Visual Question Answering system using high performance machine learning techniques

Dr. Lavika Goel

Department of Computer Science and Engineering

Malaviya National Institute of Technology, Jaipur

lavika.cse@mnit.ac.in

Introduction

- Visual Question Answering (VQA) is a very popular and very important research field.
- Being able to look at an image and ask questions to our computers can prove to be useful to people in numerous ways.
- A visually impaired person can move about more independently by simply asking questions about his neighborhood, whereabouts etc.
- Another major practical implication of VQA is human-computer interaction in order to get visual content - a person in a foreign country can explore the place by just enquiring what his eyes can see.
- The aim is to examine the relative accuracy of various image captioning models along with BERT as a base for question answering on the task of visual question answering and discuss the future scope of VQA performance at the aid of image captioning models.

Understanding VQA

- VQA is a combination of two systems- image processing and question answering(Q&A).
- The system that generates answers based on textual information have evolved a lot and are up to 89% accuracy (BERT).
- The task left in VQA is to generate textual information from the image that can very well describe the image, thus producing better results than raw systems.
- These systems rely on various fields of computer vision namely object detection, classification and comparison of detected data with the worldly knowledge.

Understanding VQA (Contd...)

- We employ four state-of-the-art image captioning models to generate direct textual information describing the image.
- The image captioning models identify the most significant parts of the image and generate a caption accordingly describing the image.
- Our goal is to develop a visual question answering system using BERT model for question answering and four state-of-the-art methods of image captioning to develop the image captions namely,

BUTD captioning	Show and tell model
Microsoft's Captionbot	Show, Attend & Tell Model

Examples of VQA

In the Example 1 and 2 below, there is an input image along with an input question and the model is supposed to answer the question based on image context.

Example 1:

Q: What is being played?

A: Football.

Example 2:

Q: How many giraffes are there?

A: Two

Datasets

There are numerous datasets for the task of visual answering. Here we describe the most frequently tested dataset, VQA and COCO-QA dataset.

- **Toronto COCO-QA Dataset :** The dataset consists of 123287 images from the coco dataset and 78736 training question-answer pairs, along with 38948 testing images. The questions mainly consist of 4 types i.e. object, number, color, location with one-word answers. This is a relatively easy dataset.

Datasets (Contd...)

- **VQA Dataset :** This dataset deals with the binary classification of data (i.e. questions with answers yes/no), counting problem (how many giraffes?) amongst others. It is the most commonly used dataset and has been considered standard to test results on for some time. It has over 260,000 images with at least 3 questions per image. It also provides answers that are correct as well as those that seem correct but are actually not.

Examples

DAQUAR 1553

What is there in front of the sofa?

Ground truth: table

COCOQA 5078

How many leftover donuts is the red bicycle holding?

Ground truth: three

Sample images and questions for Toronto COCO-QA

What color are her eyes?
What is the mustache made of?

How many slices of pizza are there?
Is this a vegetarian pizza?

Sample questions from VQA

Methodology

The goal is to get abstract information from an image and answer the corresponding question given to us. The problem can be divided into two parts

1. Extraction of relevant information from an image.
2. Answering a given question based on the information generated from the image.

We employ state-of-the-art models for both the tasks individually, we modify BERT's output and input hidden layers to employ it for context-based question answering. For the job of extracting information from images, we use four state-of-the-art models for image captioning.

The aim is to define and test the combination of different image captioning algorithms and question answering model to model an image as the context for question answering.

Methodology (Contd...)

The image captioning models generate the most appropriate one-line caption for an image, this caption is then passed into the BERT-question answering system. BERT uses image caption as its context to answer the question.

1. BERT + BUTD
2. BERT + Show and Tell model
3. BERT + Captionbot
4. BERT + Show Attend and Tell

Methodology

We started by training four models that are used to generate a sentence for images. These are the state-of-the-art models for image captioning. After this, we trained BERT model for question answering system. Finally, four combined models were generated for visual question answering.

- **BUTD Captioning (Pythia)**

  Generally, just top-down attention methods are used by people for image captioning. Here, a combination of bottom-up and top-down methods of the visual attention are used in combination. This was done to get better outcomes. Faster R-CNN's were used to implement bottom-up. This was done as Faster R-CNN provide a very natural way of executing bottom-up.

BUTD (Contd...)

- A combination of a network that helps the system be aware of the network using proposals made by Regional Proposals Network (RPN) for regions. Region generation proposals are done using selective search in Fast R-CNN, which costs much more than RPN. This is because Object Detection Network splits up the work with RPN. So, basically RPN tries to find the maximum number of objects it can detect and returns those boxes. These boxes are called anchors. Faster R-CNN work majorly using these anchors.
- After RPN has processed the image, it returns a set of few anchors that it proposes for the classifier to work on. This helps us distinguish between foreground and background. This can speed up the work of regressors and classifiers as they need to focus more on these boxes only.

BUTD (Contd...)
The Classifier of Background and Foreground

This task is divided into two sub tasks:

- We start by creating a dataset for our classifier. The dataset must have labeled the positions foreground is present as ground truth. So, whatever boxes overlap very less with "ground truth" are labeled as background and the one that coincide more are labeled as foreground.
- Now, the features of anchors need to be decided. These include the size of stride, number of anchors (2 labels per anchor, making the number of features as twice the number of anchors), the activation function to be used etc.

BUTD (Contd...)
The Regressor of Bounding Box

We must pass only the anchors labeled as foreground to the regressors. This is because we have values available to compare our outputs with i.e. the value of ground truth. Hence, we cannot in any case use background images. The number of positions that we get back defines the depth of our feature map (number of anchors x number of positions).

BUTD (Contd...)
ROI Pooling

RPN returns a set of varying sizes of boxes that it feels are part of the foreground. We cannot directly pass these results to a CNN. Here is where ROI (Region of Interest) Pooling comes into play.

This reduces our image (i.e. feature maps) into numerous feature maps of the same size. The number of feature maps generated is also same and a parameter that we decide. After that, max pooling can be applied on each individual feature map. The output received can be used for various purposes according to our needs.

BUTD (Contd...)

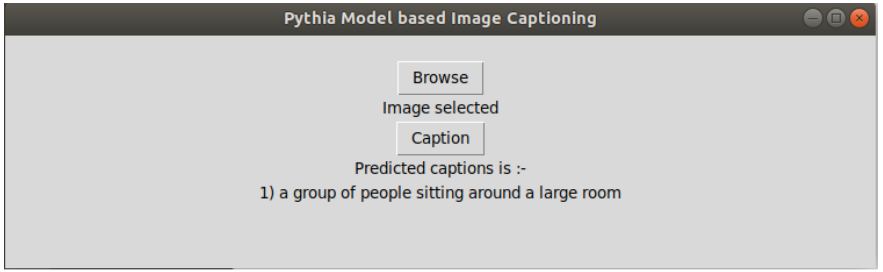
Code for captioning:

```
import os
from os import listdir
from os.path import isfile,join
mypath='/content/drive/My Drive/test2015/'
filepath='/content/drive/My Drive/output_caps/pythia_op.txt'
onlyfiles=[f for f in listdir(mypath) if isfile(join(mypath,f))]
onlyfiles.sort()
i=1
onlyfiles=onlyfiles[i-1:]
print(len(onlyfiles))
fw=open(filepath,'a')
for filename in onlyfiles:
  tokens=demo.predict(mypath+'/'+filename)
  cap="1) "+demo.caption_processor(tokens.tolist()[0])["caption"]
  print(i)
  print(cap)
  fw.write("\n Captions for image %s" % os.path.basename(filename))
  fw.write('\n'+str(i))
  fw.write('\n'+cap)
  i+=1
fw.close()
```

The parameters for Pythia are given below:

```
class PythiaDemo:
  TARGET_IMAGE_SIZE = [448, 448]
  CHANNEL_MEAN = [0.485, 0.456, 0.406]
  CHANNEL_STD = [0.229, 0.224, 0.225]
```

Example:



Methodology (Contd...)

Show and Tell: A Neural Image Caption Generator:

- Show and Tell model has been developed by researchers at Google.
- It tries to solve the problem of describing images with the help of machine learning models.
- Description part is the problem of Natural Language Processing whereas making sense of the image is at the heart of computer vision. This image captioning model uses both these techniques (with already existing state-of-the-art models) to develop a generative model.
- It uses probability to decide if the caption generated fits the image and tries to maximize the probability. After input is provided in the form of images, it tries to make a sequence of words using a dictionary (taking words that fit the image).
- The probability function P(S|I) is the probability of S (a sequence of words) being the description, given I is the image. A combination of encoder and decoder networks are used, where encoder is a CNN where decoder is an RNN. A sentence is input to the encoder, which encodes it into a fixed size vector. This vector is then given to decoder as input and this decoder outputs the final sentence.

Show and Tell (Contd...)
Model

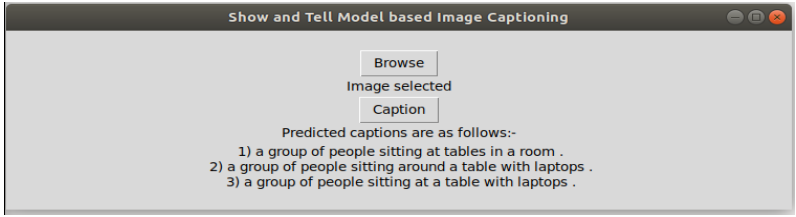
The Show and Tell model work on probability to generate image descriptions. It uses a combination of neural networks along with probability distribution to do the same.

- As Long-Short Term Memory (LSTM) net tends to have a long dependency, it is known to show state-of-the-art results on many sequencing problems. So, we shall be using LSTMs.
- We shall use a CNN (Convolutional Neural Network) model to process our images.

Show and Tell (Contd...)

Code for captioning the image using show and tell.

```
$ sudo docker run -v $PWD:/opt/app \
-e PYTHONPATH=$PYTHONPATH:/opt/app \
-it colemurray/medium-show-and-tell-caption-generator  \
python3 /opt/app/medium_show_and_tell_caption_generator/inference.py \
--model_path /opt/app/etc/show-and-tell.pb \
--vocab_file /opt/app/etc/word_counts.txt \
--input_files /opt/app/imgs/*
```



Methodology (Contd...)

CaptionBot

CaptionBot.ai is an online tool developed by Microsoft Cognitive Services.

It is machine learning technology that identifies and captions our photos. The image uploaded on the tool is analyzed at their servers, a caption is generated, and the image deleted from their servers (to ensure privacy).

It is powered three APIs, namely Computer Vision API, Emotion API and Bing Image API. All of these APIs are services provided by Microsoft.

We built a wrapper class in python that sends an image to the Microsoft server, wait for the response. Then we extracted the result from their website, process the text and then store it.

CaptionBot (Contd...)

Computer Vision API

Computer Vision API works on images and can be used to get a lot of information. Some of these tasks are text extraction, add to discoverability of the data etc.

No machine learning expertise is required to work with it. Landmarks can be identified, printed as well as handwritten words can be read, popular brand names recognized, by the API with the help of visual data processing. 10,000 + objects and over 20 languages can be recognized by it.

CaptionBot (Contd...)

Emotion API

The Microsoft Cognitive Services Emotion API returns a box enclosing a face. Along with the box, it can also be used to detect emotions by all the people in the images (marked by the box). It also works well for videos. It basically captures the video after some time gaps and then runs on it like it works on an image.

It is capable of detecting 6 emotions, namely, surprise, happiness, anger, contempt, sadness and disgust. This API can be used in various languages including python or, if required, can be called directly by any terminal.

CaptionBot (Contd...)

Bing Image API

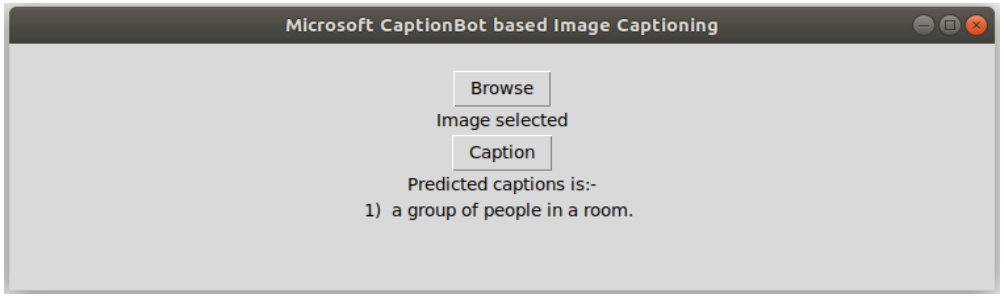
The Bing Image Search API allows you to use Bing's photograph search abilities in your application. Snapshots similar to those available at bing.com/images can be obtained with the help of this API.

The latest version v7 can be used to search for images on the internet. Not only images it works well for image URLs, image metadata, information about the website that has the image.

API v7 also allows us to use factors like brightness, contrast, color scheme to filter images or sort. Queries can be raised to the API about a specific described image and it only returns images that follow this description.

CaptionBot (Contd...)

Caption from Captionbot



Methodology (Contd...)

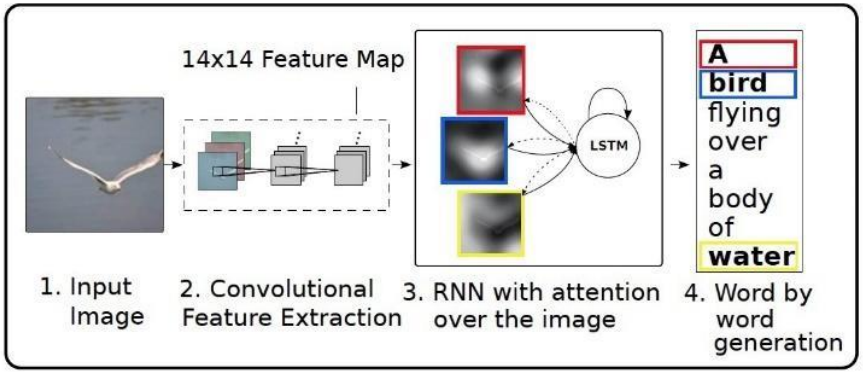
Show, Attend and Tell: Neural Image caption generation with Visual Attention

- Show, Attend and Tell uses the attention mechanism for the generation of image caption, that learns to describe an image.
- The model has to be trained in a deterministic manner by using a technique of standard backpropagation and by maximization of a variational lower bound stochastically.
- There is a CNN-LSTM network for generating image captions. The lower convolutional layers are used for extracting features unlike the previous work which use the final fully connected (FC) layer, thereby, capturing multiple objects inside an image. Thus, image representation is done by different features at different locations.

Show, Attend and Tell (Contd...)
Model Details

The model consists of an encoder (CNN) for extracting image features and a decoder (LSTM) for generating sequence of the caption one word at every time. The approach to caption generation attempts to incorporate two variants of attention mechanism: “soft” and “hard” attention mechanism.

In adjoining figure, in step (2)nd features are captured at lower layers of convolutional network. Then at (3)rd step, feature sampling is done and fed to LSTM, which generates a word correspondingly. This step (3) is repeated K number of times in order to generate a caption of K words.

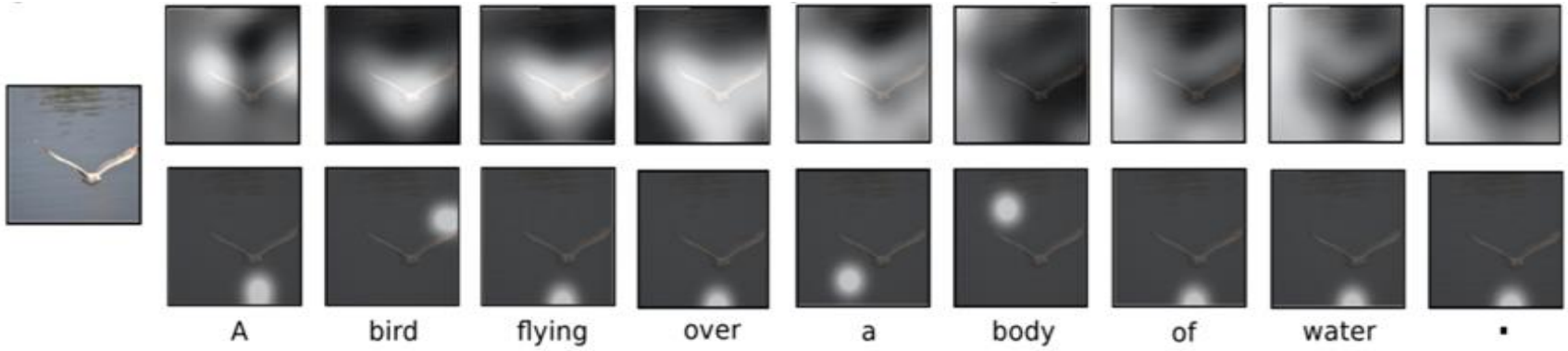


Learning of Model in steps

Show, Attend and Tell (Contd...)

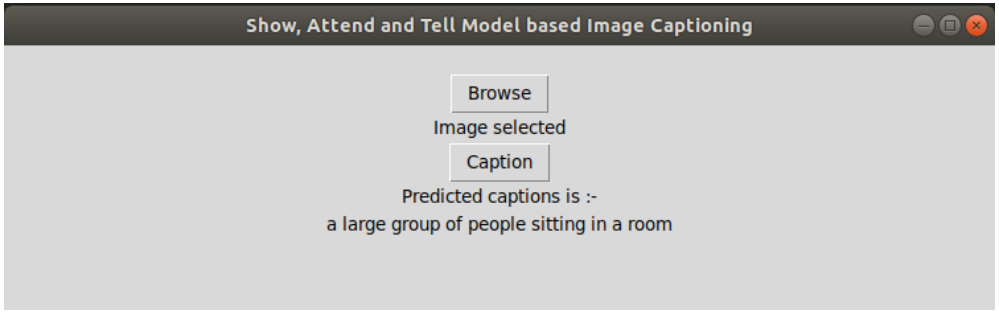
In this figure, as each word is generated, model's attention changes over time and the most relevant part of an image is reflected. Both, soft and hard attention generates same caption in this case.

Model Soft Attention (top row) and Hard Attention (bottom row).



Show, Attend and Tell (Contd...) Implementation

The code for Show, Attend and Tell was available online, which is the python 3 version of original implementation of the paper by its authors and uses the soft deterministic attention mechanism to generate models. We used the pre-trained model as a part of torchvision module of PyTorch, that is trained on MSCOCO dataset.

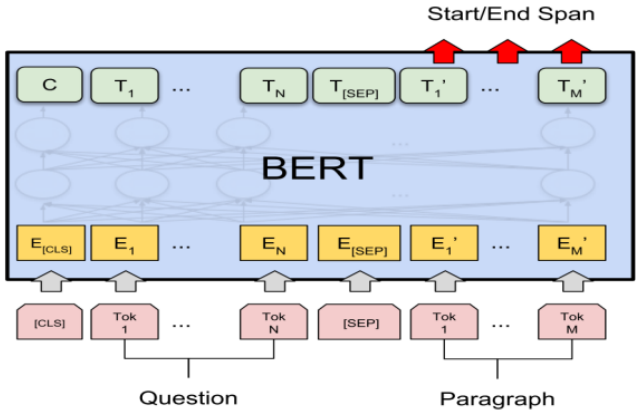


BERT: Bidirectional Encoder Representations from Transformers

- BERT was introduced by researchers in Google AI. It was a state-of-the-art model in the field of NLP.
- BERT uses an attention mechanism, Transformer that learns the contextual relations between words (or sub-words) in a text. Transformers work differently from other directional models as those models read the text input in a sequential fashion (left-to-right or right-to-left), whereas the transformers read the entire sequence at once. Thus, transformers are called Bi-directional although it’s better to call them non-directional.
- This makes BERT unique due to its ability to learn the context of word based on its surrounding words.

BERT (Contd...)

- The BERT can be used to adapt it for Question answering. The PyTorch implementation of BERT from Hugging Face includes that.
- For Question Answering the last hidden layer of BERT is taken and fed into a dense layer and SoftMax to calculate the distributions of the sentence from its start to end.
- Special token probabilities which are added to the head of the input text sequence give its probability of being answerable or not.



BERT model being used for question answering

BERT (Contd...) Implementation

We use the pre-trained model of BERT and then fine-tune it for the question answering using the training scripts provided by Hugging Face. This example code below is used to fine-tune BERT on the SQuAD dataset using distributed training and Bert Whole Word Masking uncased model to reach an F1 > 93 on SQuAD.

```
python -m torch.distributed.launch --nproc_per_node=8 ./examples/run_squad.py \
--model_type bert \
--model_name_or_path bert-large-uncased-whole-word-masking \
--do_train \
--do_eval \
--do_lower_case \
--train_file $SQUAD_DIR/train-v1.1.json \
--predict_file $SQUAD_DIR/dev-v1.1.json \
--learning_rate 3e-5 \
--num_train_epochs 2 \
--max_seq_length 384 \
--doc_stride 128 \
--output_dir ../models/wwm_uncased_finetuned_squad/ \
--per_gpu_eval_batch_size=3   \
--per_gpu_train_batch_size=3   \
```

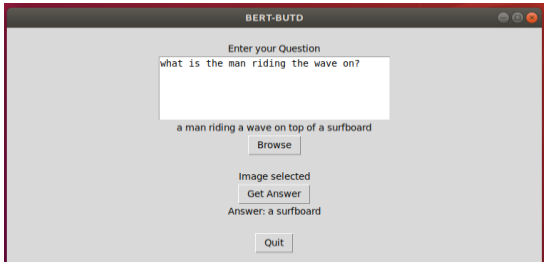
Training with these hyper-parameters as shown in gave us the following results:

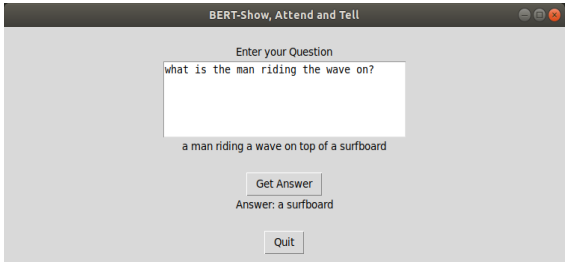
```
{"exact_match": 86.91579943235573, "f1": 93.1532499815869}
```

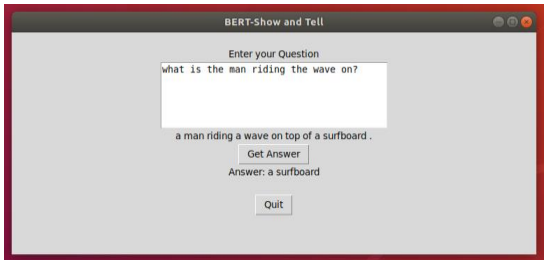
A Python class script is further used to call the saved model and run inference and answer new questions.

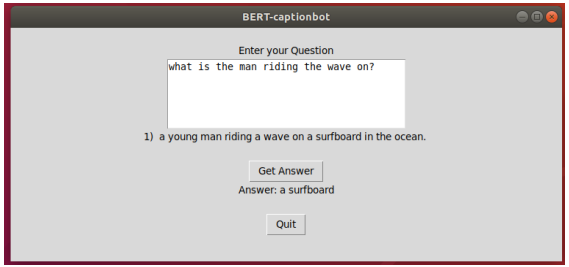
```
(venv) abc@abc-OptiPlex-7040:~/bert_implement$ python
Python 3.6.8 (default, Oct  7 2019, 12:59:55)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from bert import QA
>>> model=QA('model')
>>> q='what is the age of dad?'
>>> con="my dad\'s name is Howard. he is 40 years old"
>>> answer=model.predict(con,q)
>>> print(answer['answer'])\
...
40
```

Observation 1

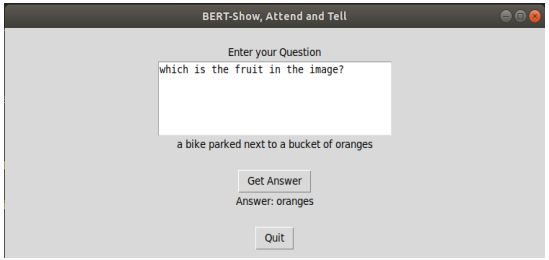


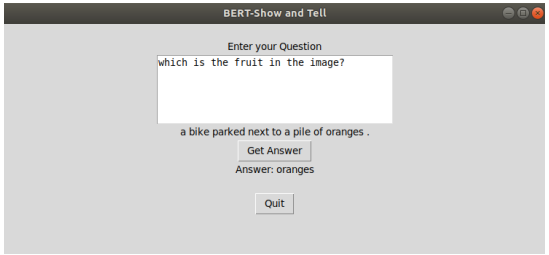


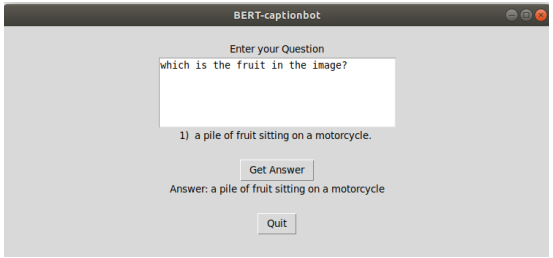




Observation 2







Conclusion

Observation 1: The result of the question “what is the man riding the wave on?” by the BERT+BUTD, BERT+Show and tell, BERT+Captionbot, and BERT+Show, Attend and tell models, all the models provide same answers due to the simplicity of the image. The same trend is followed for most images.

Observation 2: It shows an example where captionbot fails to perform on but other model performs well. One of the major reasons that this can be because the figure has multiple objects, along with shadows (darker regions) making it difficult for captionbot to work well on it. Contrast to this, it works very well on searchable things, for example- famous people.

Conclusion (Contd...)

- If an image containing Albert Einstein is used, the BERT + BUTD, BERT + Show and Tell model and BERT + Show, Attend and Tell model will only caption it as for any other man. On other hand, the caption generated by BERT + Captionbot will caption it with respect to Albert Einstein.
- Opposite to this, if there are different fruits in an image, BERT + Captionbot is more likely to answer it as if those were just fruits while other two models are more likely to answer for individual fruits. The reason for it this is because it uses Bing Search API. It has been trained on images available on Bing Image results. So, it uses that huge dataset to identify more specific details about the image by running reverse image search.
- An advantage of using Show, Attend and tell model is that it is able to generate near perfect captions for an image having lot of different objects, it is able to identify and properly caption those because of its visual attention mechanism.

Conclusion (Contd...)

- Combined models are able to achieve significant progress in terms of answering basic data questions over an image.
- Thus, a combination of two state-of-the-art technologies in different fields trained over different datasets is able to perform very well over tasks like color identification, object classification and basic numeric questions over natural images.
- Captionbot has the added advantage of classifying famous personalities but doesn’t perform very well over complex images.

Further Advancements

- Since both the models have been trained on different datasets retraining of the combined architecture on the VQA/COCO training data may help in increasing the overall accuracy and establish the model as a whole.
- Internal changes to the captioning models to generate more general longer captions to contain more information about the scene depicted in the image can also help to obtain more vivid information about the image; larger the details of the image, larger will be the scope of answering questions accurately and mimic human performance.

Further Advancements (Contd...)

- The input layer of BERT can be changed so as to use the encodings generated by the captioning models directly instead of encodings of the context generated using image caption. This will give features of image as an input instead of concise information about the image.
- Use of attention mechanism inside image captioning can help to generate better captions and better feature maps for image that may directly be used by BERT to answer questions.

Further Advancements (Contd...)

- We can use more advanced models other than BERT.
- GPT-3 by OpenAI is an autoregressive language model with 175 billion parameters, ten times more than any previous non-sparse language model. The model, equipped with few-shot learning capability, can generate human-like text and even write code from minimal text prompts.
    - Being trained on 175 billion parameters, GPT-3 becomes 470 times bigger in size than BERT-Large.
    - While BERT requires an elaborated fine-tuning process where users have to gather data of examples to train the model for specific downstream tasks, GPT-3's text-in and text-out API allows the users to reprogram it using instructions and access it. Case in point — for sentiment analysis or question answering tasks, to use BERT, the users have to train the model on a separate layer on sentence encodings. However, GPT-3 uses a few-shot learning process on the input token to predict the output result.

Further Advancements (Contd...)

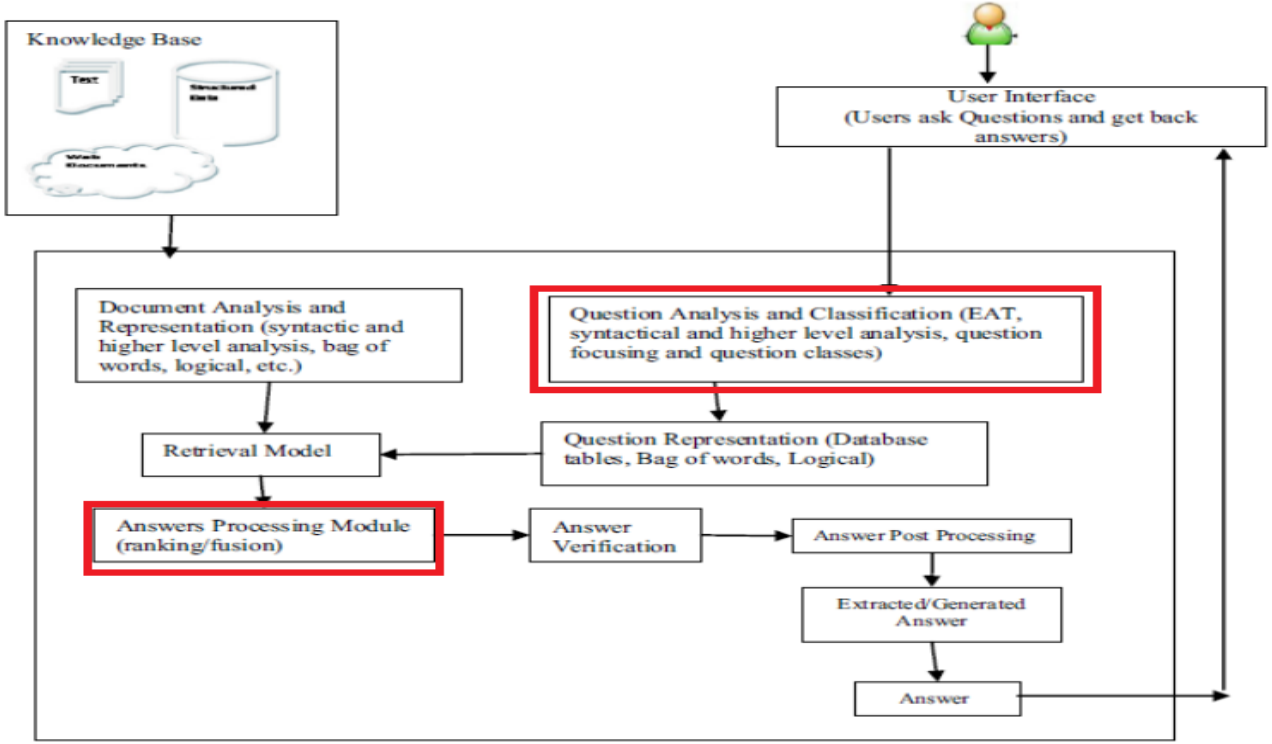
- **XLNet by Carnegie Mellon University**
  - XLNet is a generalised autoregressive pretraining method for learning bidirectional contexts by maximising the expected likelihood over all permutations of the factorization order. XLNet uses Transformer-XL and is good at language tasks involving long context. Due to its autoregressive formulation, the model performs better than BERT on 20 tasks, including sentiment analysis, question answering, document ranking and natural language inference.
- **RoBERTa by Facebook**
  - Developed by Facebook, RoBERTa or a Robustly Optimised BERT Pretraining Approach is an optimised method for pretraining self-supervised NLP systems. The model is built on the language modelling strategy of BERT that allows RoBERTa to predict intentionally hidden sections of text within otherwise unannotated language examples. It also modifies key hyperparameters in BERT, including removing BERT's next-sentence pretraining objective and training with much larger mini-batches and learning rates.
- **DistilBERT by Hugging Face**
  - DistilBERT is a distilled version of BERT. DistilBERT is a general-purpose pre-trained version of BERT, 40% smaller, 60% faster and retains 97% of the language understanding capabilities.

Further Advancements (Contd...)

	BERT	RoBERTa	DistilBERT	XLNet
Size (millions)	**Base**: 110 **Large**: 340	**Base**: 110 **Large**: 340	**Base:** 66	**Base**: ~110 **Large**: ~340
Training Time	**Base**: 8 x V100 x 12 days* **Large:** 64 TPU Chips x 4 days (or 280 x V100 x 1 days*)	**Large:** 1024 x V100 x 1 day; 4-5 times more than BERT.	**Base**: 8 x V100 x 3.5 days; 4 times less than BERT.	**Large:** 512 TPU Chips x 2.5 days; 5 times more than BERT.
Performance	Outperforms state-of-the-art in Oct 2018	2-20% improvement over BERT	3% degradation from BERT	2-15% improvement over BERT
Data	16 GB BERT data (Books Corpus + Wikipedia). 3.3 Billion words.	160 GB (16 GB BERT data + 144 GB additional)	16 GB BERT data. 3.3 Billion words.	**Base**: 16 GB BERT data **Large**: 113 GB (16 GB BERT data + 97 GB additional). 33 Billion words.
Method	BERT (Bidirectional Transformer with MLM and NSP)	BERT without NSP**	BERT Distillation	Bidirectional Transformer with Permutation based modeling

Ongoing project- Design and implementation of Content Based Recommender System for effectively answering Web based User Queries.

Architecture of General QA system (Phases highlighted in red will be our focus)



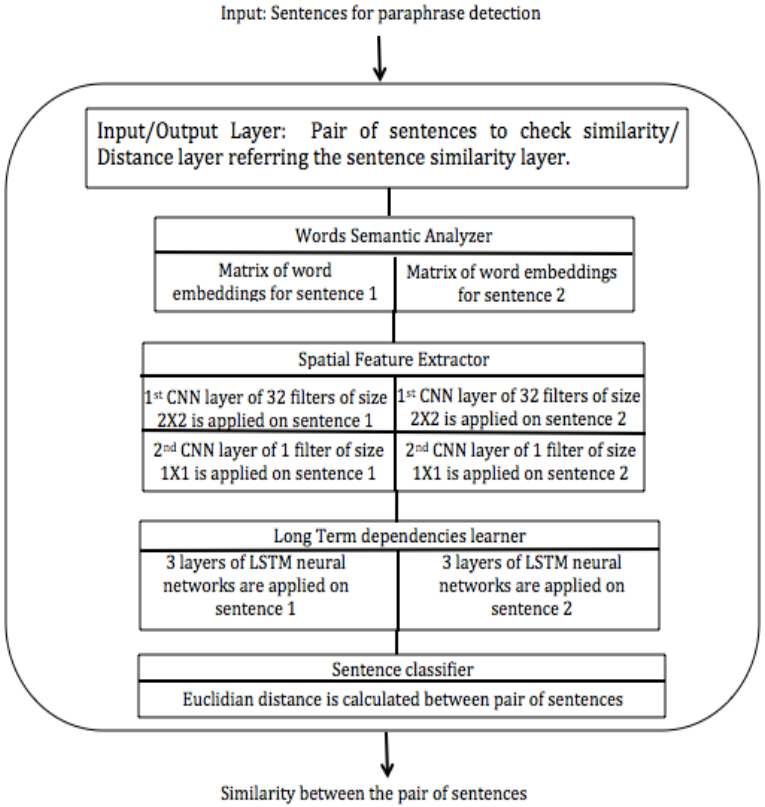
Problem statement:

Given a pair of sentences with binary output 0 or 1 signifying the similarity of the two sentences. Devise a model to predict the similarity of 2 random sentences by learning its semantics

Expected deliverables of this project:

- Working Solution for the proposed Question Answering System.
- Plug-in for Web Browsers as Add-on Extensions.
- Intelligent Algorithms to generate a recommended list of similar questions.
- Multi-Lingual support for Integrated Cyber Physical Systems.

Proposed model



Model Architecture:

- Inputs word indices of fixed length (by zero padding)
- Words look up the embedding layer given by glove 300 dimensional word to vector embeddings
- The 2 matrices are fed to 32 convolution filters with filter size 2x2 and stride one. Then the output from the above layer is passed through 1 filter of size 1x1, again with stride one.
- Since the spatial features are captured by the CNN. To capture the temporal features we use stacked LSTM layers performing back propagation through time.
- The LSTM finally outputs 50 dimensional vector, for each sentence capturing the temporal as well as spatial features. Distance between this 50 dimensional vectors can be use to capture similarity between the two sentences.
- Optimizer used is the Adam optimizer
- One minus normalized euclidean distance is used to output the final similarity of sentences in our project. (Manhattan distance can also be used)

We achieve an accuracy of 81% by training 330635 pairs of sentences and testing on 36738 pairs of sentences.

Actual/Predicted	0	1
0	14609	3852
1	2699	15578

Research Projects undertaken

- **2021-2022: Working on a project titled "Lung cancer detection using deep learning": D**eveloped a hybrid deep learning classifier for lung cancer detection the initial idea of which got published in a Springer International Conference in 2019 and currently **has around 62 citations**.
- **2019-2022: Working on a project titled "Design and implementation of content based recommender system for effectively answering web based user queries":** This project uses advanced nature inspired algorithms and deep learning for dynamic query analysis and effective response. (Budget= Rs. 40 lakhs)
- **2019-2022: Working on project titled "Design and Implementation of Computational Intelligence Based Recommender System for Crop Selection in Rajasthan, India":** This project will recommend the best crop to grow in a particular location in Rajasthan using nature inspired heuristics and machine learning and then develop a user friendly interface for farmers in the state in Hindi. (Budget = Rs 79.84 lakhs)
- **2018-2020: Developed a novel optimization algorithm: I h**ad developed and designed a new optimization technique based on the concepts of plate tectonics, **the plate tectonics based optimization**, the first of its kind which got published in the **Knowledge based systems Journal, Elsevier Publications**, 2021.
- **2016-2018: Contributed to developing extended model of original BBO algorithm: I have** designed an extended model of the original BBO technique proposed by Dan Simon based on the concepts of extinction & evolution and hence dynamic fitness function, communicated the work to **Elsevier Journal of Information Sciences (revision in process, 2021).**
- **2016-2017: Completed a project titled "Food sense" for developing a framework for safety of food:** This project was carried out in association with CEERI, Pilani and worked on image processing using nature inspired algorithms and developed hybrid of intelligent algorithms to test the quality of food also incorporating the odour based factors in the analysis. (Budget= Rs. 2.0 lakhs)
- **2015-2016: Developed hybrid variants of various nature inspired techniques for different image processing applications on normal and satellite multi-spectral images both:** Working at BITS Pilani, with a team of project students at UG and PG level, she has developed hybrid variants of Biogeography based optimization (BBO), Gravitational Search Algorithm, Big-bang crunchy algorithm, Moth Flame Optimization, etc. with other recent optimization techniques and demonstrated much better performance on real world applications like face recognition, character recognition, terrain feature extraction, etc.
- **2011-2015: Working military application of hybrid ACO-PSO-Extended BBO: I have** designed a Two-phase Recommender System for Intelligent Battlefield Preparation for very critical military operations and is **in the stage of practical implementation** by scientists working at DRDO and also has been used and enhanced by several Ph.D. scholars researching at DRDO. (Published in **Knowledge Based Systems, Elsevier Publications, 2015**)
- **2011-2013: Developed a Hybrid Bio-Inspired pattern based intelligent classifier** for land cover feature extraction in Remote Sensing applications which got accepted and published in the Journal of Applied Soft Computing, Elsevier Publications. **(Impact Factor 6.7).** ***This paper has got 42 citations as per current statistics.***

Thank you.