Histopathological Image Analysis using MIL and Transfer Learning

Team Members:

- Lokesh Vadlamudi
- Nupur Yadav
- Chetan Kulkarni

Introduction

- Classifying histopathological slides is challenging because of the two reasons:

1. the histopathological images are high-resolution images that are meant to be examined under the microscope by the pathologists
2. it is extremely difficult to obtain a large dataset of WSIs with detailed pixel-wise annotations that makes it challenging to use a supervised deep learning classifier

- In this project, we have used two techniques:

1. Attention-based Deep Multiple Instance Learning which has proven to be successful in a wide range of medical imaging applications.
2. A “DenseNet201” architecture with weights pre-trained on ImageNet.

Attention-based MIL Approach

- Multiple instance learning (MIL) is a variation of supervised learning where a single class label is assigned to a bag of instances.
- MIL uses the whole-slide labels instead of using pixel-wise annotations.
- In attention-based MIL approach, we have cropped an image in small square patches and made a bag of it. This bag of images will act like a batch.
- Passing it to the feature extractor which is basically a CNN block to generate instance level features.
- Then we are passing the Instance level features to the classifier for getting Instance level attention.
- Here we are getting the attention weights which we are further using for attention aggregation to get the bag level features.
- Then we are applying Dense layer for the classification of the Benign or Malignant

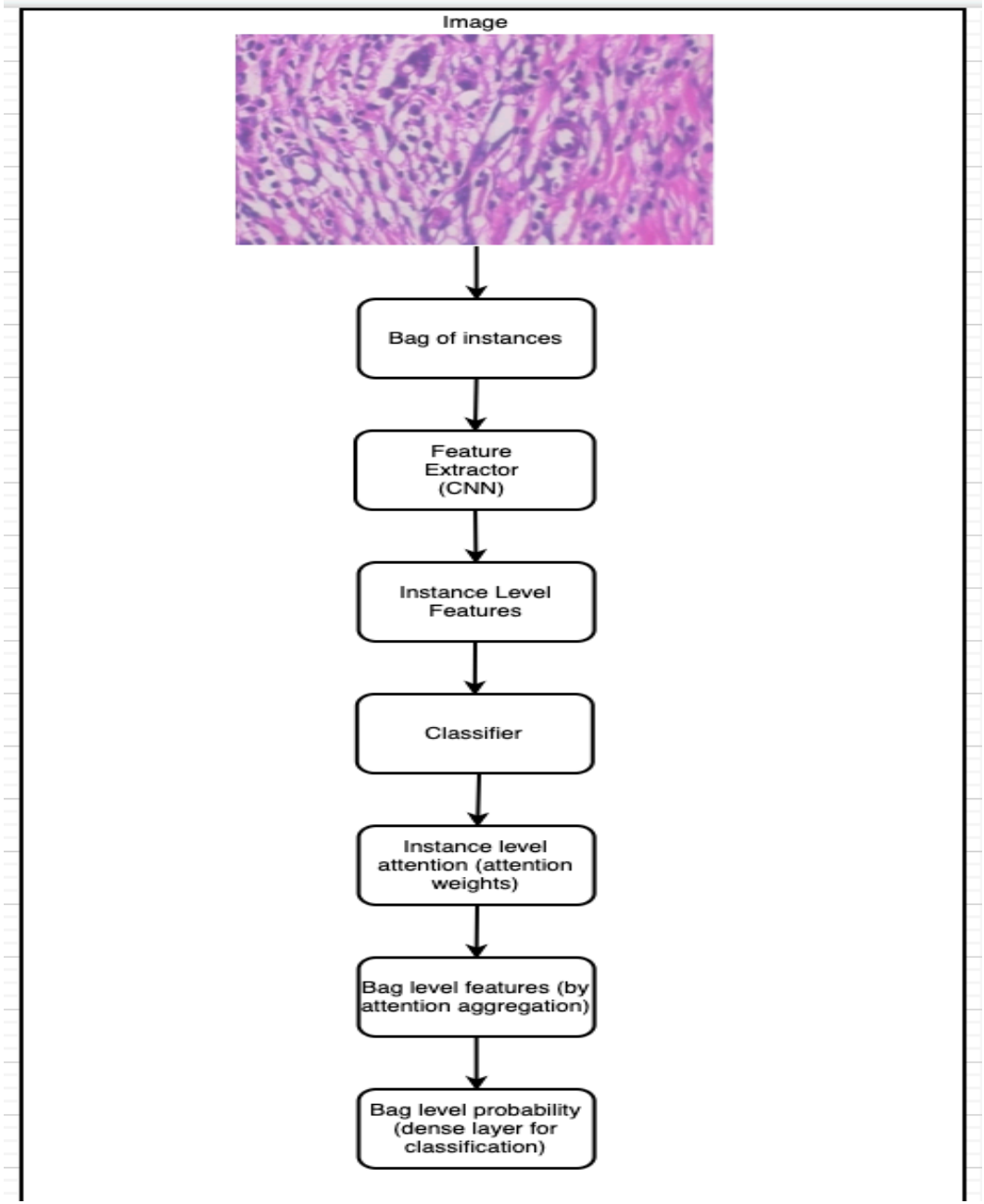


Fig 1. Attention-based MIL approach

Transfer Learning Approach

- Transfer learning allows us to train deep networks using significantly less data that we would need if we had to train from scratch.
- The idea is that the two tasks are not totally disjoint, and as such we can leverage whatever network parameters that model has learned through its extensive training, without having to do that training ourselves.
- Transfer learning has been consistently proven to boost model accuracy and reduce require training time. Less data, less time, more accuracy.
- In this approach, we have used “DensetNet201” architecture with weights pre-trained on ImageNet data.
- We have used ReduceLROnPlateau callback which Reduce learning rate when a metric has stopped improving. fHelps to fine-tuning model weights.
- Models often benefit from reducing the learning rate by a factor of 2-10 once learning stagnates. This callback monitors a quantity and if no improvement is seen for a 'patience' number of epochs, the learning rate is reduced.
- This model achieved a better accuracy than the attention-based MIL approach on breast cancer dataset (BreakHis).

Datasets

- We have used two datasets in our project. Both are taken from Kaggle.

1. BreakHis (Breast Cancer Histopathological Database) : This dataset is divided into two main groups: benign tumors and malignant tumors. It is composed of 9,109 microscopic images of breast tumor tissue collected from 82 patients using different magnifying factors (40X, 100X, 200X, and 400X). To date, it contains 2,480 benign and 5,429 malignant samples (700X460 pixels, 3-channel RGB, 8-bit depth in each channel, PNG format).
2. Breast Histopathology Images (IDC) dataset : Invasive ductal carcinoma (IDC) is the most common form of breast cancer. The original dataset consisted of 162 whole mount slide images of Breast Cancer specimens scanned at 40x. From that, 277,524 patches of size 50 x 50 were extracted (198,738 IDC negative and 78,786 IDC positive).

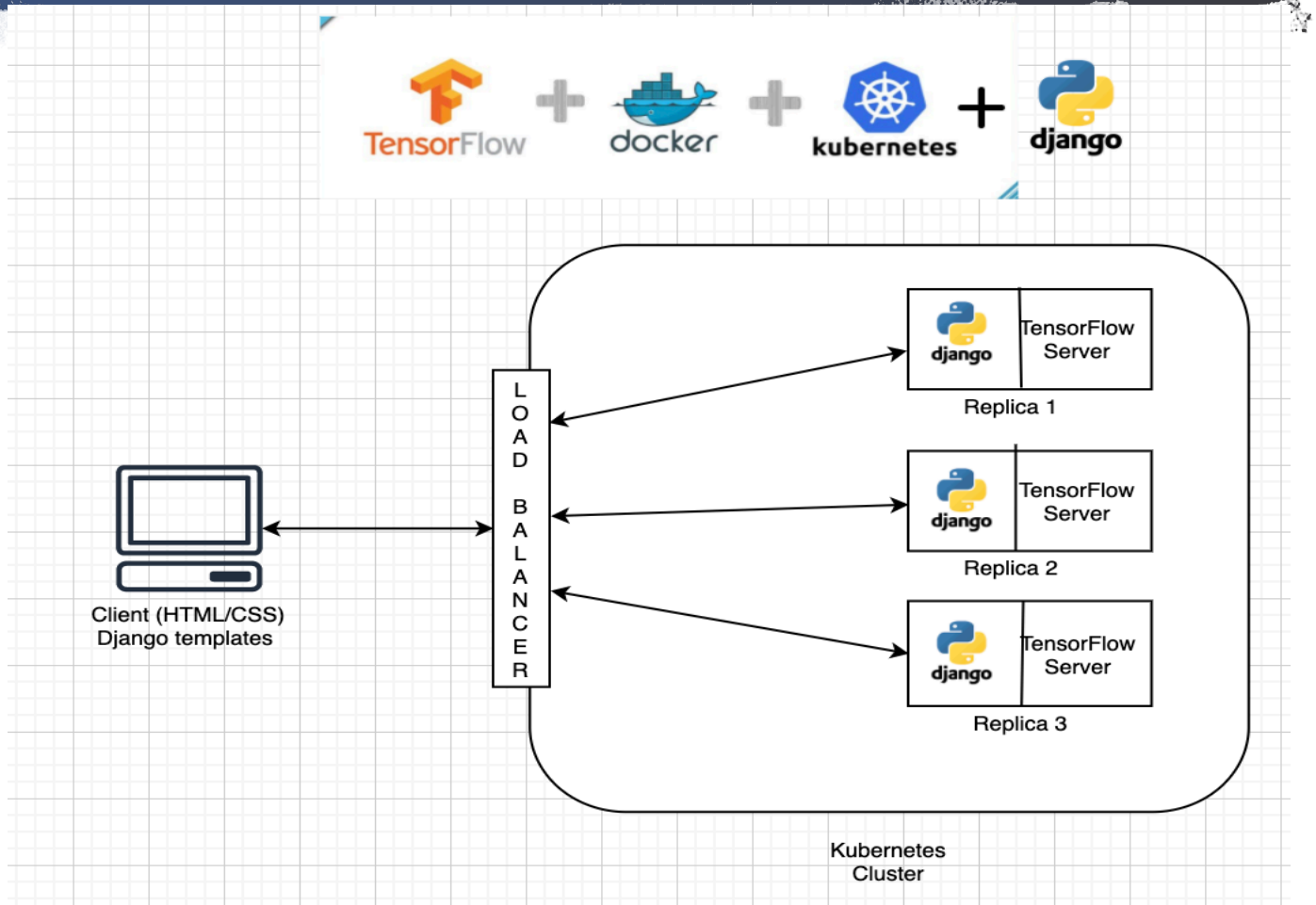
Observations

Technique	Accuracy
Attention-based MIL on BreakHis dataset (Pytorch implementation)	69%
Attention-based MIL on Breast Cancer histopathological (IDC) images (IDC dataset, data augmentation and Keras implementation)	73%
Transfer Learning on BreakHis dataset (using DenseNet201)	94%

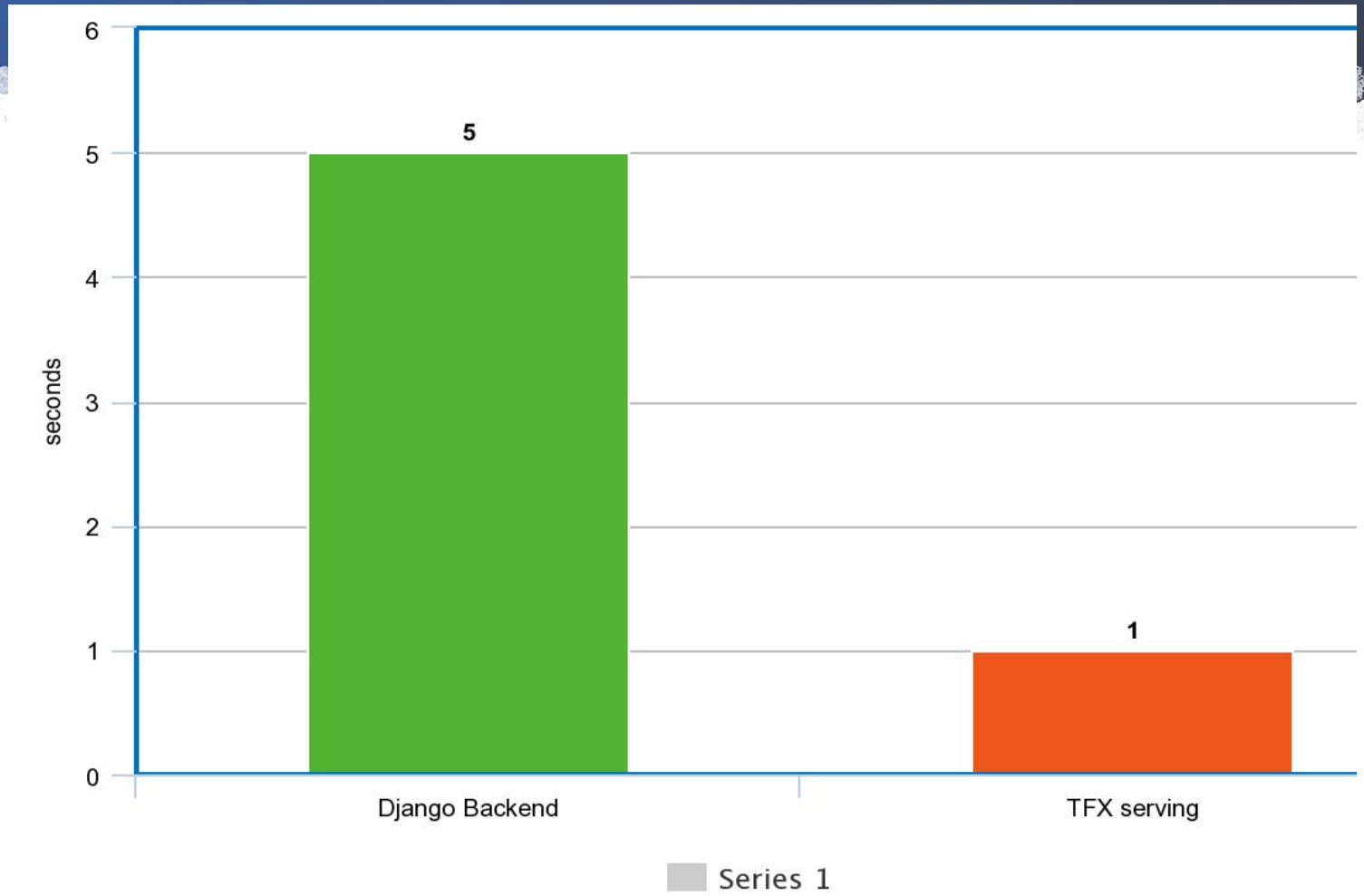
Hence, we used this transfer learning model for deployment and making future inferences.

Deployment architecture

We have used Django for building web application and TensorFlow serving, docker, Kubernetes for model deployment.



Why tensorFlow serving?



Creating your own serving image

Steps

- docker run -d --name serving_base tensorflow/serving
- docker cp models/<my model> serving_base:/models/<my model>
- docker commit --change "ENV MODEL_NAME <my model>" serving_base <my container>

Run the model:

- docker run -p 8501:8501 <container name>

Dockerize Web Application

Dockerfile

```
FROM python:3.6

ENV PYTHONUNBUFFERED 1

RUN apt-get update

RUN apt-get install 'ffmpeg'\
    'libsm6'\
    'libxext6'  -y


RUN mkdir /app

WORKDIR /app

ADD . /app/

RUN pip3 install --upgrade pip

RUN pip install -r requirements.txt

EXPOSE 8000
CMD ["python", "manage.py", "runserver", "0.0.0.0:8000"]
```

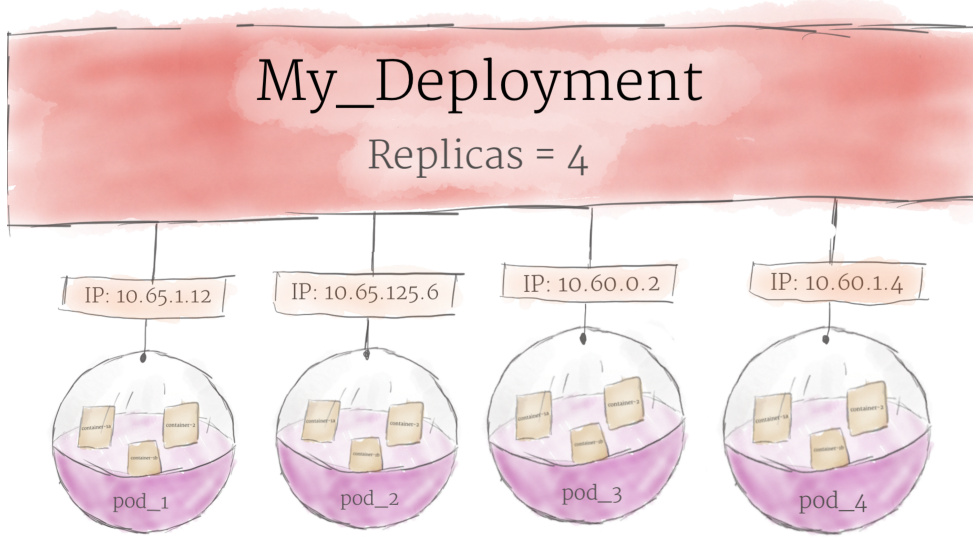
Upload To Docker Hub

- docker push dockerRepo:latest

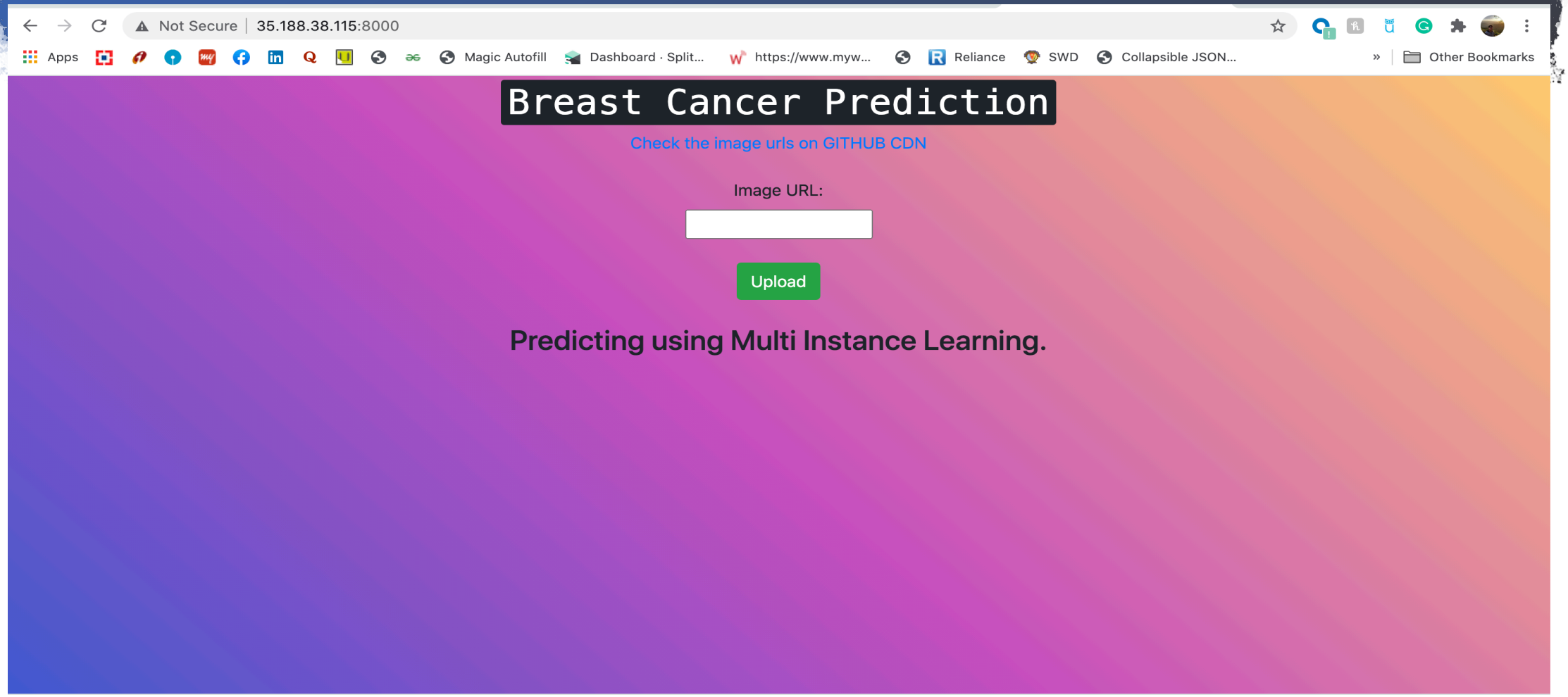
Kubernetes

Using Kubernetes Engine on GCP.
kubectl create deployment --image lokv007/dldjango:latest django

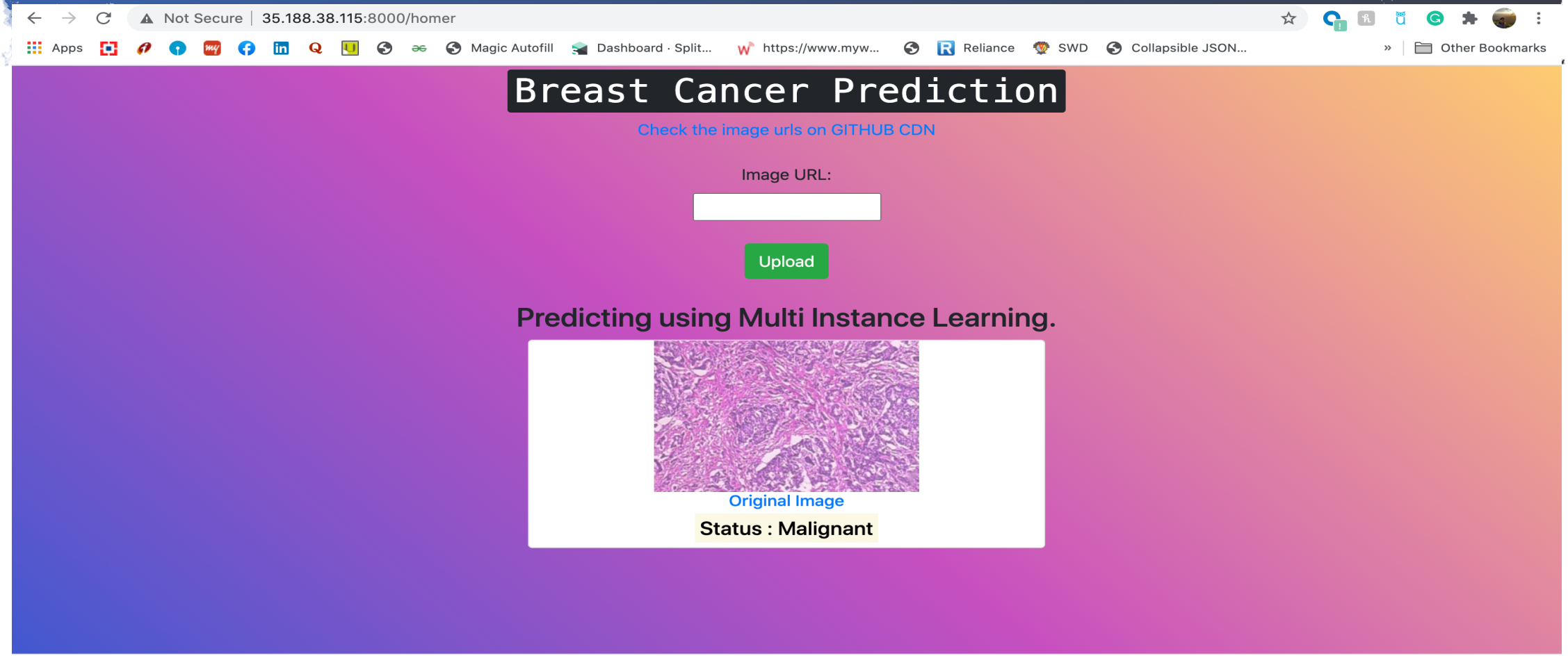
To expose the service,
kubectl expose deployment django --type=LoadBalancer --name=my-service-app –port 8000



UI screenshots



UI screenshots



Team members and their contributions

Team Member	Contribution(s)
Chetan Kulkarni	Attention-based MIL on BreakHis dataset (Pytorch implementation), model deployment
Lokesh Vadlamudi	Attention-based MIL on Breast Cancer histopathological (IDC) images (IDC dataset, data augmentation and Keras implementation), model deployment
Nupur Yadav	Transfer Learning on BreakHis dataset (using DenseNet201 with weights pre-trained on ImageNet), model deployment

Delta

We have taken the initial code from https://github.com/Dipeshtamboli/Image-Classification-and-Localization-using-Multiple-Instance-Learning which is in pytorch.

- The delta that we have done includes:
- We have **implemented the Multiple instance learning using Keras**. And have used a **different dataset that is IDC dataset** taken from Kaggle that had more data samples.
- We have also **performed data augmentation** to increase training data and improve model accuracy.
- We also **implemented transfer learning technique** on BreakHis dataset where we have **used "DenseNet201" architecture with pre-trained weights**.
- We **have implemented TFX interactive context** for the above transfer learning colab.
- And are **using tensorflow serving, docket and Kubernetes to serve and deploy our model** in production.

Code Walkthrough and Demo..