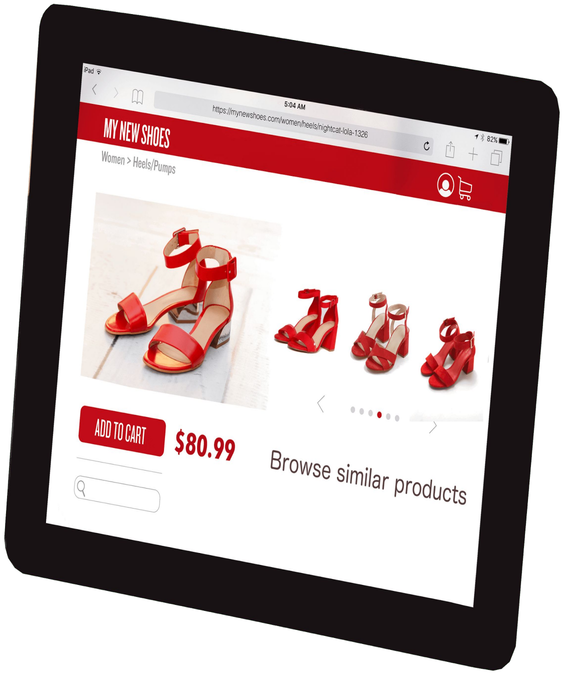
IMAGE BASED RECOMMENDER SYSTEM

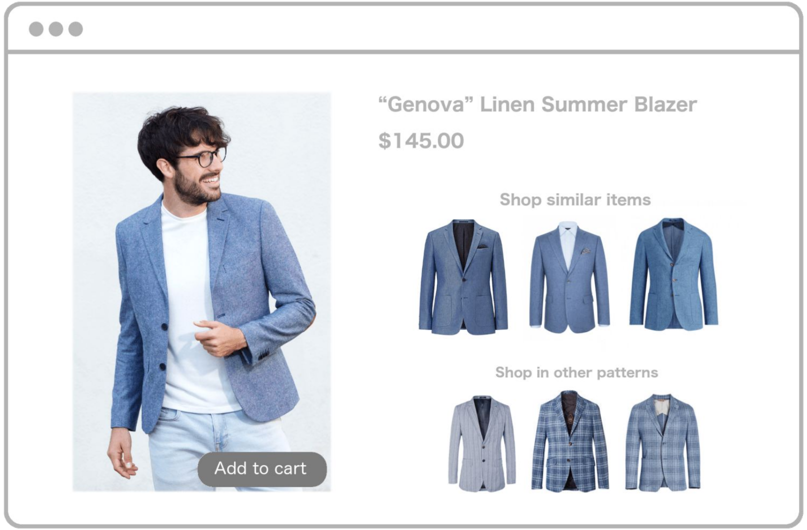
Katakam Vasavi
Guided by:Prof.Nagesh Kolagani

PROBLEM STATEMENT

- Most on-line shopping search engines are still largely depend on knowledge base and use key word matching as their search strategy to find the most likely product that consumers want to buy. This is inefficient in a way that the description of products can vary a lot from the seller's side to the buyer's side.
- So Aim is to develop a smart search engine for online shopping. Basically it uses images as its input, and tries to understand the information about products from these images.
- i.e.,Suggesting visually similar Images based on the input given to the system.

UPLOADED IMAGE-VISUALLY SIMILAR IMAGES





ABOUT THE DATASET

AMAZON PRODUCT DATA

- This data includes title of the product,imageurl,price,sales-rank,brand info,category to which the product belongs to of nearly 9.4 million products spanning over 20 categories.
- For this task I am taking 500 images from each category so overall 10000 images.
- Splitting this data into Training,validation,test sets in the ratio of 7:2:1
- Those 20 categories are

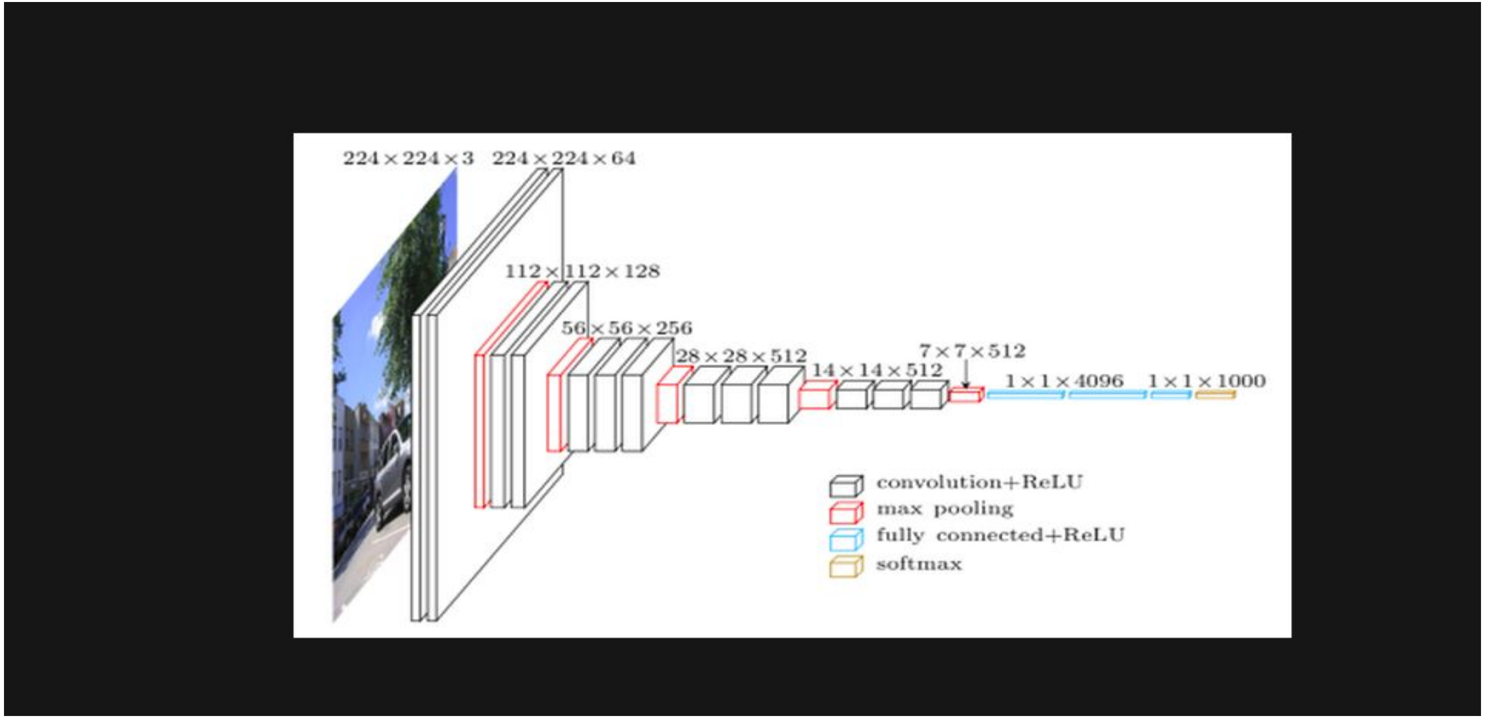
1)Music	2)CellPhones	3)Sports	4)Toys	5)Beauty	6)Grocery	7)Clothing	8)Home	9)Office	10)Health
11)Tools	12)Baby	13)Electronics	14)Movies and TV	15)Pet Supplies	16)Lawn and Garden	17)Video Games	18)Automotive	19)Apps	20)Books

IMPLEMENTATION

The input to our algorithm is an image of any object that the customer wants to buy. We then use a Convolutional Neural Network(CNN) model to classify the category that this object probably belongs to,and use the input vector of the last fully connected layer as a feature vector to feed in a similarity calculation CNN model to find the closest products in our database. More concretely, the two functionalities that we want to achieve in the recommendation system are:

1. **Classification:** given a photo of the product taken by the customer, find the category that this product most likely belong to.
2. **Recommendation:** Given the features of the photo and the category that this product belongs to,calculate similarity scores within this category and find the most similar products in our database. Ideally, people looking for iPhones would be recommended iPhones.

VGG-16 ARCHITECTURE



FINE TUNING VGG-16 MODEL FOR CLASSIFICATION TASK

- This model is for 1000 classification problem but we have 20 categories so removing the top layer in the model and adding the softmax layer over 20 classes.
- Using weights of Imagenet as Initialization to the model.
- Right now experimenting with different optimizers and Hyperparameters to select the best one for the task.

  Optimizers like Adam,Rmsprop,sgd etc.,

VGG-16 MODEL AS FEATURE EXTRACTOR FOR RECOMMENDATION

- After classification is done we know the category to which the product belongs.
- By removing the dense layers of the trained VGG model we will get the features of every image in the category.
- we check the similarity between the features of the Input Image vs category Images.
- The ones having the similarity greater than the defined threshold value are recommended for the users.

RESULTS

Adam Optimizer with Learning rate=0.0001

Batch size:64

Epochs:40

Training set Accuracy:85%

Validation set:51%

Test set:49%

TRAINING PROCESS

```
Epoch 15/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.9764 - acc: 0.3726 - val_loss: 2.4468 - val_acc: 0.2553
Epoch 16/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.9230 - acc: 0.3946 - val_loss: 2.3003 - val_acc: 0.3158
Epoch 17/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.8011 - acc: 0.4334 - val_loss: 2.3598 - val_acc: 0.2958
Epoch 18/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.7093 - acc: 0.4606 - val_loss: 2.3466 - val_acc: 0.3079
Epoch 19/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.5698 - acc: 0.5003 - val_loss: 2.3358 - val_acc: 0.3095
Epoch 20/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.4779 - acc: 0.5266 - val_loss: 2.4962 - val_acc: 0.3279
Epoch 21/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.3624 - acc: 0.5693 - val_loss: 2.5666 - val_acc: 0.3432
Epoch 22/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.2738 - acc: 0.5934 - val_loss: 2.5838 - val_acc: 0.3268
Epoch 23/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.1863 - acc: 0.6209 - val_loss: 2.8275 - val_acc: 0.3311
Epoch 24/200
6650/6650 [==============================] - 220s 33ms/step - loss: 1.0537 - acc: 0.6588 - val_loss: 2.8549 - val_acc: 0.3332
Epoch 25/200
6650/6650 [==============================] - 220s 33ms/step - loss: 0.9507 - acc: 0.6973 - val_loss: 2.7781 - val_acc: 0.3311
Epoch 26/200

4544/6650 [==================>..........] - ETA: 1:03 - loss: 0.7807 - acc: 0.7535

[ ] !python2 "/content/drive/My Drive/GpuNotebooks/vgg16_drive.py"
```

FUTURE PLANS

- Recommendations on Styles and Substitutes.



- Improving the performance of the system.
- Developing a good User Interface(Mini Recommender System).

REFERENCES

- Dataset link:https://cseweb.ucsd.edu/~jmcauley/
- http://cs231n.stanford.edu/reports/2017/pdfs/105.pdf
- https://cseweb.ucsd.edu/~jmcauley/pdfs/sigir15.pdf
- https://arxiv.org/pdf/1703.05192.pdf

THANK YOU