Study Oriented Project (CS F266) Multilingual Text Visual Question Answering System

Akshay Shukla (2022A7PS0087P), *Student, BITS Pilani – Pilani Campus*

***Abstract*—Text Visual Question Answering (Text-VQA) systems integrate visual elements and embedded text in images to answer questions. Models like PaLI, TAP, M4C, and SSBaseline employ multimodal attention mechanism to merge modalities to achieve accurate results. These models can be used to aid the visually impaired in daily tasks and basic navigation. Scaling neural networks, as seen in T5, GPT-3, and ViT, enhances performance in tasks like image captioning and VQA. The PaLI model, introduced by Google Research in 2022, excels in image-only, language-only, and combined (Image and Language) tasks across multiple languages. This project aims to enhance the PaLI model, focusing on Indian languages by either diversifying the dataset used or fine-tuning the model. Multilingual features will complement OCR, NLP, and the decoder functions, with enhancements like 2D-Attention and variational encoder for improved accuracy.**

I. Introduction

Text Visual Question Answering (Text - VQA) models aim to answer questions based on an image given as input by extracting information from the visual elements as well as the embedded text in the images. They try to effectively use the individual and multimodal attention mechanism for joining all the modalities in a space to provide the exact answer. There are several models available which deal with VQA. These include PaLI, TAP, M4C and SSBaseline to name a few. Direct applications of such models include helping the visually impaired by answering questions on images in daily life as well as helping them in navigation from one location to another.

Making neural networks larger has proven beneficial for tasks involving language and vision modeling. Well-known models like T5, GPT-3, and ViT have demonstrated clear advantages from scaling, resulting in better performance in tasks such as image captioning and visual question-answering. One of the most recently proposed models is the PaLI model introduced by Google Research in 2022. PaLI was introduced as a model that handles image-only, language-only, and combined tasks across multiple languages through a unified interface. It achieves a balanced parameter distribution between language and vision components, leveraging existing large models like mT5-XXL and ViT-e for efficient training and transfer learning. PaLI-17B, trained on the WebLI dataset with billions of image-text pairs in over 100 languages, achieves state-of-the-art results on benchmarks like COCO Captioning and VQAv2. PaLI has a test time accuracy of around 73% on most popular datasets.

The objective of this project is to try to improve on the capabilities of the PaLI model while specifically focusing on Indian languages. This implies that we either diversify the WebLI dataset (10 billion images from 109 languages) to include more Indian languages or we fine-tune the existing model to be compatible with Indian languages. The multilingual functionalities should be included in a way that it supplements the OCR, NLP and decoder functions. Further we also attempt to improve the accuracy of the PaLI model while working on this expanded dataset. This can be done by implementation of the 2D-Attention and variational encoder for the already proposed Text-VQA model.

II. Related Work

The existing baseline models for Text Visual Question Answering are LoRRA, M4C and SSBaseline.

- LoRRA (27.63% accuracy) - A basic model using self and contextual attention over visual and textual features along with a module to copy OCR tokens directly to the answers.
- M4C (40.46% accuracy) - Different modalities taken through transformer embeddings and then a joint embedding is created and sent to a single-multimodal transformer.
- SSBaseline (45.66% accuracy) - OCR and Visual features are handled separately by vanilla attention blocks and then served to a transformer.

Of these, the model which was heavily focused on in the past is SSBaseline. This model has several features included within it. It uses the BERT model (designed by Google Research) for the Natural Language Processing (NLP) task. This provides us with an entry point to pose a question to the model. The ResNet-152 model is used for reading the image provided to the model. All questions asked to the model will be answered with the help of this image. Faster R-CNN is used for Optical Character Recognition (OCR) from the visual images given to the model. This model is trained over the VQA 2.0 (https://visualqa.org/download.html) and TextVQA

(https://textvqa.org/dataset/) datasets. The TextVQA dataset contains 45336 questions posed over 28408 images. A Transformer based neural network with self-attention mechanism is used for encoding the output vectors obtained by a combination of the above mentioned models. Additionally, Long Short-Term Memory (LSTM) is used for decoding the vector and generating the output sequence of characters. This model possessed the desired Multilingual Text-VQA functionality but did not have a sufficient number of Indian languages included in its dataset nor did it have a decently satisfying test-time accuracy. The SSBaseline dataset is tricky to work with and the OCR tokens had a defective reading ability. All of this added up forced us to look for an alternative model.

III. Proposed Techniques and Algorithms

We looked at various multimodal models and their overall success rates and found that the PaLI model is the most successful model.

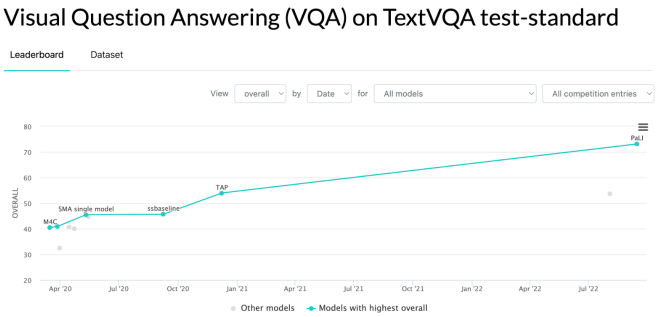


Figure 1 graph showing the comparison of accuracies of leading models

Rank	Model	overall	Extra Training Data	Paper	Code	Result	Year	Tags
1	PaLI	73.1	×	PaLI: A Jointly-Scaled Multilingual Language-Image Model			2022	
2	TAP	53.97	×				2020	
3	TAG	53.69	×	TAG: Boosting Text-VQA via Text-aware Visual Question-answer Generation			2022	
4	ssbaseline	45.66	×				2020	
5	SMA single model	45.51	×				2020	
6	SAM (Single Model)	44.8	×				2020	
7	colab_buaa	44.73	×				2020	
8	CRN (Single Model)	40.96	×				2020	
9	CIG	40.77	×				2020	
10	M4C	40.46	✓				2020	
11	Shuai	39.95	×				2020	
12	mmgnn	32.46	×				2020	

Figure 2 PaLI model accuracy compared with other leading models with time of release

With the goal of answering questions based on an image, PaLI aims to perform unimodal (language, vision) and multimodal (language and vision) tasks with the help of a simple and scalable main architecture. It uses an encoder-decoder transformer model, with a large-capacity ViT component for image processing. For the text encoder-decoder, it uses pre-trained mT5 models, while for the image encoder, it reuses large vanilla ViT models.

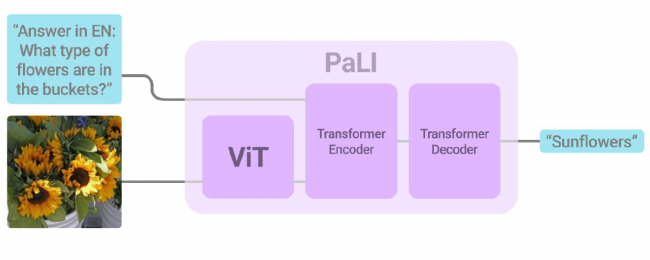


Figure 3 above shows a high-level schematic of the model architecture.

At its core, PaLI has a text encoder decoder Transformer. To include vision as input, the text encoder is fed with a sequence of visual 'tokens': output patch features of a Vision Transformer which takes as input an image. It uses previously trained unimodal checkpoints.

The visual component uses the largest vanilla ViT architecture to date, named ViT-e. ViT-e has the same architecture and uses the same training recipe as the 1.8B parameter ViT-G model, while scaling to 4B parameters. The only other difference is that it applies learning rate cool-down twice, once with and once without inception crop augmentation, and averages the weights of the two models.

The language component uses the mT5 backbone as the language component. Pre-trained mT5-Large (1B parameters) and mT5-XXL (13B parameters) were experimented and the language encoder-decoder of PaLI was initialized.

Overall, the model includes three model sizes:

1. PaLI-3B, where the language component is initialized from mT5-Large (1B parameters), and the vision component is ViT-G (1.8B parameters).
2. PaLI-15B, where the language component is initialized from mT5-XXL (13B parameters), and the vision component is ViT-G (1.8B parameters).
3. PaLI-17B, where the language model is initialized from mT5-XXL, and the vision component is the newly-trained ViT-e model (4B parameters).

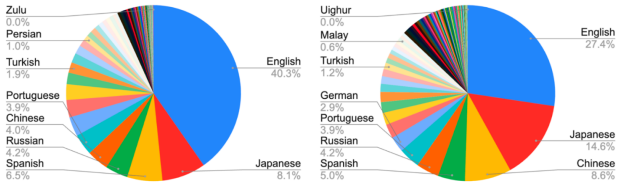
IV. Dataset Used

Scaling studies for deep learning show that larger models require larger datasets to train effectively. In order to effectively utilize a model like PaLI-17B, we require a large dataset. To unlock the potential of multilingual image-language pre-training, PaLI uses WebLI, a multilingual image-language dataset built from images and texts available on the public web.

Sampled images from WebLI associated with multilingual alt-text and OCR. The second image is by jopradier (original), used under the CC BY-NC-SA 2.0 license. Remaining images are also used with permission.

Figure 4 Sample images from WebLI with multilingual alt-text and OCR

WebLI scales up the image language data collection from English-only datasets to 109 languages, which enables us to pre-train PaLI multilingually, and perform downstream tasks across many languages.



Statistics of recognized languages from alt-text and OCR in WebLI.

Figure 5 Coverage of various languages by WebLI dataset

Due to the abundance of multilingual content on the internet, the collection process for the WebLI dataset can be scaled to cover 10 billion images and 12 billion alt-texts. In addition to annotation with web text, publicly available automatic service is used to extract OCR annotations on all images, resulting in 29 billion image-OCR pairs. To balance quality and retain scale, the dataset was filtered to the highest quality subset retaining only the top 10% scoring of the original WebLI image-text pairs (about 1B examples), which were finally used to train PaLI.

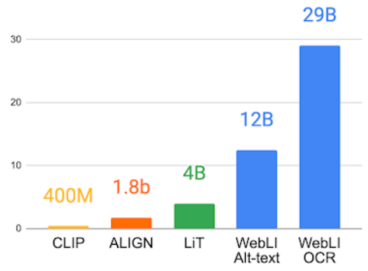


Image-text pair counts of WebLI and other large-scale vision-language datasets, CLIP, ALIGN and LiT.

Figure 6 Comparison of various datasets

We also made use of the Imagenette dataset. Imagenette is a subset of 10 easily classified classes from the Imagenet dataset. It was originally prepared by Jeremy Howard of FastAI. The objective behind putting together a small version of the Imagenet dataset was mainly because running new ideas/algorithms/experiments on the whole Imagenet take a lot of time.

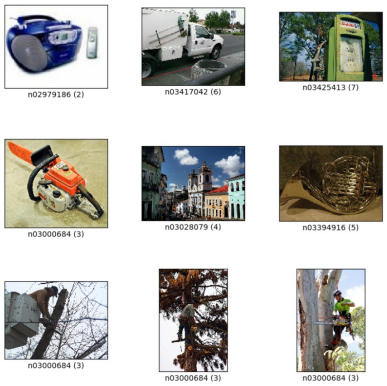


Figure 7 Subset of Imagenette dataset

In addition, we also made use of the CIFAR100 dataset. This dataset is just like the CIFAR-10, except it has 100 classes containing 600 images each. There are 500 training images and 100 testing images per class. The 100 classes in the CIFAR-100 are grouped into 20 superclasses. Each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs).

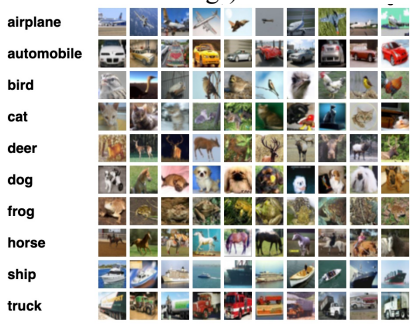


Figure 8 Subset of CIFAR100 dataset

V. Experiments and Results

Using the code available on the big_vision GitHub page (https://github.com/google-research/big_vision/tree/main), we were able to enter the training phase of the PaLI model while using the ImageNet 2012 dataset. This is primarily a unimodal image to text training. The goal is to perform this training on image to text, text to text and then multimodal image+text to text.

The hardware available with us is a single Nvidia A100-PCIE-40GB GPU machine.

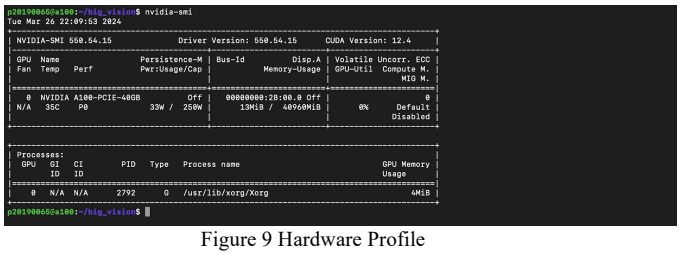


Figure 9 Hardware Profile

It took some time and availability of SUDO access for us to get the Nvidia CUDA drivers, CUDA Toolkit, CuDNN and TensorRT working. These were essential for the training of the model on the GPU.

We had significant challenges with respect to memory limitations. The original batch size was 1024 which was too large for the machine to handle due to RAM limitations and hence the training process killed before a single step of training took place. We changed the batch size in order to understand the effect it would have on the training process.

We were eventually able to run with a batch size of 4. This is obviously very small as compared to the original batch size of 1024. With a batch size of 4, the model would take 546 days to train which is an unreasonably long time to wait for. These are the results obtained after 950 steps out of a total of 28,537,988 steps.

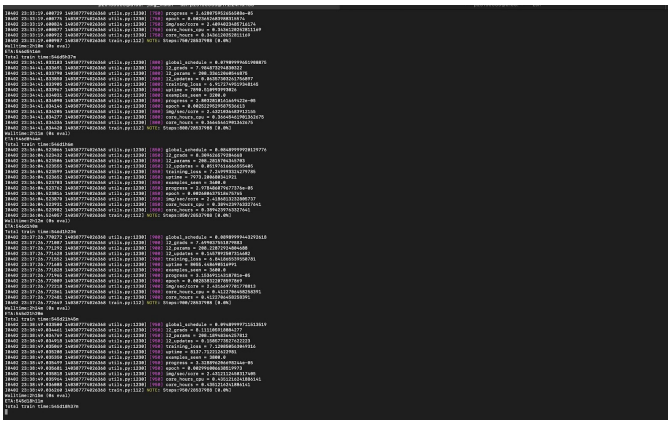


Figure 10 Results of training PaLI with a batch size of 2

Given the size of the dataset and the number of parameters, we believe we will need a larger GPU cluster.

Since training the WebLI dataset on the limited hardware capacity available to us was not feasible, we decided to go ahead with transfer learning. We tried CapPa (autoregressive captioning without parallel prediction), CLIPPO (CLIP with pixels only), Image/Text model (LiT: Zero-Shot Transfer with Locked-image text Tuning). We decided to use the Imagenette dataset for this purpose. Imagenette is a subset of 10 easily classified classes from the Imagenet dataset. It was originally prepared by Jeremy Howard of FastAI. The objective behind putting together a small version of the Imagenet dataset was mainly because running new ideas/algorithms/experiments on the whole Imagenet take a lot of time.

We were able to successfully run transfer learning but came across with similar issues to what we had while training. We ran out of GPU memory with the default batch size of 1024.We kept reducing the batch size and were finally able to get it running with a batch size of 4. This is sub-optimal because it will take a very long time even for transfer learning.

VI. Conclusion and Future Work

We are still working on the model. Having gotten the Nvidia GPUs working after a lot of hit & trial and a special SUDO access for the purpose, we are now exploring optimizing the training on the available hardware. A more comprehensive description of the same is below in the 'Work Update' section.

VII. Work Update

A. Work Completed

We initially tried to set up the server by using the local installations of the various CUDA drivers and other dependencies required for running the model but were unsuccessful because of several clashes with the system-wide dependency versions. Got the Nvidia CUDA drivers, CUDA Toolkit, CuDNN and TensorRT working on server p20190065@172.24.16.136 after SUDO access was given to us for the server.

We initially tried to get the environment setup for training and running the m4c model on the server. This model is from 2017 and given the pace at which innovation is happening in this field many of the libraries required for the model have become obsolete and incompatible as a result of which we could not get the model running. We then moved over to the PaLI model which is the state of the art model for this purpose.

We had significant hardware challenges. We have an Nvidia A100 GPU with 40GB RAM but training this dataset requires a significantly higher amount of memory and processing power. The original model is configured with a batch size of 1024 but the best we could get running on the given hardware was a batch size of 4.

Since training the WebLI dataset on the limited hardware capacity available to us was not feasible, we decided to go ahead with transfer learning. We tried CapPa (autoregressive captioning without parallel prediction), CLIPPO (CLIP with pixels only), Image/Text model (LiT: Zero-Shot Transfer with Locked-image text Tuning). We finally decided to use the Imagenette and CIFAR100 datasets for this purpose.

We were able to successfully run transfer learning but came across with similar issues to what we had while training. We ran out of GPU memory with the default batch size of 1024.We kept reducing the batch size and were finally able to get it running with a batch size of 4. This is sub-optimal because it will take a very long time even for transfer learning.

B. Work Remaining

Given that we have got the training initiated for the PaLI model with a maximum batch size of 4 and on facing the same issue when trying to fine tune a pre-trained model with transfer learning, we have learnt that computer vision based projects are hardware intensive and running them on a single GPU is not feasible. We will need to explore changing configurations to support a higher batch size. One possibility for this is by running the model on distributed GPUs or a GPU cluster so that we can tap into the processing power of each one to speed up the training process.

Once we succeed in training the model or in transfer learning in a reasonable amount of time we can work towards the next goal of expanding the model to be compatible with Indian languages which can be done by either diversifying the WebLI dataset or by appropriate fine-tuning of the model. Another way is to add a translator at both ends of the model so that the model effectively still has to deal with input only from a language it

has already been trained in.

ACKNOWLEDGMENT

I would like to express my heartfelt gratitude to my esteemed professor, Prof. Yashvardhan Sharma, for his invaluable guidance, unwavering support, and mentorship throughout this research endeavor. I am deeply thankful to the dedicated research scholar, Ms. Vijaykumari, whose expertise and insights have enriched this study. Furthermore, I am grateful to the college, BITS Pilani – Pilani Campus, for providing exceptional facilities and resources that have been instrumental in the progression of this research. Their collective contributions have been pivotal in shaping this work and fostering my academic growth.

REFERENCES

[1] https://github.com/google-research/big_vision/tree/main - Github repository with the code for PaLI model
[2] https://paperswithcode.com/paper/pali-a-jointly-scaled-multilingual-language - Paper by Google Research describing the PaLI model
[3] https://docs.nvidia.com/cuda/cuda-installation-guide-microsoft-windows/ - Cuda Installation guide
[4] https://developer.nvidia.com/cuda-12-0-0-download-archive?target_os=Linux&target_arch=x86_64&Distribution=Ubuntu&target_version=22.04&target_type=deb_network - used for downloading CUDA Toolkit 12.0 required for the server
[5] https://docs.nvidia.com/deeplearning/cudnn/installation/linux.html Guide for installing CUDA Deep Neural Network (CUDNN)
[6] https://docs.nvidia.com/deeplearning/tensorrt/install-guide/ NVIDIA Deep Learning TensorRT installation guide
[7] https://paperswithcode.com/sota/visual-question-answering-on-textvqa-test-1 Comparision of various models used for Text VQA
[8] https://github.com/ZephyrZhuQi/ssbaseline Github repository with the code for the SSBaseline model
[9] 2012.05153.pdf (arxiv.org) Research paper by Meta on the SSBaseline model

Akshay Shukla (B.E Computer Science, Second Year, BITS Pilani – Pilani Campus) was brought up in Bangalore and completed his higher secondary school certification from Sri Kumaran Children's Home Educational Council (CBSE) located in Bangalore, Karnataka, India.

Akshay Shukla is currently a member of the Postman API and Innovation Lab, Kalipatnapu AR/VR Lab, Wall Street Club and Coding Club all located in BITS Pilani, Pilani Campus. He is currently doing a project under the guidance of Prof. Yashvardhan Sharma (BITS Pilani). He is also a national level basketball player and has represented Karnataka in various tournaments and is currently a member of the BITS Pilani college team.