PAN SOFTWARE USE CASE DEMO

USE CASE – PROVIDE DETAILS BASED ON THE QUERY :

The use case is to build a model that could break down the text keyed in by a User to meaningful JSON items as shown in the Pan software pdf.

TASKS TO BE DONE :

To show the output in a presentable form. More varieties can be added that a user can key in and any standard tool can be used to show the output.

This solution should be a smart solution in which it can be expanded and reach to greater levels

STEPS TAKEN :

There can be various solutions that can be brought to the table but I am thinking from the perspective of the jobs which is related to NLP (Data Science).

To convert the string into json output we have to import json package in python. And after that, we can use json.loads() method which accepts a valid json string and returns the output in the form of a dictionary. Below is the output in the jupyter notebook.

```
import json
```

```
json_data = '[{"ID":10,"Name":"Pankaj","Role":"CEO"},' \
            '{"ID":20,"Name":"David Lee","Role":"Editor"}]'
```

```
#using json.loads for creating json object from json string
json_object = json.loads(json_data)
```

```
# using json.dumps() method takes the json object and returns json formatted strings
json_formatted_str = json.dumps(json_object, indent=2)
```

```
print(json_formatted_str)
```

```
[
  {
    "ID": 10,
    "Name": "Pankaj",
    "Role": "CEO"
  },
  {
    "ID": 20,
    "Name": "David Lee",
    "Role": "Editor"
  }
]
```

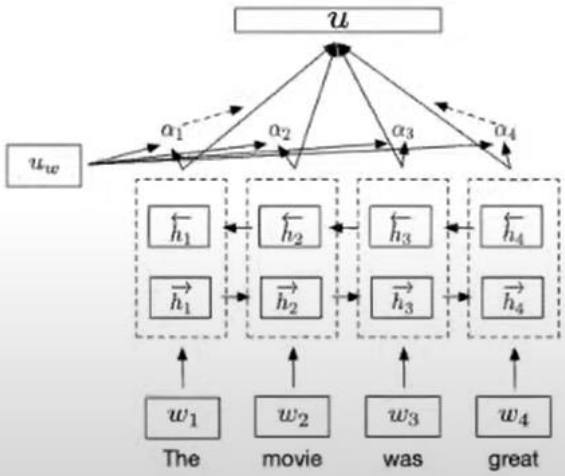
The above example shown is just a simple version of converting string to json in a tabular form but the main question is how can we get a more detailed version of the query asked. The answer to this question depends on the type of answer we want to generate from the query.
There can be multiple solutions to get an informative solution to the questions asked :

Through my research, I find out that semantic search is not the best way to reach the solution because semantic search is not generalized as it is expected it.
So, there's Search query expansion which uses Machine learning to predict the relationship between the queries to get the best result. It is called Implicit Relation Extraction as shown in the below image.

Implicit Relation Extraction

n to understand which parts of the input were more important to
ertain task.

is part of the transformer
nowing the relative



The above picture shows the graph of a transformer in which certain attention weights will give certain importance to each input inside the sentence to find the output.

There's a layer in the transformer which is called the self-attention layer which will help in understanding if there's an implicit relationship between the queries.

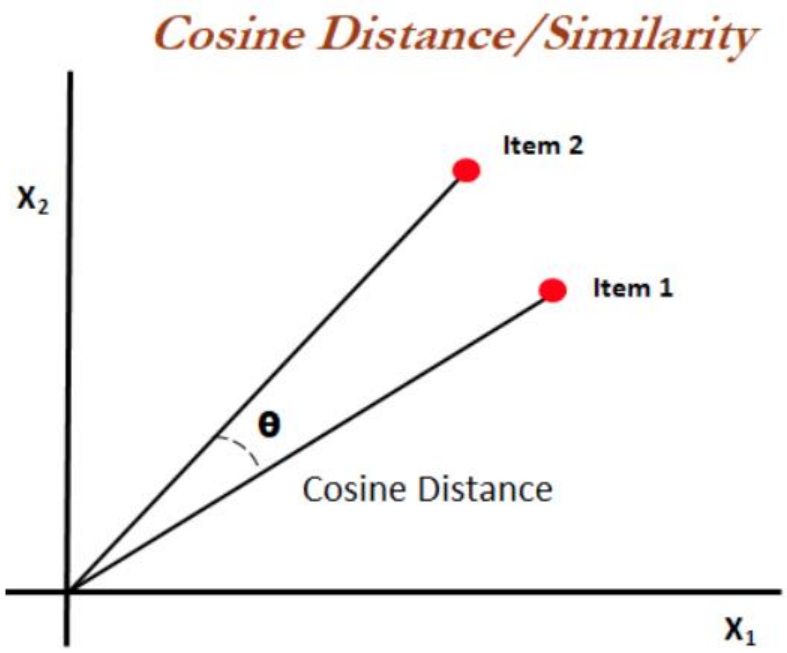
Alternatively, there are other methods to find the solution. One of them could be a Sentiment search in which we can use sentence embeddings using Sentence Transformer (a pre-trained model).

Result :

For shorter queries, Supervised learning should be done and for longer queries or queries which require various outputs, unsupervised learning techniques should be used.

```
Query: A man is eating pasta.

Top 5 most similar sentences in corpus:
A man is eating food. (Score: 0.7035)
A man is eating a piece of bread. (Score: 0.5272)
A man is riding a horse. (Score: 0.1889)
A man is riding a white horse on an enclosed ground. (Score: 0.1047)
A cheetah is running behind its prey. (Score: 0.0980)


======================


Query: Someone in a gorilla costume is playing a set of drums.

Top 5 most similar sentences in corpus:
A monkey is playing drums. (Score: 0.6433)
A woman is playing violin. (Score: 0.2564)
A man is riding a horse. (Score: 0.1389)
A man is riding a white horse on an enclosed ground. (Score: 0.1191)
A cheetah is running behind its prey. (Score: 0.1080)
```

The above image shows the answers to the queries based on the score each sentence scored opposite to the queries keyed in.



How the similarity is checked – it is based on the angle between the two items which is Q as shown in the above image. The shorter the distance the greater the similarity between them.

These query outputs can be then transformed into json outputs and the queries which have the highest score can be placed opposite to the queries keyed-in in a tabular format as shown in the Pan software use case.