Visual Question Answering System

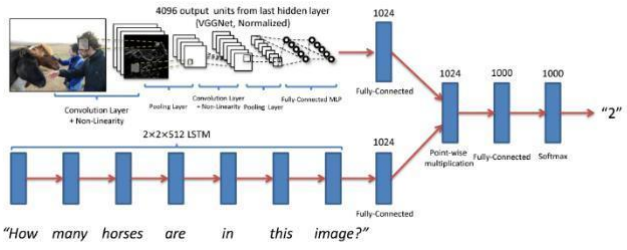
OVERVIEW

A Visual Question Answering (VQA) system is a sophisticated artificial intelligence model designed to comprehend and respond to questions related to visual content. This integration of computer vision and natural language processing enables machines to interpret and generate meaningful responses based on both visual and textual inputs.

The proposed model as described in the paper poses a problem as a K-class classification challenge, utilizing diverse embeddings for image and text inputs. However, to reduce the complexity, I have restricted the problem statement to single-word answers where k is 100.

The study evaluates various models employing pretrained VGG-16, ResNet, Glove-based embeddings, and BERT (Bidirectional Encoder Representation from Transformers) based embeddings. Acknowledging that multiple questions may be posed for a single image, the system accommodates this variability. The architecture allows the flexibility of addressing a multitude of inquiries, contributing to the robustness of the Vanilla VQA system. In this project, I have experimented with different ways of generating image embeddings (using pretrained VGG16, pretrained ResNet50, EfficientNet50 and Custom Densenet models). Furthermore, for text embeddings, I have used the LSTM (Long Short-Term Memory) and BERT models with custom tokenizer for the text.

Inputs:



The inputs typically consist of both visual and textual data. The integration of these modalities allows the system to comprehend questions about visual content.

The first pillar of input is the image itself. Leveraging the MS COCO dataset that consists of 82,783 images, the system encounters a diverse array of visual content. Each image serves as a canvas for potential questions, with annotators contributing some queries per image.

Figure 1:

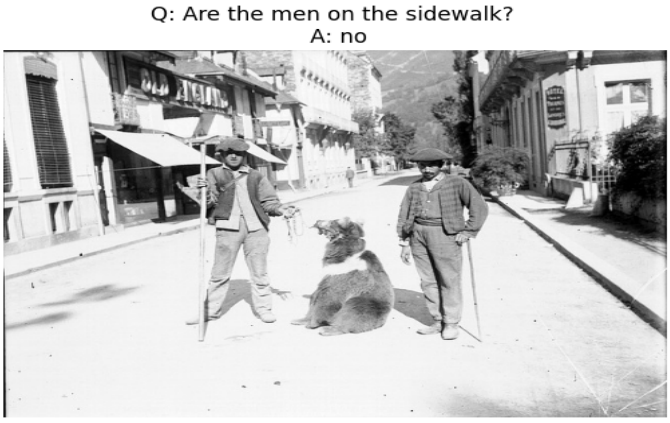


Figure 2:

The above two examples are taken from the dataset. From Fig (1) and Fig (2), it is evident that the questions are open-ended. Answers to these questions can be a yes/no, numbers or a general answer like Figure 2.

Outputs:

Outputs consist of textual answers (can be multiple answers).

Output 1:	Output 2:
No	Skiing

State-of-the-Art:

The state-of-the-art in VQA is often characterized by the use of deep learning models, especially multimodal models that can effectively process both image and text data.

1. The foundation of Vanilla VQA is laid, commencing with an understanding of the Vanilla VQA baseline as the initial framework. This involves framing the problem as a K-class classification, thereby establishing a robust starting point for exploration. Insights drawn from contemporary research papers, guide the narrative by providing context for the recent advancements within the field.
2. Multimodal architecture: State-of-the-art VQA models often use multimodal architectures that combine Convolutional Neural Networks (CNNs) for image processing and recurrent/transformer-based models for text processing. These models can handle the complexity of both visual and textual information.

The most commonly used architecture consists of LSTM layers followed by an FC (Fully Connected) layer to get the text embeddings.On the other hand, for image embeddings pre-trained models like

- VGG16 and RESNET are used.

3. Attention Mechanism: This project incorporates attention mechanisms, such as those found in transformers and these have been applied to VQA models to allow the model to focus on relevant parts of the image and question when generating answers. This helps improve the interpretability of the model while capturing important semantic information.

APPROACH

1. Based on Exploratory Data Analysis, it was found that most of the answers are one-word (75%), therefore restricting the problem to a one-word answer, making it a K-class classification task. Moreover, due to the dataset being too large and containing resource constraints, the training dataset is further randomly sampled to generate 50000 data points for training and 20000 for validation.

2. Models Trained:

i). Using pretrained VGG16 to generate image embeddings and using LSTM to generate text embeddings on input questions. Later, using the concatenation of both embeddings using dot-wise multiplication to generate the final embeddings. The final embeddings are passed through the FC layer with the last layer as sigmoid to get K-class output.

ii). Using pretrained ResNet to generate image embeddings and using the BERT model to generate text

embeddings on the input questions. Later, using the concatenation of both embeddings using the dot-wise multiplication to generate the final embeddings. The final embeddings are passed through the FC layer with the last layer as a sigmoid to get a K-class output.

iii) Using VGG16 output followed by 2FC layers and Conv2d to generate image embeddings and using the BERT model to generate text embeddings on input questions. Later, using the concatenation of both the embeddings using dot-wise multiplication with more weights given to text embeddings where weight is learnt during training to generate the final concatenated embeddings. The final embeddings are passed through the FC layer with the last layer as sigmoid to get a K-class output.

iv) Using the pretrained EfficientNetB7 to generate image embeddings and using the BERT model to generate text embeddings on input questions. Later, using the concatenation of both embeddings using dot-wise multiplication to generate the final embeddings. The final embeddings are passed through the FC layer with the last layer as a sigmoid to get a K-class output.

v). Using a custom built DenseNet model to generate image embeddings and using the BERT model to generate text embeddings on input questions. Later, using the concatenation of both embeddings using the dot-wise multiplication to generate the final embeddings. The final embeddings are passed through the FC layer with the last layer as a sigmoid to get a K-class output.

Reference for first two models used: [link]

The last three models were coded on their own. Based on the analysis, the answers could be predicted solely based on NLP (Natural Language Processing) questions. Therefore, I decided to give more weightage to text embeddings, and made the weight learnable for model 3.
For Model 5, initial experimentation was done using multiple transition and dense blocks which led to enormous trainable params. This reduced the complexity by using less blocks and decreasing the number of filters. However, DenseNet for image embeddings showed positive results when compared to other models.

EXPERIMENTAL PROTOCOL

This project requires training on a multimodal dataset consisting of images and text (questions and human annotated answers). The dataset can be found at [link].Though the original paper is trained on the entire dataset, due to resource constraints, the scope of the

project is limited to 1/3rd of the training dataset. The resulting data is then split into a training and a validation set.

Based on the EDA of the answers, it is observed that most of the questions have one-word answers. Therefore

the analysis on the questions is restricted to one-word answers only. The dataset is further divided into a train and a test set. The training set is used to optimize the model parameters, and the testing set serves as an independent evaluation to measure the final model's performance. The rimary metric used in the paper is custom accuracy defined by the equation below.

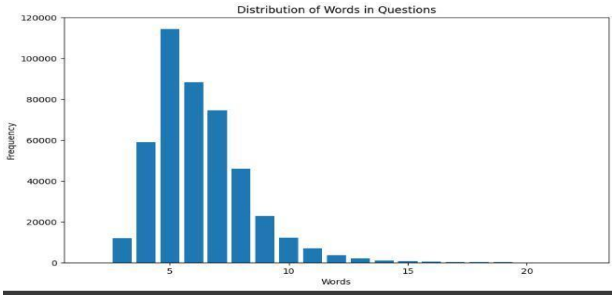
$$\text{accuracy} = \min\left(\frac{\#\text{ humans that provided that answer}}{3}, 1\right)$$

Since the problem is posed as a K-class classification problem, 'accuracy' is used as a performance metric to evaluate the model's performance.

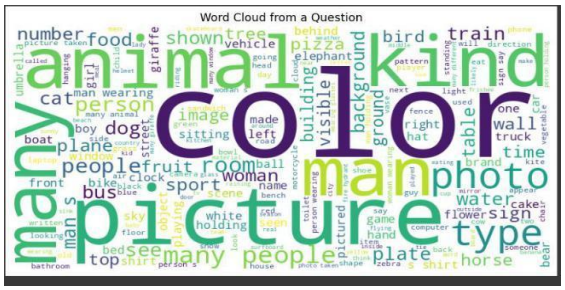
The implementation of deep learning models required significant computing resources, particularly for training complex architectures like BERT and transfer learning with VGG16. I utilized Ggoogle Colab T4 GPU for model training. GPUs significantly expedites the training time.

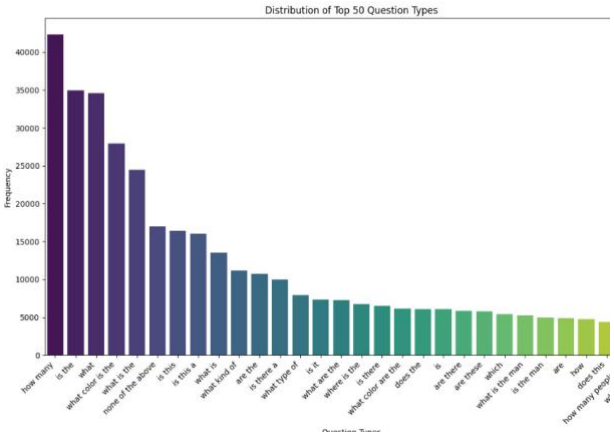
RESULTS

Based on the EDA performed done on the input images, text questions and answers :

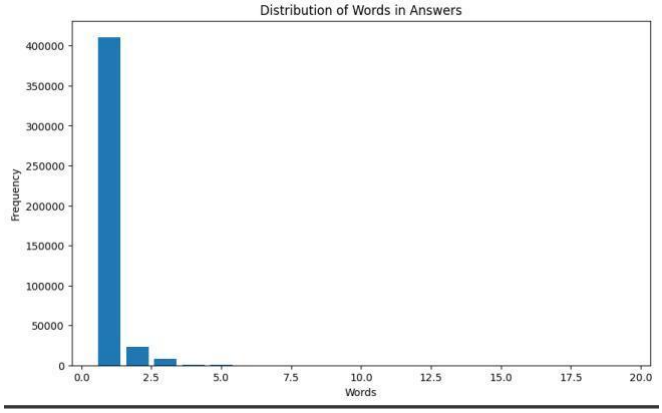


It can be seen that most of the words are in the range of 5 to 10.

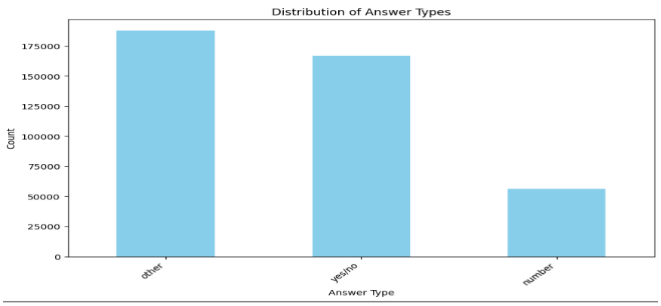


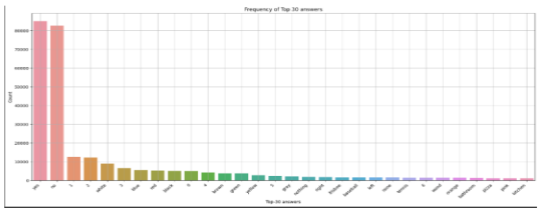


Answers:

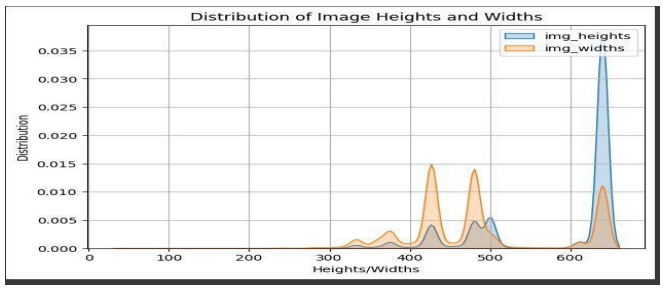


Most of the words in the answers are less than 3.
So I restrict it to one word answers.



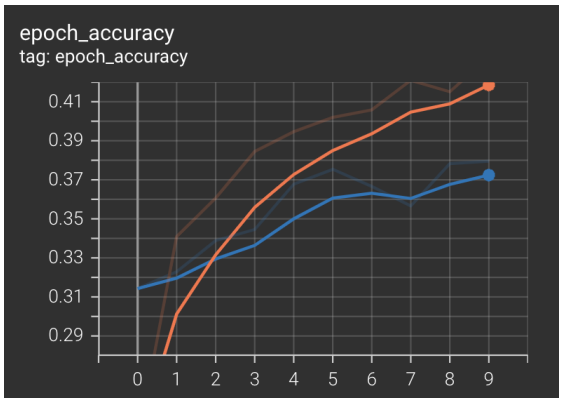


Images:

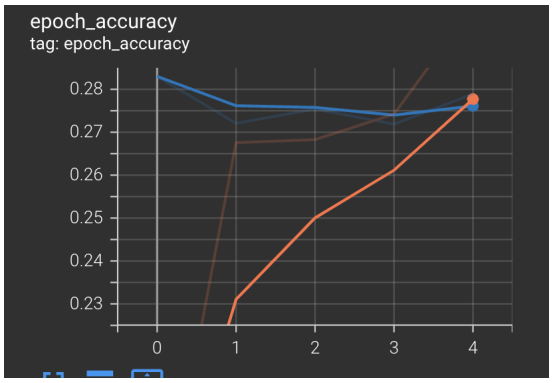


- Also, based on EDA all the images have three channels (R,G,B).

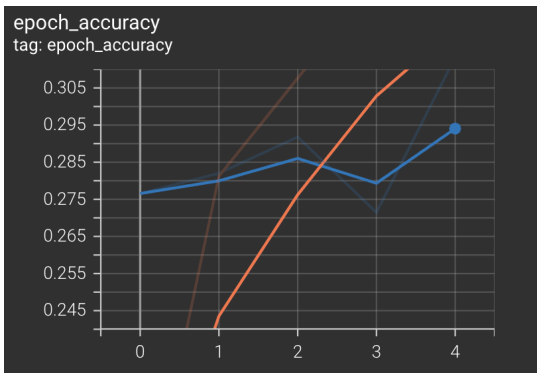
Model 1:



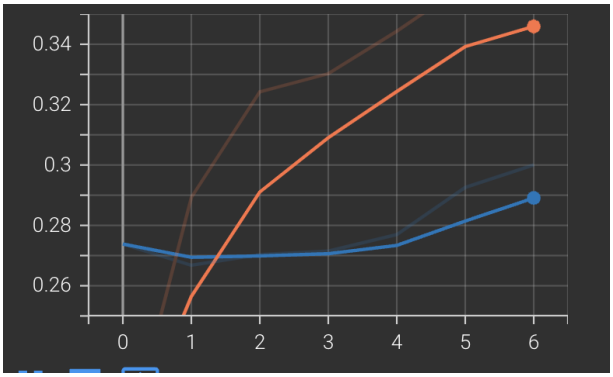
Model 2:



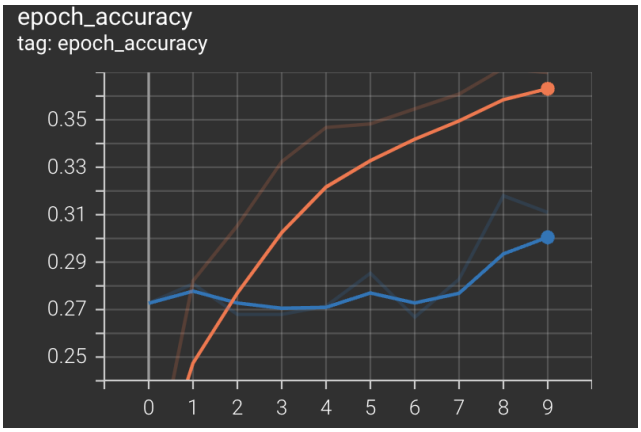
Model 3:



Model 4:



Model 5:



Analysis

The models from the original paper were trained on the entire dataset. Therefore, it is not possible to compare the models built here with that of the state-of-the-art. This is because all these deep learning models work very well when trained on larger datasets.

Pretrained models like VGG16 , ResNet50 took less time to run. For EfficientNetB7 and Densenet, I reduced the model complexity as it was taking longer time to run even with a powerful T4 GPU. Moreover, Densenet was custom implemented with a simpler and less complex architecture to suit the need of the problem. However, with a more complex Densenet architecture, one can expect the model's performance to improve with an increased number of epochs.

Training and validation accuracies were computed for a lesser number of epochs. However, if the number of epochs is increased, then the training and validation accuracies would improve. Moreover, the dataset can be expanded to train the best model that is selected out of the five models and its performance can be checked.

Discussion and Lessons Learnt

- Generating efficient text embeddings is important as compared to image embeddings.
- Transformer based models work really well with large datasets.
- Pre-trained and transformer models need larger datasets.
- Fine-tuning pre-trained models is important.

Future work :
- Use entire dataset for model training and increase the model complexity for DenseNet model.
- Vision transformer can be used to generate image embeddings and BERT for text embeddings, both being trained on the entire dataset.
- State-of-the-art models such as SAN (Stacked Attention Models) can be used since they are better suited for VQA tasks.

BIBLIOGRAPHY

Research papers:

1. https://arxiv.org/pdf/1909.01860.pdf
   https://arxiv.org/pdf/2010.08189.pdf

Github resources:

1. https://github.com/Rak5hith-S/Visual_Question_Answering
2. https://github.com/paarthneekhara/neural-vqa-tensorflow
3. https://towardsdatascience.com/deep-learning-and-visual