HINDUSTAN AERONAUTICS LIMITED
AIRCRAFT DIVISION
NASHIK

TRAINING PROJECT REPORT

"QUESTION ANSWERING MODEL BASED ON AI AND NATURAL LANGUAGE PROCESSING"

Mr. SIDDHIVINAYAK SAHOO

B.E (CSE) 3RD YEAR

SVKM NMIMS

MPSTME ,SHIRPUR

PROJECT GUIDE:

Mr. M. RAJENDER

DGM (DESIGN)

AIRCRAFT UPGRADE RESEARCH AND DESIGN CENTRE

CERTIFICATE

This is to certify that Mr. SIDDHIVINAYK SAHOO, SAP ID-70021119051, of B.E. (CSE) 3rd Year from SVKM NMIMS MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMNT AND ENGINEERING SHIRPUR, has successfully completed training and project work during the period 4th MAY to 4th JUNE 2021 on ***"QUESTION ANSWERING MODEL BASED ON ARTIFICIAL INTELLIGENCE AND NATURAL LANGUAGE PROCESSING"*** at Design Quality department at Aircraft Upgrade Research and Design Centre (AURDC), Hindustan Aeronautics Limited (HAL), Nashik, as a part of Industrial Internship.

PROJECT GUIDE	HEAD OF DEPARTMENT
Mr. M. RAJENDER	Mr. S.J.BHOLE
DY. GENERAL MANAGER (DESIGN)	ADDL. GENERAL MANAGER (DESIGN)

ACKNOWLEDGEMENT

This is the report of the project I undertook during Internship for duration of 4 weeks at Hindustan Aeronautics Limited, Nashik.

I would like to express my gratitude to my project guide **Mr. M. RAJENDER,DY. GENERAL MANAGER (DESIGN)** who has guided me through each phase of my project. His guidance, suggestions and invaluably problem solving and friendly advice during our discussions, are much focused to give me a proper direction to analyze and understand the project objective.

I am grateful to **TRAINING DEPARTMENT** for giving me the opportunity to undertake the project work at "M/s Hindustan Aeronautics Limited, Aircraft Division, Nashik, India".

I would also like to extend my sincere thanks to all the staff members and employees.

ABSTRACT

The Aircraft Manufacturing and Overhaul is a complex and technical intensive job. At various stages of aircraft manufacturing and overhaul faults do occur due to various reasons. To resolve these faults technician have to go through a lot of manuals and paper work .

Since this is monotonous and time consuming sometimes takes very long time to go through all the documentation and find out the desired solution. It is proposed to implement Machine Learning/Artificial Intelligence/Natural Language Processing model to automate this long procedure .Due to this project technician can upload the paperwork and manuals on their local machine and directly search for answers.

This project is an implementation of Natural Language Processing (NLP) field of Artificial Intelligence. This project aims to develop, train, and train a question answering model . This is a simple implementation of search engine

TABLE OF CONTENTS

1.ACKNOWLEDGEMENT
2.ABSTRACT
3.ABOUT THE COMAPANY
4.INTRODUCTION
5.SYSTEM CONFIGURATION
6.PROJECT
    1.PROBLEM STATEMENT
    2.MODEL 1
    - UNDERSTANDING THE DATA
    - EXPLORING THE ENTITIES
    - PREPROCESSING
    - NATURAL LANGUAGE PROCESSING
    - OUTPUT
    3. MODEL 2 :YES OR NO ANSWERING ON LIMITED VOCABULARY USING NEURAL NETWORKS
    - UNDERSTANDING THE DATA
    - TOKENINZING THE DATA
    - VECTORIZING THE DATA
    - CREATING THE MODEL
    - MODEL SUMMARY
    - MODEL ACCURACY PLOT
    - TESTING THE MODEL
    4. MODEL 3:USING PRETRAINED BERT MODEL ON SQUAD DATASET
    - TRANSFER LEARNING AND BERT MODEL
    - PREPROCESSING
    - CREATING MODEL AND FINE TUNING
    - EVALUATING ON TEST DATASET
    - PREDICTIONS

7. CONCLUSIONS

8. REFERENCES

ABOUT COMPANY

Hindustan Aeronautics Limited (HAL) is a premier Aerospace company in Asia which is engaged in Design, Development and Manufacture of military and civil aircraft for over 75 years. It was established as Hindustan Aircraft in Bangalore in 1940 by Seth Walchand Hirachand to produce military aircraft for Indian Air force.

HAL is an organization where integrated airborne systems in the form of fighter aircraft and helicopters are conceived, developed, manufactured and serviced. It is one of the few corporate giants in Asia whose capabilities span the entire range of activities from product conception to after sale support. HAL is also involved in manufacture and assembly of structure required for India's Space programs.

Hindustan Aeronautics Limited is the largest Public Sector Unit (PSU) under Ministry of Defense and is a Navratna Company. The company takes up maintenance and overhaul services to cover the life cycle requirement of all old and new products. Presently, 13 types of aircraft/helicopters and 17 types of engines are being overhauled. Additionally, facilities for repair/overhaul of various accessories and avionics integrated on aircraft of Russian, Western and Indigenous designs are also provided. HAL has 20 production Divisions, 10 Research and Development Centers and one Facility Management Division.

HAL Nashik has 3 divisions:

- Aircraft Manufacturing Division (AMD)
- Aircraft Overhaul Division (AOD)
- Aircraft Upgrade Research and Design Centre (AURDC)

Aircraft Division Nasik, established in the year 1964 for license manufacture of MiG-21FL aircraft & K-13 Missiles, is located at Ojhar, 24 kilometers from Nasik and approximately

200 kilometers from Mumbai in the state of Maharashtra. The division since then manufactured other MiG variants; viz MiG-21M, MiG-21 BIS, MiG-27 M and the state-of-the-art aircraft i.e. Su-30 MKI. Along with manufacturing, the division also carries out overhaul of the MiG series aircraft and started ROH of Su-30 MKI.

With the introduction of New a state-of-the art project i.e. Su-30 MKI and for smooth activities and operation of the division then Aircraft Division, Nasik is de-lineated into two Divisions viz Aircraft Manufacturing Division (AMD) for manufacturing activities and Aircraft Overhaul Division (AOD) for Repair & Overhauling (ROH) activities.

SU-30MKI

SU-30MKI Touchdown

The Division is currently engaged in the following activities:

- Manufacturing of Su-30 MKI Aircraft.
- Supply of spares / units /consumables and aerospace fasteners.
- Support to HAL and Non-HAL made aircraft of Russian Origin in terms of Life extension, Modifications, Site Repairs and investigations.
- Diversification into Civil Aircraft manufacture by taking up work packages.
- Export of MiG spares to Egypt, Syria, Vietnam, Malaysia, Algeria, Poland & Russia.

MiG 29

MiG 27 flying in Formation

AURDC Division:

- AURDC was established as Design Department in 1964 to provide design support to the manufacturing program, it has grown into a full-fledged R&D center named Aircraft Upgrade Research and Design Centre (AURDC). With vast experience on Su30MKI, MiG series aircraft in design and development for product improvements, role capability enhancement, indigenization, technology up-gradation, structural integrity studies for life extension, flight test analysis and mid-life upgrade are its strengths.
- AURDC over the years gained experience in design and development of new airborne system, Integration of systems and weapons in the aircraft. Evolved many modifications in terms of reliability, maintainability, operation, capability, enhancement and etc.

INTRODUCTION

Aircraft manufacturing and overhauling is a diverse and vast process. In each step of the process, we need to make sure the quality and standards are maintained with utmost importance. Furthermore, each part or equipment of the aircraft is manufactured locally at the Aircraft Manufacturing Division (AMD) of HAL, Nashik, or has to be sourced from another HAL manufacturing facilities, and some specialized licensed parts are also sourced from. With this level of the abstruse, multistage process it fairly regular for minute things to go wrong , and resolving each of this error technician have to refer to a lot of documentation , manuals which consumes a lot of time and slows down the process of production . We will try to automate this process of searching for the answers by implementing question answering models using natural language processing .

Question answering (QA) is a computer science discipline within the fields of information retrieval and natural language processing (NLP), which is concerned with building systems that automatically answer questions posed by humans in a natural language.

Question answering research attempts to deal with a wide range of question types including: fact, list, definition, *How*, *Why*, hypothetical, semantically constrained, and cross-lingual questions.

- *Closed-domain* question answering deals with questions under a specific domain (for example, medicine or automotive maintenance), and can exploit domain-specific knowledge frequently formalized in ontologies. Alternatively, *closed-domain* might refer to a situation where only a limited type of questions are accepted, such as questions asking for descriptive rather than procedural information. Question answering systems in the context of machine reading applications have also been constructed in the medical domain, for instance related to Alzheimer's disease.
- *Open-domain* question answering deals with questions about nearly anything, and can only rely on general ontologies and world knowledge. On the other hand, these systems usually have much more data available from which to extract the answer.

In this project we will try to implement the closed domain question answering model where the model is suppose to answer the question based on the given context .

SYSTEM CONFIGURATION, SOFTWARE AND EQUIPMENT

PC Model: HP ZBook 17

Processor: Intel(R) Core(TM) i7-4600M 2 Core(s) 4 Logical Processor(s) @2.9GHz

RAM: 8 GB

OS: Windows 7 Professional 64-bit (6.1, Build 7601)

Display: 1920*1080 (32 bit) (60Hz)

Software used:

- Anaconda Navigator/Prompt
- Jupyter Notebook

Programming Language:

- Python 3.8.5

Datasets:

Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset, consisting of questions posed by crowd workers on a set of Wikipedia articles, where the answer to every question is a segment of text, or *span*, from the corresponding reading passage, or the question might be unanswerable.

PROJECT

1. Problem Statement

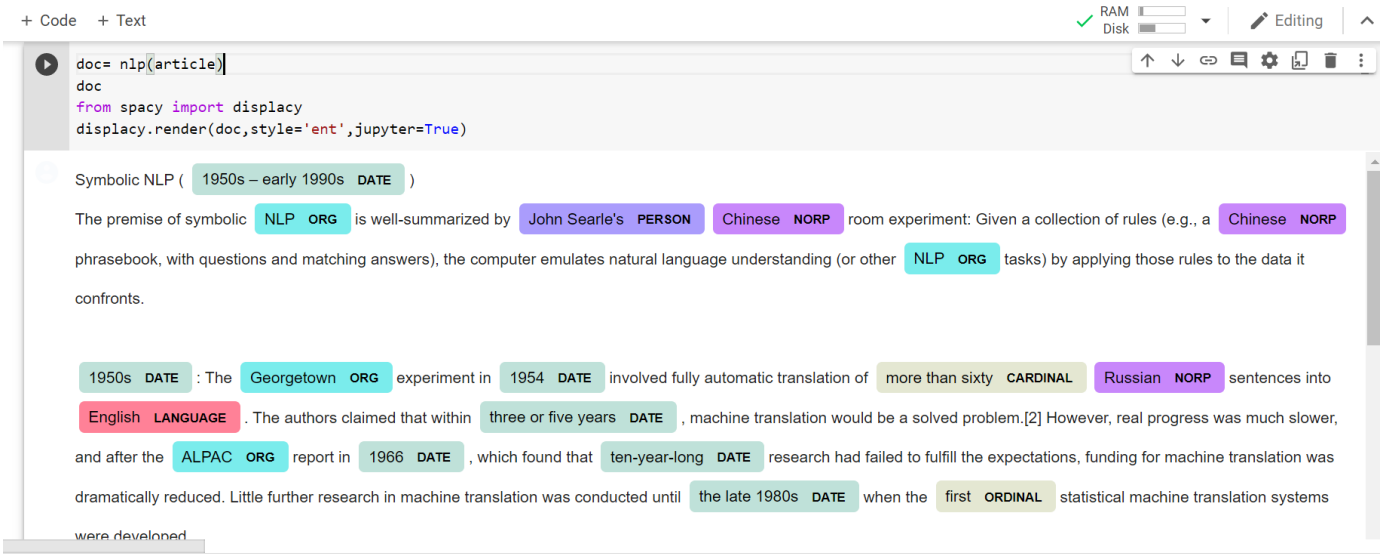
To Design and Develop a Machine Learning (Artificial Intelligence) Model for implementing Comprehension Question Answering Model using Natural Language Processing and neural network. Squad dataset is used for both training and validation of the model

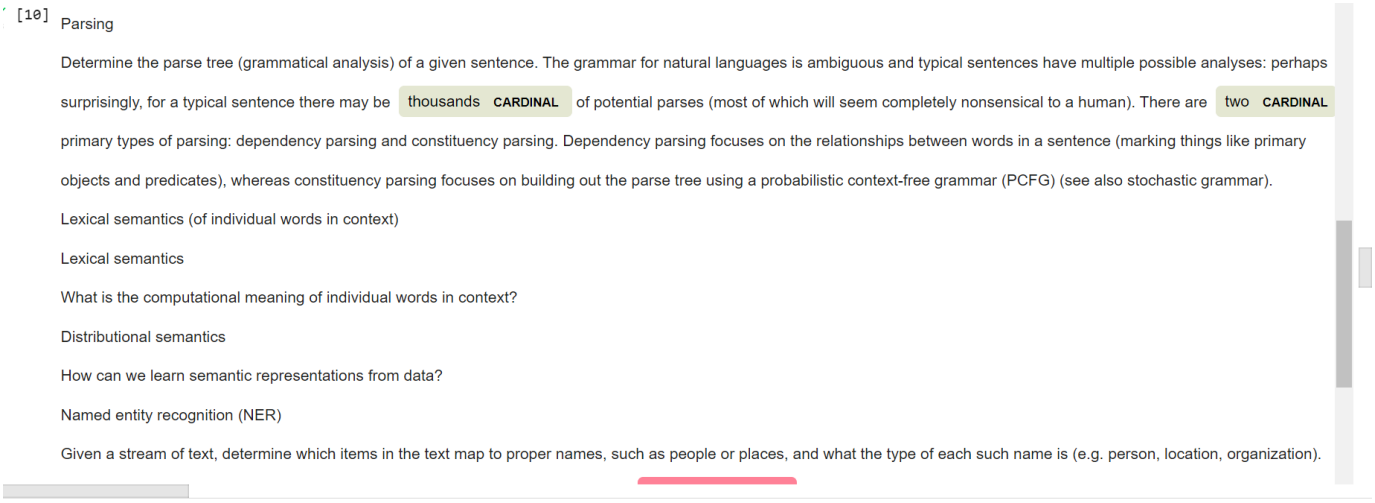
MODEL 1 : Basic Model using NLP Libraries Spacy ,NLTK

This model works a small English pipeline trained on written web text (blogs, news, comments), that includes vocabulary, syntax and entities.

For this model an article from Wikipedia is given as a comprehension and based on the comprehension model tries to answer the asked questions in the form of a chatbot.

UNDERSTANDING THE DATA:





EXPLORING THE ENTITIES:

```
for entity in doc.ents:
    if entity.label_=='CARDINAL':
        print(entity.text)
```

```
more than sixty
two
1990s–2010s
2015,[19
thousands
at least five
thousands
two
more than one
2.bring
3
4
5.manage
```

```
[11] for entity in doc.ents:
         if entity.label_=='PERSON':
             print(entity.text)
```

```
John Searle's
Joseph Weizenbaum
TaleSpin
Lehnert
Lehnert
Moore
Markov
```

PREPROCESSING:

While preprocessing the following changes in the sentences of the comprehension is been done:

1. Senetence.lower : To convert the sentence to lower case
2. “ . ” are replaced by space

3. Stop words , punctuation marks, numbers are removed from the article :Stop words are the commonly used words in a language i.e. a, the, is , are ,there , here..etc

```
def preprocessing(sentence):
    sentence=sentence.lower()
    sentence = sentence.replace('.',' ')
    tokens=[]
    tokens=[token.text for token in nlp(sentence) if not (token.is_stop or token.like_num or token.is_punc
    tokens=' '.join([element for element in tokens])
    return tokens
```

```
[ ] article_cleaned=preprocessing(article)
    article_cleaned
```

NATURAL LANGUAGE ROCESSING

TFIDFVectorizer:

TFIDF works by proportionally increasing the number of times a word appears in the document but is counterbalanced by the number of documents in which it is present. Hence, words like ‘this’, ’are’ etc., that are commonly present in all the documents are not given a very high rank. However, a word that is present too many times in a few of the documents will be given a higher rank as it might be indicative of the context of the document.

Term Frequency:

Term frequency is defined as the number of times a word (i) appears in a document (j) divided by the total number of words in the document.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

Inverse Document Frequency:

Inverse document frequency refers to the log of the total number of documents divided by the number of documents that contain the word. The logarithm is added to dampen the importance of a very high value of IDF.

$$idf(w) = log(\frac{N}{df_t})$$

TFIDF is computed by multiplying the term frequency with the inverse document frequency.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

Let us now see an illustration of TFIDF in the following sentences, that we refer to as documents.

Document 1: Text processing is necessary.

Document 2: Text processing is necessary and important.

Word	TF		IDF	
	Doc 1	Doc 2		D
Text	1/4	1/6	log (2/2) = 0	
Processing	1/4	1/6	log (2/2) =0	
Is	1/4	1/6	log (2/2) =0	
Necessary	1/4	1/6	log (2/2) =0	
And	0/4	1/6	log (2/1) =0.3	
Important	0/4	1/6	log (2/1) =0.3	

The above table shows how the TFIDF of some words are zero and some words are non-zero depending on their frequency in the document and across all documents.

The limitation of TFIDF is again that this vectorization doesn't help in bringing in the contextual meaning of the words as it is just based on the frequency.

COSINE SIMILARITY:

Cosine similarity is a metric, helpful in determining, how similar the data objects are irrespective of their size. We can measure the similarity between two sentences in Python using Cosine Similarity. In cosine similarity, data objects in a dataset are treated as a vector. The formula to find the cosine similarity between two vectors is –
Cos(x, y) = x . y / ||x|| * ||y||

where,

- x . y = product (dot) of the vectors 'x' and 'y'.
- ||x|| and ||y|| = length of the two vectors 'x' and 'y'.
- ||x|| * ||y|| = cross product of the two vectors 'x' and 'y'.

WORKING:

```
welcome_words_input=('hey','hello','hii')
welcome_words_output=('hey','hello','how u doin','welcome ')
original_sentences = [sentence for sentence in nltk.sent_tokenize(article)]

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import random
def welcome_message(text):
    for word in text.split():
        if word.lower() in welcome_words_input:
            return random.choice(welcome_words_output)
def answer(user_text,threshold=0.001):
    cleaned_sentences=[]
    for sentence in original_sentences :
        cleaned_sentences.append(preprocessing(sentence))

    chatbot_answer=''
    user_text=preprocessing(user_text)
    cleaned_sentences.append(user_text)
    tfidf=TfidfVectorizer()
    X_sentences=tfidf.fit_transform(cleaned_sentences)
    similarity = cosine_similarity(X_sentences[-1],X_sentences)
    sentence_index=similarity.argsort()[0][-2]

    if similarity[0][sentence_index]<threshold:
        chatbot_answer+='no answer found'
    else:
        chatbot_answer+=original_sentences[sentence_index]


    return chatbot_answer
```

```
cont=True
print('hello !welcome back i will like to answer your questions')
while cont==True:
    user_text=input()
    if user_text !='quit':
        if welcome_message(user_text)!=None:
            print('Chatbot: '+welcome_message(user_text))
        else:
            print('Chatbot:')
            print(answer(user_text))
    else:
        cont=False
        print('Chatbot: I ll see you again hope your queries are cleared')
```

OUTPUT:

```
hello !welcome back i will like to answer your questions
hey
Chatbot: how u doin
what is nlp
Chatbot:
 Symbolic NLP (1950s – early 1990s)
The premise of symbolic NLP is well-summarized by John Searle's Chinese room experiment: Given a collection of rules (e.g., a Chinese phrasebook, with question
what is natural language processing
Chatbot:
Starting in the late 1980s, however, there was a revolution in natural language processing with the introduction of machine learning algorithms for language pr
who is Joseph Weizenbaum
Chatbot:
1960s: Some notably successful natural language processing systems developed in the 1960s were SHRDLU, a natural language system working in restricted "blocks
what is named entity recognition
Chatbot:
Entity linking
Many words - typically proper names - refer to named entities; here we have to select the entity (a famous individual, a location, a company, etc.)
quit
Chatbot: I ll see you again hope your queries are cleared
```

MODEL 2 : Yes or No Answering on a limited vocabulary using deep learning and neural network

Loading the dataset:

```
[1] import pickle
    import numpy as np
```

```
[2] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[ ] path='/content/drive/My Drive'
```

```
[3] with open("/content/drive/My Drive/train_qa.txt", "rb") as fp:   # Unpickling
        train_data =  pickle.load(fp)
```

```
[4] with open("/content/drive/My Drive/test_qa.txt", "rb") as fp:   # Unpickling
        test_data =  pickle.load(fp)
```

UNDERSTANDING THE DATA:

```
[9]  train_data[0]
```

```
(['Mary',
  'moved',
  'to',
  'the',
  'bathroom',
  '.',
  'Sandra',
  'journeyed',
  'to',
  'the',
  'bedroom',
  '.'],
 ['Is', 'Sandra', 'in', 'the', 'hallway', '?'],
 'no')
```

```
[10]  ' '.join(train_data[0][0])
```

```
'Mary moved to the bathroom . Sandra journeyed to the bedroom .'
```

```
[11]  ' '.join(train_data[0][1])
```

```
'Is Sandra in the hallway ?'
```

```
[12]  train_data[0][2]
```

```
'no'
```

Creating the vocab :

```
[13] # Create a set that holds the vocab words
     vocab = set()
```

```
     all_data = test_data + train_data
```

```
[62] for story, question , answer in all_data:

         vocab = vocab.union(set(story))
         vocab = vocab.union(set(question))
```

```
[63] vocab.add('no')
     vocab.add('yes')
```

```
[64] vocab
```

```
{'.',
 '?',
 'Daniel',
 'Is',
 'John',
 'Mary',
 'Sandra',
 'apple',
 'back',
 'bathroom',
 'bedroom',
 'discarded',
```

TOKENIZING THE DATA:

```
[72] #from tensorflow import keras
     from tensorflow.keras.preprocessing.sequence import pad_sequences
     from tensorflow.keras.preprocessing.text import Tokenizer
```

```
[73] # integer encode sequences of words
     tokenizer = Tokenizer(filters=[])
     tokenizer.fit_on_texts(vocab)
```

```
     tokenizer.word_index
```

```
{'.': 30,
 '?': 36,
 'apple': 21
```

Preparing the training dataset:

```
[75] train_story_text = []
     train_question_text = []
     train_answers = []

     for story,question,answer in train_data:
         train_story_text.append(story)
         train_question_text.append(question)
```

```
[76] train_story_seq = tokenizer.texts_to_sequences(train_story_text)
```

VECTORIZING THE DATA:

```
[93] def vectorize_stories(data, word_index=tokenizer.word_index, max_story_len=max_story_len,max_question_len=max_question_len):

        X = []
        Xq = []
        Y = []
        for story, query, answer in data:
            x = [word_index[word.lower()] for word in story]
            xq = [word_index[word.lower()] for word in query]
            y = np.zeros(len(word_index) + 1)
            y[word_index[answer]] = 1
            X.append(x)
            Xq.append(xq)
            Y.append(y)
        return (pad_sequences(X, maxlen=max_story_len),pad_sequences(Xq, maxlen=max_question_len), np.array(Y))
```

```
[94] inputs_train, queries_train, answers_train = vectorize_stories(train_data)
```

```
[95] inputs_test, queries_test, answers_test = vectorize_stories(test_data)
```

Creating the Model

1. Creating the encoders

```
[100] from tensorflow.keras.models import Sequential, Model
      from tensorflow.keras.layers import Embedding
      from tensorflow.keras.layers import Input, Activation, Dense, Permute, Dropout
      from tensorflow.keras.layers import add, dot, concatenate
      from tensorflow.keras.layers import LSTM
```

```
[101] input_sequence = Input((max_story_len,))
      question = Input((max_question_len,))
```

```
[102] input_encoder_m = Sequential()
      input_encoder_m.add(Embedding(input_dim=vocab_size,output_dim=64))
      input_encoder_m.add(Dropout(0.3))
```

```
[103]
      input_encoder_c = Sequential()
      input_encoder_c.add(Embedding(input_dim=vocab_size,output_dim=max_question_len))
      input_encoder_c.add(Dropout(0.3))
```

```
[104] question_encoder = Sequential()
      question_encoder.add(Embedding(input_dim=vocab_size,
                                     output_dim=64,
                                     input_length=max_question_len))
      question_encoder.add(Dropout(0.3))
```

2. Creating the model:

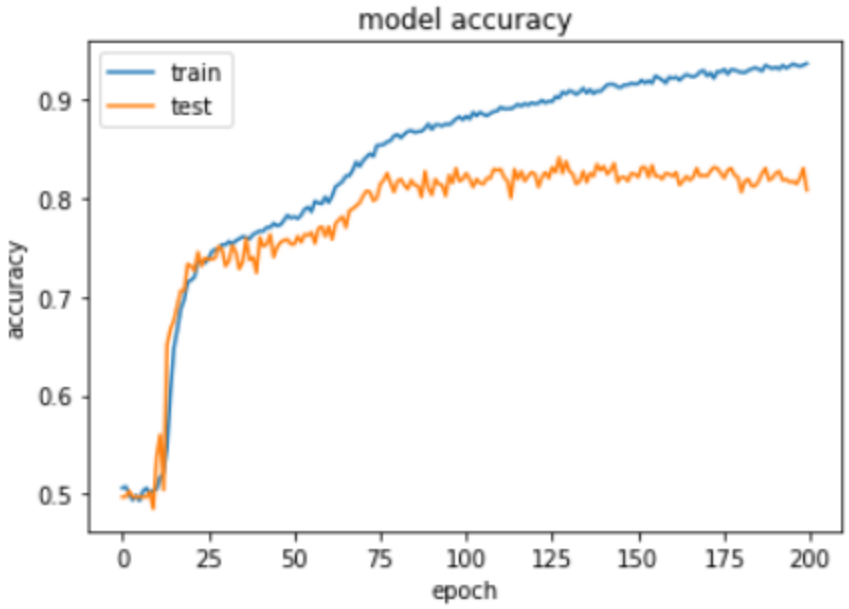
```
115] match = dot([input_encoded_m, question_encoded], axes=(2, 2))
     match = Activation('softmax')(match)
     response = add([match, input_encoded_c])
     response = Permute((2, 1))(response)
     answer = concatenate([response, question_encoded])
     answer = LSTM(32)(answer)
     answer = Dropout(0.5)(answer)
     answer = Dense(vocab_size)(answer)
     answer = Activation('softmax')(answer)
     model = Model([input_sequence, question], answer)
     model.compile(optimizer='rmsprop', loss='categorical_crossentropy',
                   metrics=['accuracy'])
```

```
# train
history = model.fit([inputs_train, queries_train], answers_train,batch_size=32,epochs=200,validation_data=([inputs_test, queries_test],
    monitor='accuracy',
    min_delta=0,
    patience=0,
    verbose=1,
    mode='max',
    baseline=None,
    restore_best_weights=False
)])
```

Model.summary:

```
Model: "model_1"
__________________________________________________________________________________________________
 input_3 (InputLayer)           [(None, 156)]        0           []

 input_4 (InputLayer)           [(None, 6)]          0           []

 sequential_3 (Sequential)      (None, None, 64)     2432        ['input_3[0][0]']

 sequential_5 (Sequential)      (None, 6, 64)        2432        ['input_4[0][0]']

 dot_1 (Dot)                    (None, 156, 6)       0           ['sequential_3[0][0]',
                                                                  'sequential_5[0][0]']

 activation_3 (Activation)      (None, 156, 6)       0           ['dot_1[0][0]']

 sequential_4 (Sequential)      (None, None, 6)      228         ['input_3[0][0]']

 add_2 (Add)                    (None, 156, 6)       0           ['activation_3[0][0]',
                                                                  'sequential_4[0][0]']

 permute_2 (Permute)            (None, 6, 156)       0           ['add_2[0][0]']

 concatenate_2 (Concatenate)    (None, 6, 220)       0           ['permute_2[0][0]',
                                                                  'sequential_5[0][0]']

 lstm_1 (LSTM)                  (None, 32)           32384       ['concatenate_2[0][0]']

 dropout_7 (Dropout)            (None, 32)           0           ['lstm_1[0][0]']

 dense_1 (Dense)                (None, 38)           1254        ['dropout_7[0][0]']

 activation_4 (Activation)      (None, 38)           0           ['dense_1[0][0]']

==================================================================================================
Total params: 38,730
Trainable params: 38,730
Non-trainable params: 0
```

MODEL ACCURACY PLOT:



Testing the model:

```
[132] story =' '.join(word for word in test_data[0][0])
      print(story)

      Mary got the milk there . John moved to the bedroom .

[133] query = ' '.join(word for word in test_data[0][1])
      print(query)

      Is John in the kitchen ?

[134] print("True Test Answer from Data is:",test_data[0][2])

      True Test Answer from Data is: no

      #Generate prediction from model
      val_max = np.argmax(pred_results[0])

      for key, val in tokenizer.word_index.items():
          if val == val_max:
              k = key

      print("Predicted answer is: ", k)
      print("Probability of certainty was: ", pred_results[0][val_max])

      Predicted answer is:  down
      Probability of certainty was:  0.033008404
```

Making prediction using he vocabulary :

```
[138] my_question = "Is the football in the garden ?"

[139] my_question.split()

      ['Is', 'the', 'football', 'in', 'the', 'garden', '?']

[140] mydata = [(my_story.split(),my_question.split(),'yes')]

[141] my_story,my_ques,my_ans = vectorize_stories(mydata)

[142] pred_results = model.predict(([ my_story, my_ques]))

[143] #Generate prediction from model
      val_max = np.argmax(pred_results[0])

      for key, val in tokenizer.word_index.items():
          if val == val_max:
              k = key

      print("Predicted answer is: ", k)
      print("Probability of certainty was: ", pred_results[0][val_max])

      Predicted answer is:  down
      Probability of certainty was:  0.029591842
```

MODEL 3 :Using pretrained bert model on the SQuAD dataset

TRANSFER LEARNING AND BERT MODEL:

Transfer learning:

Transfer learning is a machine learning method where we reuse a pre-trained model as the starting point for a model on a new task. To put it simply—a model trained on one task is repurposed on a second, related task as an optimization that allows rapid progress when modeling the second task.

BERT Model:

BERT is an open source machine learning framework for natural language processing (NLP). BERT is designed to help computers understand the meaning of ambiguous language in text by using surrounding text to establish context. The BERT framework was pre-trained using text from Wikipedia and can be fine-tuned with question and answer datasets.

BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on Transformers, a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection. (In NLP, this process is called *attention*.

PREPROCESSING:

```
new_squad = []



for group in squad['data']:
    for paragraph in group['paragraphs']:
        context =paragraph['context']
        for qa_pair in paragraph['qas']:
            question=qa_pair['question']
            if 'answers' in qa_pair.keys() and len(qa_pair['answers'])>0:
                answer=qa_pair['answers'][0]['text']
            elif 'plausible_answers' in qa_pair.keys() and len(qa_pair['plausible_
                answer = qa_pair['plausible_answers'][0]['text']
            else:
                answer=None
            new_squad.append({
                'question':question,
                'answer':answer,
                'context':context
            })
```

CREATING MODEL AND FINE TUNING :

```
[9]  from transformers import BertTokenizer, BertForQuestionAnswering
     modelname='deepset/bert-base-cased-squad2'
     tokenizer=BertTokenizer.from_pretrained(path)
     model = BertForQuestionAnswering.from_pretrained(path)
```

```
[10] from transformers import pipeline
     qa = pipeline('question-answering',model=model,tokenizer=tokenizer)
```

EVALUATING ON TEST DATASET :

```
answers[:5]
```

```
[{'predicted': 'France', 'true': 'France'},
 {'predicted': '10th and 11th centuries', 'true': '10th and 11th centuries'},
 {'predicted': 'Denmark, Iceland and Norway',
  'true': 'Denmark, Iceland and Norway'},
 {'predicted': 'Rollo', 'true': 'Rollo'},
 {'predicted': '10th', 'true': '10th century'}]
```

```
[ ] from rouge import Rouge
    rouge=Rouge()
```

```
[ ] model_out = [ans['predicted'] for ans in answers]
    reference = [ans['true'] for ans in answers]
```

```
[ ] rouge.get_scores(model_out,reference,avg=True)
```

```
{'rouge-1': {'f': 0.9333333284444445, 'p': 1.0, 'r': 0.9},
 'rouge-2': {'f': 0.399999998, 'p': 0.4, 'r': 0.4},
 'rouge-l': {'f': 0.9333333284444445, 'p': 1.0, 'r': 0.9}}
```

PREDICTIONS:

```
[15] context = '''There was once a hare who was friends with a tortoise. One day, he challenged the tortoise to a race. Seeing how slow the tortoise was going, the hare thought he'll win this easily.
      So he took a nap while the tortoise kept on going. When the hare woke up, he saw that the tortoise was already at the finish line. Much to his chagrin, the tortoise won the race while
       he was busy sleeping.'''
     qa({
         'question': 'who is slower ?',
         'context': context
     })

     {'answer': 'tortoise.', 'end': 54, 'score': 0.0006804025615565479, 'start': 45}
```

```
context = '''After flying a long distance, a thirsty crow was wandering the forest in search of water. Finally, he saw a pot half-filled with water.
He tried to drink from it but his beak wasn't long enough to reach the water inside. He then saw pebbles on the ground and one by one, he put them in the pot until the water rose to the brim.
 The crow then hastily drank from it and quenched his thirst.'''
qa({
    'question': 'where was the crow going',
    'context': context
})

{'answer': 'the forest in search of water.',
 'end': 89,
 'score': 0.1486070070858002,
 'start': 59}
```

```
[17] context ='''The ant and the grasshopper were good friends. In the summer, the ant works hard to fill his storage with food. While the grasshopper was enjoying the fine weather and playing all day.
     When winter came, the ant was lying cozily in his home surrounded by the food he stored during the summer. While the grasshopper was in his home, hungry and freezing.
     He asked the ant for food and the ant gave him some. But it wasn't enough to last the entire winter. When he tried to ask the ant again, the latter replied:
      "I'm sorry my friend but my food is just enough for my family to last until the end of winter. If I give you more, we too will starve.
      We had the entire summer to prepare for the winter but you chose to play instead.'''
    qa({
        'question': 'who worked hard ',
        'context': context
    })

    {'answer': 'the ant works hard to fill his storage with food.',
     'end': 111,
     'score': 0.08409392088651657,
     'start': 62}
```

```
[13] #using elastic search
     context = '''The Intergovernmental Panel on Climate Change (IPCC) is a scientific intergovernmental body ur
      It was first established in 1988 by two United Nations organizations, the World Meteorological Organizatio
       and later endorsed by the United Nations General Assembly through Resolution 43/53. Membership of the IPC
       The IPCC produces reports that support the United Nations Framework Convention on Climate Change (UNFCCC)
       The ultimate objective of the UNFCCC is to \"stabilize greenhouse gas concentrations in the atmosphere at
        interference with the climate system\". IPCC reports cover \"the scientific, technical and socio-economi
        human-induced climate change, its potential impacts and options for adaptation and mitigation.'''

     qa({
         'question': 'What organization is the IPCC a part of?',
         'context': context
     })
```

```
{'answer': 'United Nations,',
 'end': 133,
 'score': 0.4684184193611145,
 'start': 118}
```

```
     context = '''There was once a hare who was friends with a tortoise. One day, he challenged the tortoise to
      So he took a nap while the tortoise kept on going. When the hare woke up, he saw that the tortoise was alr
      Much to his chagrin, the tortoise won the race while he was busy sleeping.'''
     qa({
         'question': 'who won the race ',
         'context': context
     })
```

```
{'answer': 'the tortoise',
 'end': 349,
 'score': 0.099199078977108,
 'start': 337}
```

CONCLUSION

The project titled "**QUESTION ANSWERING MODEL BASED ON ARTIFICIAL INTELLIGENCE AND NATURAL LANGUAGE PROCESSING**" has been completed within the stipulated time of the project. The objective of the project was to Design and Develop an Artificial Intelligent model which should be able to answers questions from the given context , SQUAD dataset has been used for training and testing model. The model was successfully developed and is ready to be put in real-world systems.

This internship has given me a new perspective on various topics regarding military aircrafts. It has given a lot of exposure to the industrial side of a corporation. Through this, I got acquainted with the workflows, procedures, and safety measures involved in the industry. The Training Department gave me a brief knowledge about HAL, Nashik. The AURDC division gave me an in-depth overview of aircraft designing and various avionic system integration in aircrafts.

REFERENCES

To gather information about the company and fighter aircrafts-

https://hal-india.co.in/

https://en.wikipedia.org/wiki/Hindustan_Aeronautics_Limited

https://en.wikipedia.org/wiki/Sukhoi_Su-30MKI

Theoretical knowledge and understandings-

https://machinelearningmastery.com/gentle-introduction-bag-words-model/

https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/

https://scikit-learn.org/stable/modules/naive_bayes.html

https://huggingface.co/blog/bert-101

Documentation of various python 3 libraries used in the process-

https://docs.python.org/3/

https://pandas.pydata.org/docs/

https://numpy.org/doc/

https://scikit-learn.org/stable/

https://matplotlib.org/

https://seaborn.pydata.org/