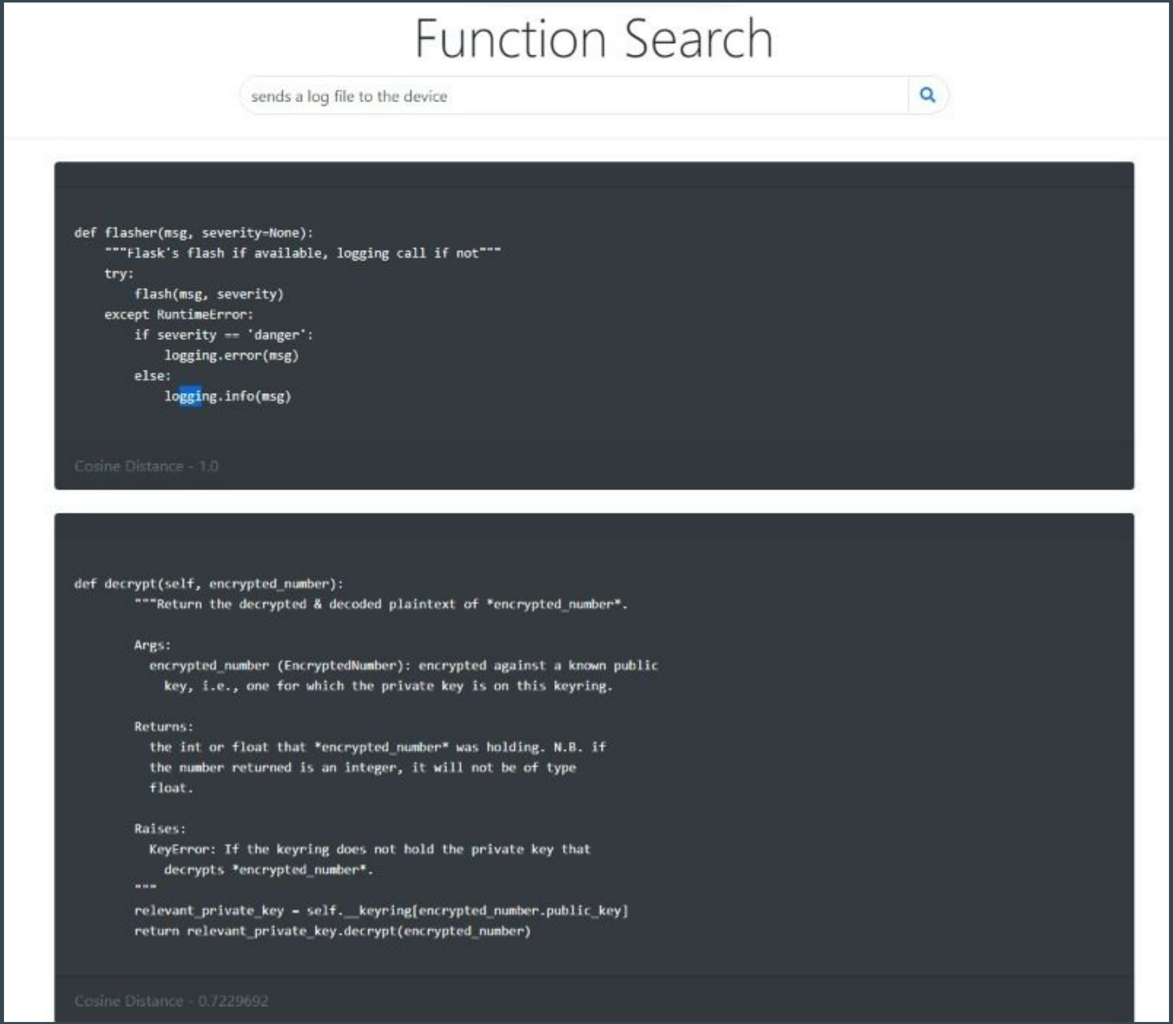
```
20000: "add a user to the user.",
20001: "r sends a log file to the device.",
20002: "convert a string to a string.",
20003: "create a new function that is used to create a new function.",
20004: "set the number of values for a given type.",
20005: "set the value of a single variable.",
20006: "r.. versionadded : : 2015. 8. 0",
20007: "return a list of ( name _ id ) tuples.",
20008: "run the request.",
20009: "r compute the expectation of a gaussian distribution of a given function.",
20010: "create a new elastic network with a given name.",
20011: "save the data to the file.",
20012: "r compute the difference between two columns.",
20013: "return the default options for the given environment.",
20014: "ensure that the named file exists.",
20015: "return a dictionary of parameters for a given field.",
20016: "return a list of dictionaries that are used to create a new list of dictionaries.",
20017: "ensure that the named object is present in the given namespace.",
20018: "r return a list of vim. vm. virtualdevicespec objects representing the specified properties.",
20019: "add a new layer to the layer.",
20020: "returns a list of tasks that are not in the context.",
20021: "create a new instance from a module.",
20022: "run the plugin",
20023: "creates a new state for a given state.",
20024: "set the default configuration.",
20025: "r configures the container.",
20026: "return a new value.",
20027: "r set the number of devices to be used to use this function to ensure that the user is not well as the number of ways.",
20028: "return a dictionary of vim. vm. vm. vm. vm. vm. vm. vm. vm. vm. vm. vm. vm. vm _ name.",
20029: "returns a dictionary of config files.",
20030: "run a single file on the system.",
20031: "set the default value for the section section section.",
20032: "return the default value for a given section.",
20033: "return the default value for a section of the section section.",
20034: "ensure that the named key is present in the config file.",
20035: "return a dictionary of parameters for the given key.",
20036: "r return a dictionary of parameters for a given key.",
20037: "run a command.",
20038: "add a new package to the given package.",
20039: "add a new message to the database.",
20040: "return a new file with the given name.",
```

Searching Via Semantic Similarity

- Semantic similarity scores words based on how similar they are, even if they are not exact matches. It borrows techniques from Natural Language Processing (NLP).
- We have used the word embedding model GloVe which maps words into numerical vectors which are points in a multi-dimensional space so that words that occur together often are near each other in space.
- We create a similarity matrix, that contains the similarity between each pair of words, weighted using the term frequency then calculate the soft cosine similarity (as regular cosine similarity return zero for vectors with no overlapping terms),which considers the word similarity between the query and each of the documents.

Building the Search engine Web Application

- We have used Flask which is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier.
- After receiving the input query the browser embed it as a GET request and convert it to vector for our model. The search function will return us with top five results having the highest cosine similarity.



```
def flasher(msg, severity=None):
    """Flask's flash if available, logging call if not"""
    try:
        flash(msg, severity)
    except RuntimeError:
        if severity == 'danger':
            logging.error(msg)
        else:
            logging.info(msg)
```

Cosine Distance - 1.0

```
def decrypt(self, encrypted_number):
        """Return the decrypted & decoded plaintext of *encrypted_number*.

        Args:
          encrypted_number (EncryptedNumber): encrypted against a known public
            key, i.e., one for which the private key is on this keyring.

        Returns:
          the int or float that *encrypted_number* was holding. N.B. if
          the number returned is an integer, it will not be of type
          float.

        Raises:
          KeyError: If the keyring does not hold the private key that
            decrypts *encrypted_number*.
        """
        relevant_private_key = self.__keyring[encrypted_number.public_key]
        return relevant_private_key.decrypt(encrypted_number)
```

Cosine Distance - 0.7229692

Approach-2

S.No	TOPICS
1.	Data Implementation (Same as Approach 1)
2.	Converting Docstring to Vector
3.	Converting Functions to Vectors
4.	Building the Search Logic

Converting Docstring to Vector

- The docstrings are converted to vectors using a pretrained ALBERT model which is fine-tuned on our data set. ALBERT is chosen because its faster to train, low on memory consumption and trains on harder tasks as compared to BERT.

- Fine-tune the word embedding weights and the weights of the encoders to make it understand the words and infer its meaning from a computer programming context.

- Learn representations for even programming jargons, like ‘SQL, csv ’ etc., which might not be present in its own vocabulary.

Converting Functions to Vectors

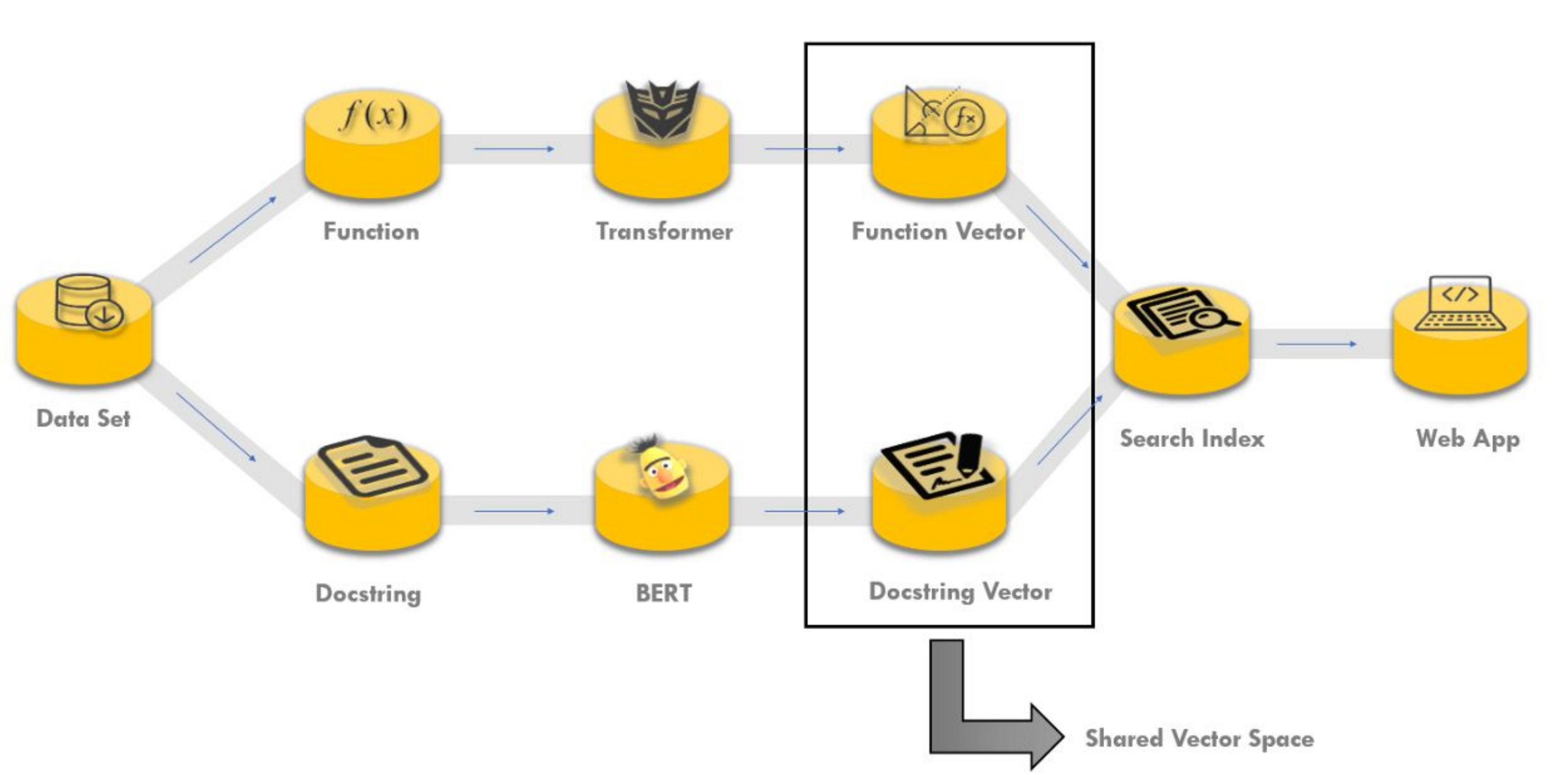
- To convert the functions into 768-dimensional vectors such that the function vector and the docstring vector are in a shared vector space.

- State-of-the-art results in the field of machine translation.

- Remove the decoder from the transformer architecture and use the trained encoders of the transformer to give an encoded representation of the function.

- Send the input from the encoder layers of the transformer to an LSTM layer which passes its output to a dense layer to finally output a 768-dimensional vector.

Building the Search Logic



- Used Non-Metric Space Library (nmslib)

- Encode the search query to a vector using our trained ALBERT model.

- Function vectors similar to the search query vector are searched and we are returned index values and the distances of five nearest neighbors to the search query.

- Extract its details using the index value which corresponds to its index value in the data set and display the results

Thank you!