NOT VADAPAV

CS725/403 - Project

140100025 - Tejas Srinivasan
140260007 - Anuj Shetty
140260016 - Kumar Ayush
160040093 - Alok Bishoyi

The Problem

We aim to **classify if a given image is that of a vadapav or not (binary classification task).**

This can be used as a basis for a general food classification app, if we have training data for lots of different types of food items.

The challenge is to classify images which are taken in a natural setting, under different conditions of lighting and angles. Moreover, vadapavs look similar to burgers, and while burgers are commonly referred to as the American vadapav, we believe there is a huge difference.

CNNs: Not Fake News

- **Convolutional Neural Networks (CNNs)** are a set of neural network architectures which are commonly used for image classification tasks.
- Have recently come to the forefront, as hardware has caught up.
- Architecture generally consists of many layers
  - **Convolution layers**: Applied convolution operation to input; emulates the response of an individual neuron to visual stimuli.
  - **Pooling layers**: Reduces size of convolution outputs
  - **Fully connected layers**
- Shared parameters at a given layer allow us to reduce number of parameters trained

The Data

Images were sourced from the following:

 **Vadapavs**: ~2000 pictures scraped from Instagram using #vadapav, **840 manually selected** as positive images in the dataset.

 **Not Vadapav**: **640 pictures** from Instagram, using #food & #fast food (includes pictures of people and non-food items)

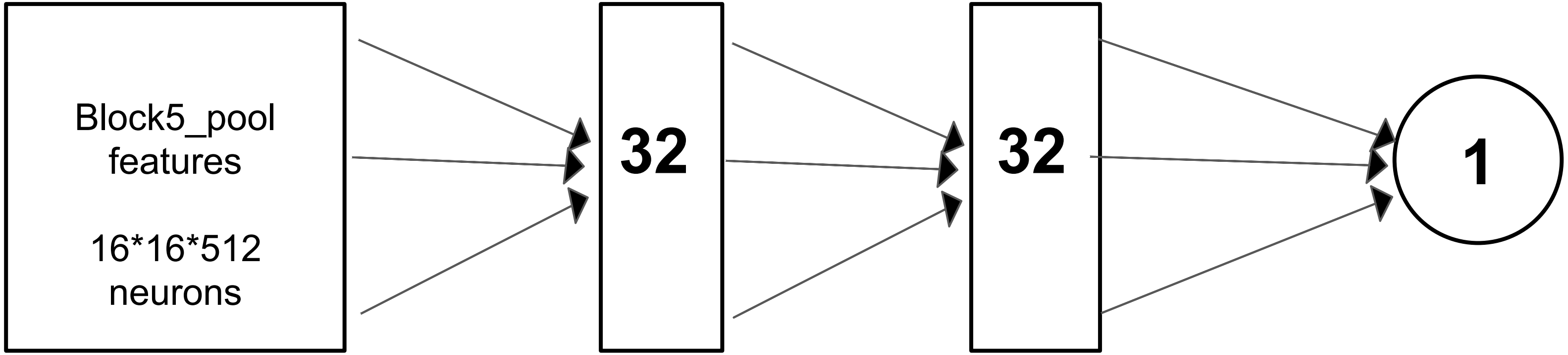
 **Burgers**: **200 pictures** from Instagram, using #burger, 160 manually selected

Google Images did not yield images of good quality. Instagram gave **high quality natural images,** under **different conditions of lighting and angles**.

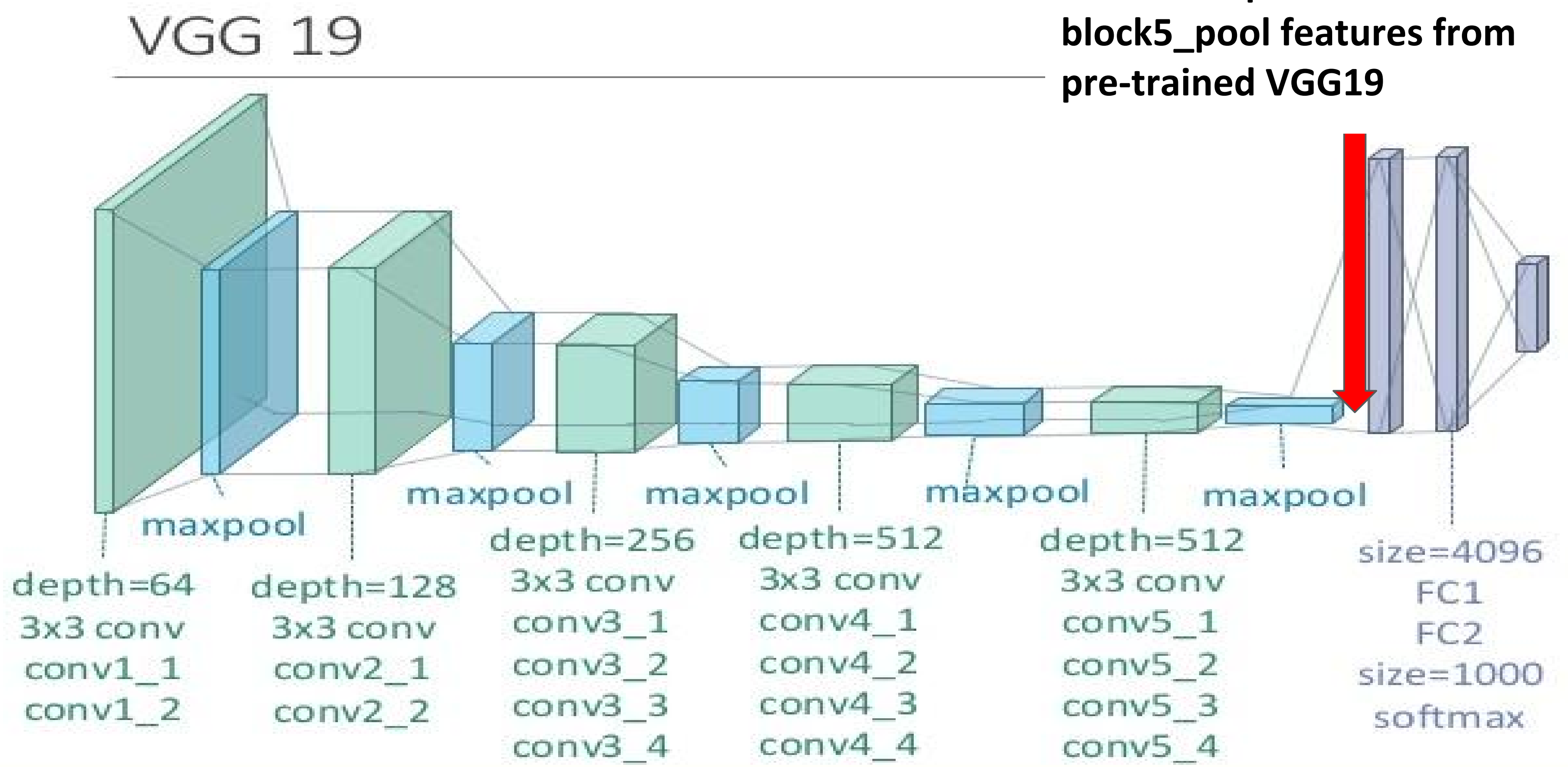
A script was developed for us to easily scroll through hundreds of pics and tag them easily.

Approach I: Training Fully Connected Layers

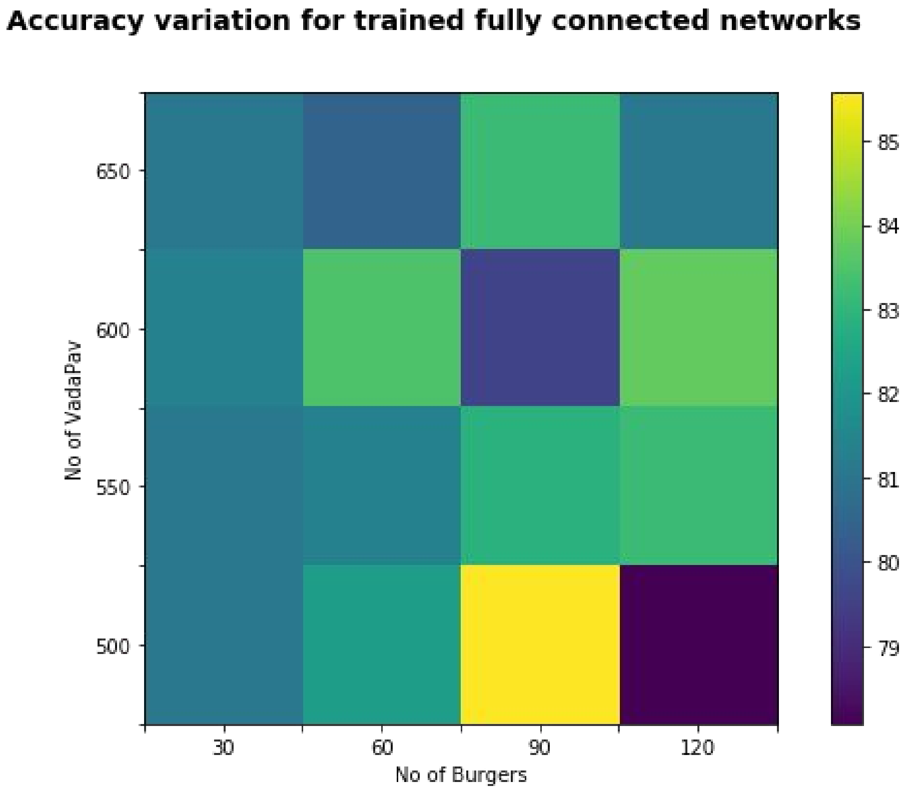
- We used the **VGG19 architecture pre-trained weights** to extract convolution features for the images.
- We extracted features after the **block5_pool** layer.
- The extracted features were then flattened, and **3 fully connected layers** were trained. Adam optimizer was used, and training was for 10 epochs.



Extraction point for block5_pool features from pre-trained VGG19



Results I: Training Fully Connected Layers



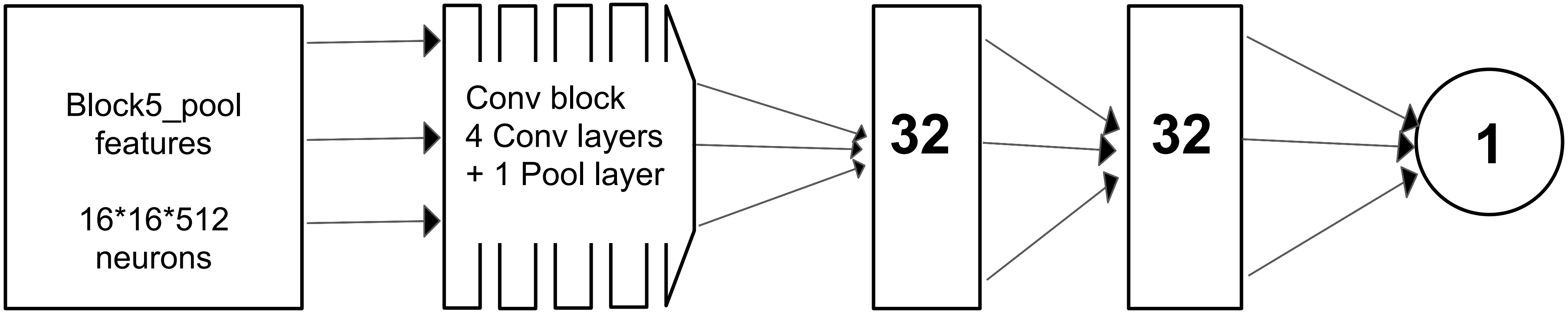
Results I: Training Fully Connected Layers

The best model was obtained from using 500 vadapavs and 90 burgers in the training. The confusion matrix on the test data was as follows

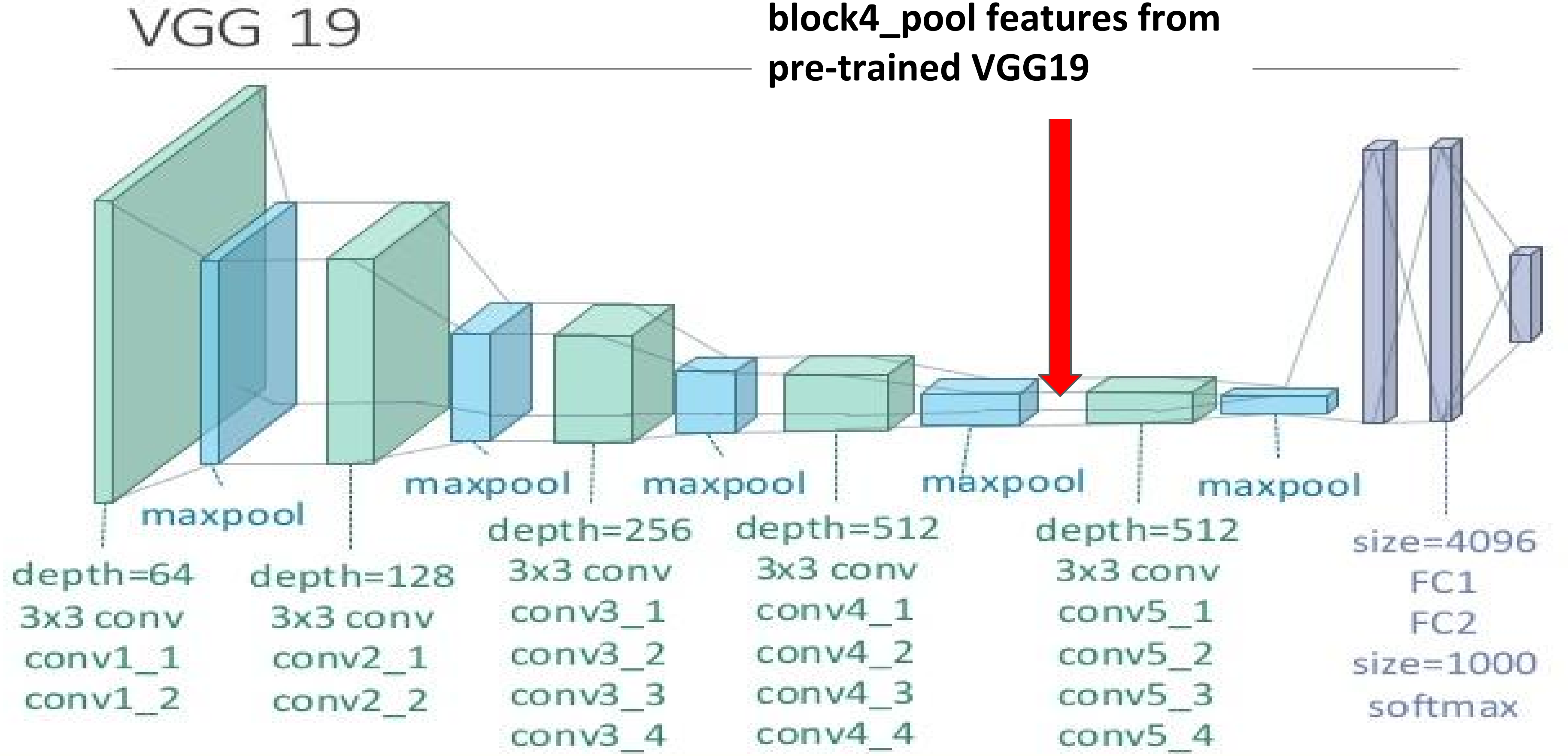
True label (->) Predicted label	Not Vadapav	Vadapav	Burger
Not Vadapav	120	30	26
Vadapav	10	139	8

Approaches II: Training Convolutional Layers

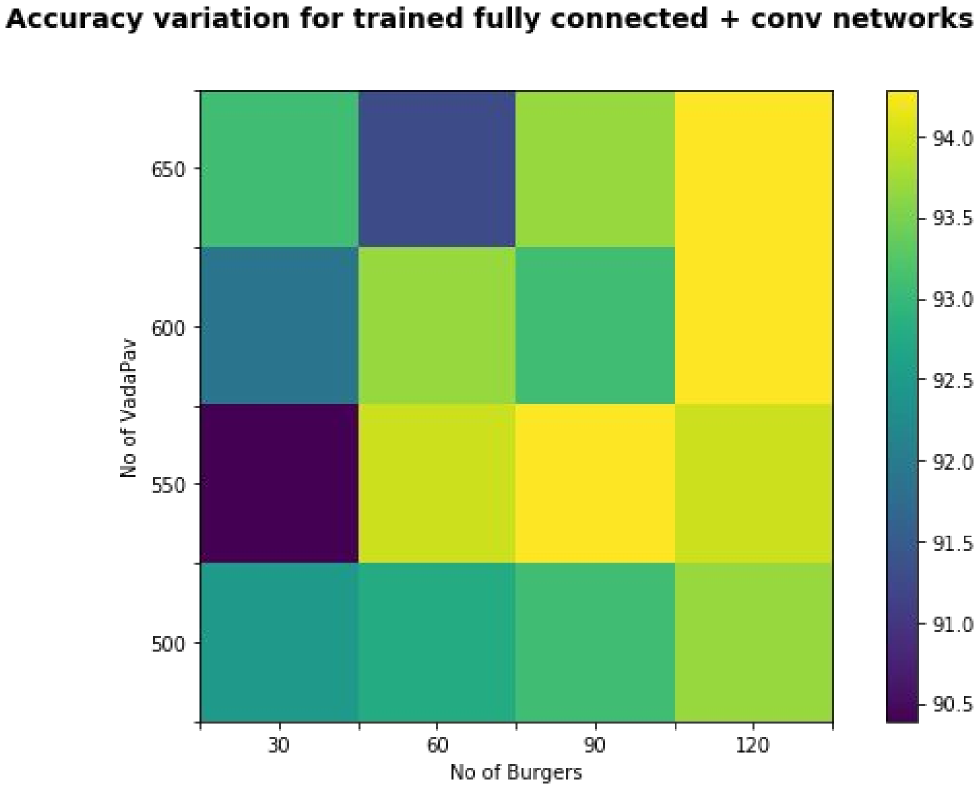
- Extracted features after the **block4_pool layer** of **VGG 19.**
- Features passed through our **convolutional network block**, then through 3 **fully connected** layers as before.
- We **initialised the weights** of our 5th block to the **pre-trained imagenet weights** to improve accuracy.
- **SGD** used with no momentum, slight decay



Extraction point for block4_pool features from pre-trained VGG19



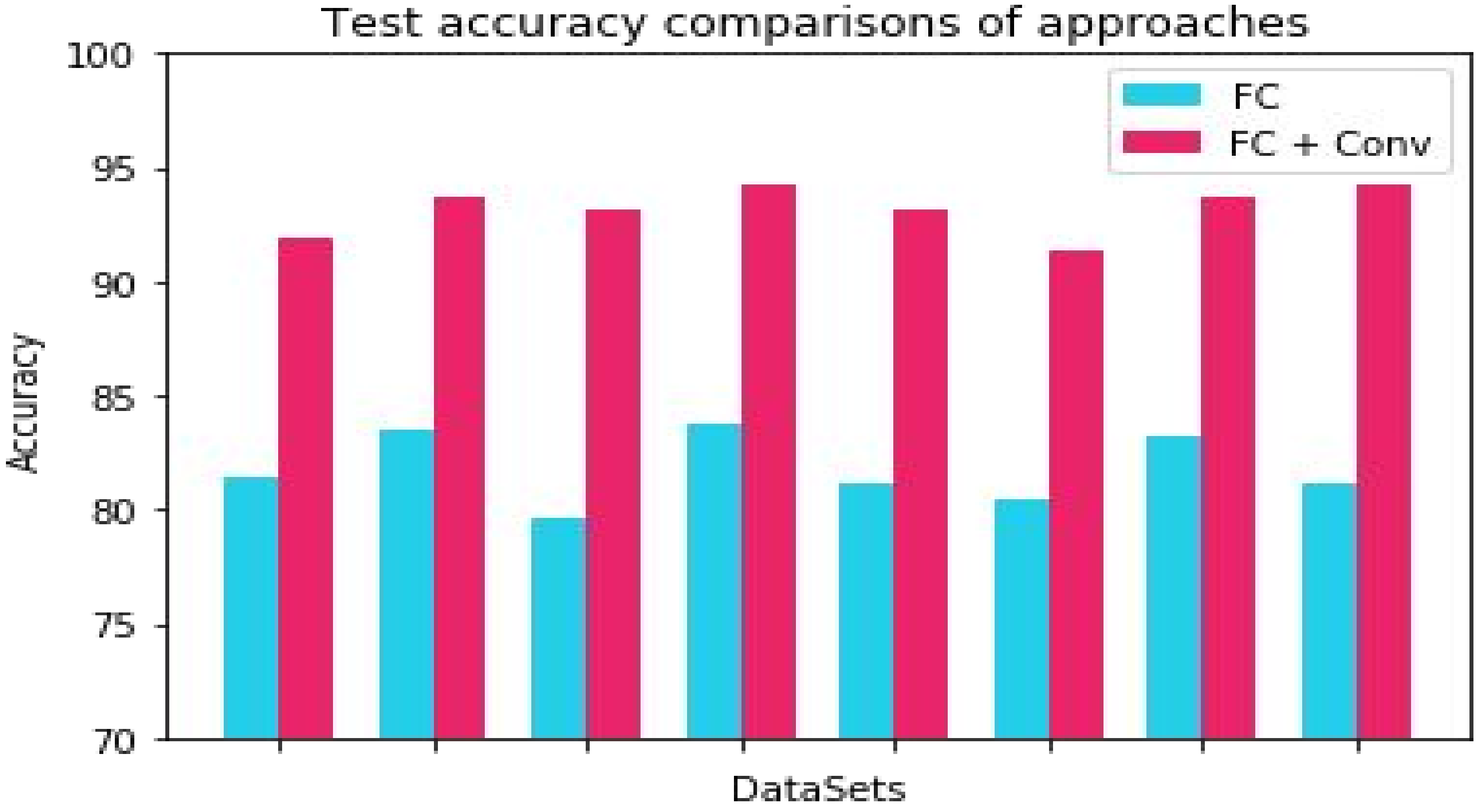
Results II: Training Convolutional Layers



Results II: Training Convolutional Layers

The best model was obtained from using 650 vadapavs and 120 burgers in the training. The confusion matrix on the test data was as follows

True label (->) Predicted label	Not Vadapav	Vadapav	Burger
Not Vadapav	121	9	33
Vadapav	9	160	1


Test accuracy comparisons of approaches
FC
FC + Conv
Accuracy
DataSets
100
95
90
85
80
75
70

The Demo

Now, if you would kindly turn your attention to the small screen...