Automatic image captioning

Capstone Project

## 

# GROUP 13

# TEAM

## Hari Krishna Suguru

## Manohar Rao Pagadala

## Keerthi Pakala

# Mentor

## Nayan Anand

Contents

[Abstract: 4](#_Toc113425233)

[Problem Statement 4](#_Toc113425234)

[Objective: 4](#_Toc113425235)

[Dataset 4](#_Toc113425236)

[Tools: 4](#_Toc113425237)

[Methodologies 5](#_Toc113425238)

[Machine Learning Process flow 5](#_Toc113425239)

[Data collection 5](#_Toc113425240)

[Pre-processing 5](#_Toc113425241)

[Model selection 5](#_Toc113425242)

[Training model 6](#_Toc113425243)

[Evaluation 6](#_Toc113425244)

[Tuning Model 6](#_Toc113425245)

[Prediction 6](#_Toc113425246)

[Deployment 6](#_Toc113425247)

[Metrics 6](#_Toc113425248)

[Model Deployment 7](#_Toc113425249)

[Deployment Architecture: 7](#_Toc113425250)

[Fast API: 7](#_Toc113425251)

[Ngrok: 7](#_Toc113425252)

[Model: 7](#_Toc113425253)

[Web pages: 7](#_Toc113425254)

[Steps to deploy model: 8](#_Toc113425255)

[Version Control 10](#_Toc113425256)

[Literature Review and References 10](#_Toc113425257)

# Abstract:

Captioning the images with proper description is a popular research area of Artificial Intelligence. A good description of an image is often said as “Visualizing a picture in the mind”. The generation of descriptions from the image is a challenging task that can help and have a great impact in various applications such as usage in virtual assistants, image indexing, a recommendation in editing applications, helping visually impaired persons, and several other natural language processing applications. In this project, we need to create a multimodal neural network that involves the concept of Computer Vision and Natural Language Process in recognizing the context of images and describing them in natural languages (English, etc). Deploy the model and evaluate the model on 10 different real-time

# Problem Statement

Captioning the images with proper description is a popular research area of Artificial Intelligence. A good description of an image is often said as “Visualizing a picture in the mind”. The generation of descriptions from the image is a challenging task that can help and have a great impact in various applications

# Objective:

Build an Image captioning model to generate properly formed English sentences, to describe the content of the selected image.

# Dataset

Flickr8k

Flickr30k

MSCOCO

# Tools:

Natural Language Toolkit, TensorFlow,

Keras,

Transformers

GitHub

Google Colab

# Methodologies

* Xception with LSTM
* RESTNET152 with LSTM
* Transformer model

# Machine Learning Process flow

### Data collection

Data collected and experimented with different datasets like 8k, 30k of flicker data sets and CoCo (common object in context) datasets. Data is collected through Kaggle <https://www.kaggle.com/datasets/adityajn105/flickr30k> and Coco <https://cocodataset.org/#home>. some datasets has clear splits like Train images (83K) and test images (41k). this help us to evaluation model performance. This data is also associated test description of image and it will be helpful in preparing LSTM network.

### Pre-processing

Collected data is pre processed and cleaned in several steps. Resized the image to fit convolutional network, descriptions are cleaned and removed punctuational characters and converted to lower case.

### Model selection

Model contains two DL/ML models one for objection detection, segmentation and extracting image features and another one is language model to preparing description.

#### Feature Extraction:

Transfer learning methodology applied to re-use the model weights from pre-trained models. Removed last layer fully connected layers to satisfy our need.

Team is experimented with different models

Xception model

RESNET 152

EfficientNet

#### Language Model (Sequence building Model)

For sequence building applied different methodologies like Long Short Term Memory (LSTM) and Transformer model

##### LSTM

Long Short-Term Memory (LSTM) networks are a type of Recurrent Neural Network (RNN) capable of learning order dependence in sequence prediction problems.

##### Transformer Model

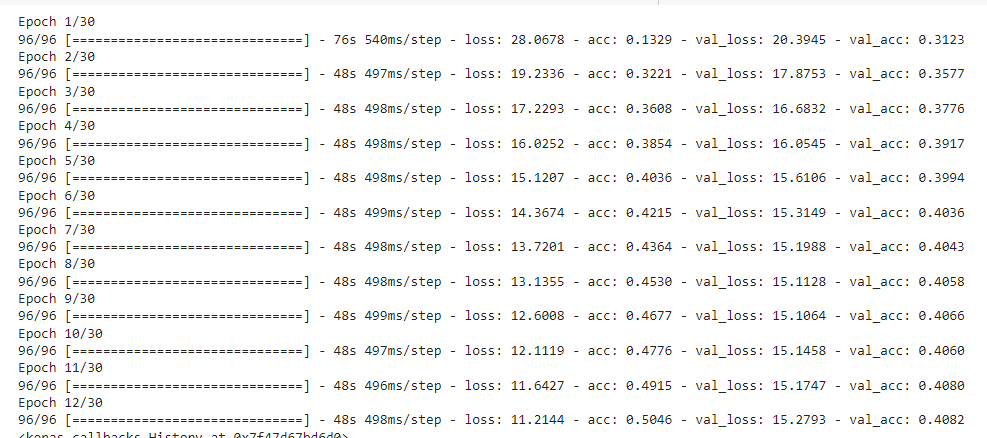
The Transformer in NLP is a **novel architecture** that aims to solve sequence-to-sequence tasks while handling long-range dependencies with ease. It relies entirely on self-attention to compute representations of its input and output **WITHOUT** using sequence-aligned RNNs or convolution.

Attention: *Attention allowed us to focus on parts of our input sequence while we predicted our output sequence*.

### Training model

Train the model for 15 epocs and evaluated for accuracy, also applied validation accuracy to overcome overfitting. Model training is stopped when validation accuracy score is going down.

Also saved the trained model to deploy in production with help *model.save* method in HD5 format.



### Evaluation

Model is evaluated with different metrics, like the Bilingual Evaluation Understudy (BLEU) for quality of text generation, Accuracy and Loss functions. Achieved belu score more than 1.

### Tuning Model

Model trained and tuned with different hyper parameters like batch size, learning rate, number of epocs and etc.

### Prediction

Experimented different images how model is predicting. With trained images and caption is predicting well, however with unknown image is given small discrepancy is observed for forging images.

### Deployment

Model is deployed in FastAPI for production use. In this phase developed REST API user to upload image. Saved model is loaded in FAST API web environment and predicted caption and displayed back to user. Detailed deployment steps are provided below.

# Metrics

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Dataset | Architecture/model | Epoch | Accuracy | Loss | Validation Accuracy | Validation Loss | Bleu score |
| 8K | LSTM | 10 | 0.35 | 2.7868 | NA | NA | 1.288 |
| 30K | LSTM 30k | 10 | 0.29 | 3.9605 | NA | NA | 1.28 |
| 8K | CNN-Transformers | 12 | 0.50 | 11.21 | 0.4002 | 15.5382 | 8.17 |
| 30K | CNN-Transformers 30k | 9 | 0.50 | 11.07 | 0.40 | 15.44 | 8.188 |
| 30K | RESNET150 | 2 | NA | 1.9591 | NA | NA | 7.0931 |

# Model Deployment

Machine learning model can be deployed in webserver for production purpose or wide client access. Following document describe about architecture and steps in deployment. Model can deployed Uvicorn server or google colab for experimental purpose.

## Deployment Architecture:

Server

Browser

Ngrok

<<gateway>>

FastAPI

<<webapi>>

Machine Learning Model

Web pages

<<static>>

### Fast API:

FastAPI is a modern, fast (high-performance), web framework for building APIs with Python 3.6+ based on standard Python.

### Ngrok:

Ngrok is gateway and load balancer, it will open secure web connections with browser and revere proxy to FASTAPI web server. it also provide public url to access website.

### Model:

ML model is trained and tested model and it can be saved in different formats. save\_model

Provide option to saving model.

### Web pages:

These are static html file, scripts and css.

## Steps to deploy model:

* Install ngork

!pip install flask**-**ngrok

!pip install flask**==**0.12.2

* Install DL/ML libarries

!pip install tensorflow

!pip install keras

!pip install pillow

!pip install numpy

!pip install tqdm

!pip install keras**.**utils

* Load google drive

from google.colab import drive

drive**.**mount('/content/drive')

* Import important API

**from** keras.models **import** Model, load\_model

**from** keras.applications.xception **import** Xception, preprocess\_input

* Update nrok token

!**/**ngrok authtoken 2DI4bkOAQIhGOsc75CHmWQFEVuq\_5pFDKAMNaBtXXXXX

* Develop API

# return webpage

@app**.**route("/")

**def** home():

**return** render\_template('index.html')

# load model and prediction

@app**.**route('/getprediction', methods**=**['POST'])

**def** getprediction():

tokenizer **=** load(open('/content/drive/MyDrive/website/model/tokenizer.p',"rb"))

model **=** load\_model('/content/drive/MyDrive/website/model/model\_9.h5')

xception\_model **=** Xception(include\_top**=False**, pooling**=**"avg")

photo **=** extract\_features(img\_path, xception\_model)

img **=** Image**.**open(img\_path)

description **=** generate\_desc(model, tokenizer, photo, max\_length)

description **=** description**.**replace("start ", "")

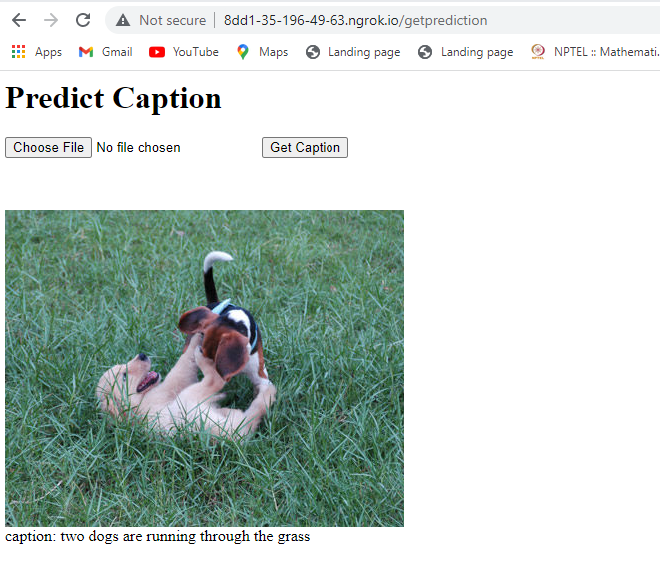
description **=** description**.**replace(" end", "")

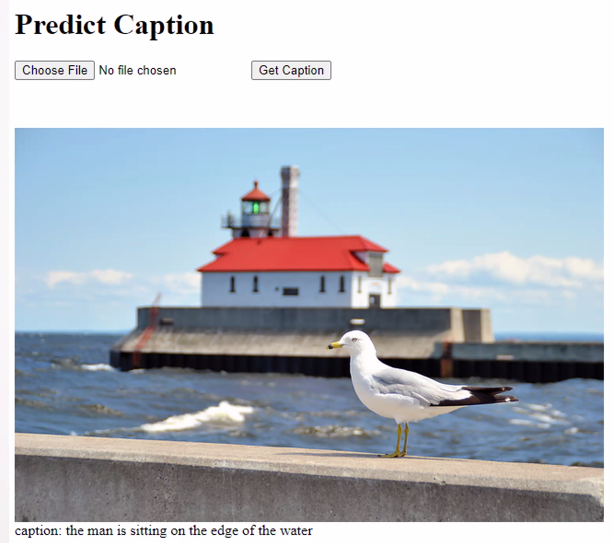
* Public URL

\* Running on <http://95e0-34-148-31-45.ngrok.io>

\* Traffic stats available on http://127.0.0.1:4040

User Interface and output





Challenges

* For 30k training is taking long time and Google colab’s runtime is getting frequently disconnected.
* Initially time taken to understand transformer model
* There is no significant accuracy improvement 8k and 30k, need to explore more on different hyper parameters

# Version Control

Source code is deployed in git repository can be access through following link

<https://github.com/harisuguru06/Automatic-Image-Captioning>

# Literature Review and References

* <https://flask.palletsprojects.com/en/2.2.x/>
* <https://ngrok.com/>
* Coco dataset: <https://github.com/cocodataset/cocoapi>
* [Automatic Image Captioning Based on ResNet50 and LSTM with Soft Attention](https://downloads.hindawi.com/journals/wcmc/2020/8909458.pdf)
* [Automatic Image Captioning using Deep Learning (CNN and LSTM) in PyTorch](https://www.analyticsvidhya.com/blog/2018/04/solving-an-image-captioning-task-using-deep-learning/)
* [Deep Visual-Semantic Alignments for Generating Image Descriptions](https://cs.stanford.edu/people/karpathy/cvpr2015.pdf)
* [Attention Is All You Need Link](https://arxiv.org/pdf/1706.03762.pdf)
* [Automatic Image Caption Generation System](https://ijisrt.com/assets/upload/files/IJISRT21JUN776.pdf)
* <https://keras.io/examples/vision/>