Offline Shopping Assistant

The University of Adelaide

Supervisors

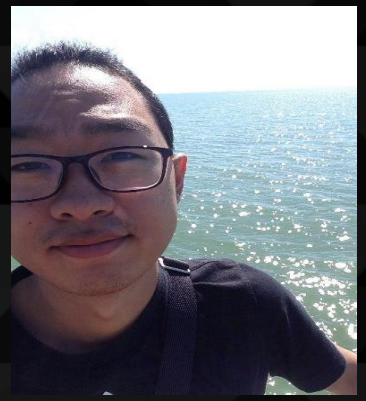
Pro. Ali

Dr. Lingqiao

Group members

Qiaoyang Luo
a1728768

Qiming Shao
a1699631



Junwei Li
a1710640

Achievements & Problems in Semester 1

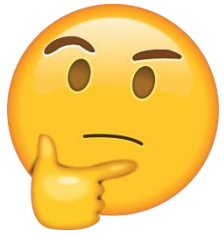
Buy the fruits with the lowest price

1 Taking a photo on Android Application

2 Recognizing fruits by classification model

3 Searching information on DB

4 Presenting results on mobile phone



Problems & Limitations

- **Slow Speed**
  - Uploading image
  - Communication
  - Parsing Txt
- **Types**
  - Only 10 different fruits
- **Simple System**
  - Only for single user
  - Only for customer
  - Only for Android

Semester 2

Build a platform

Customer and Seller

Build a cross-platform App

Android and iOS

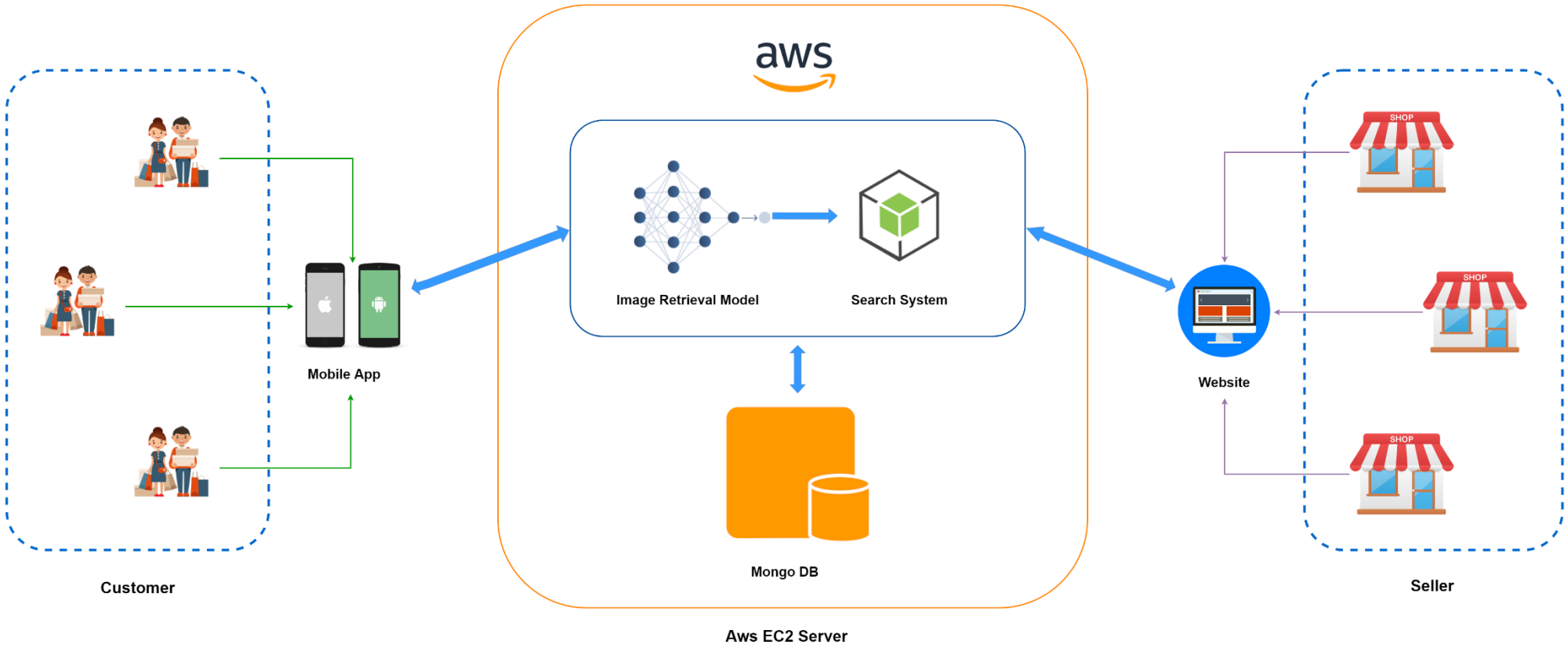
Reconstruct the system

React-Native, Express,
Node.JS, Redux

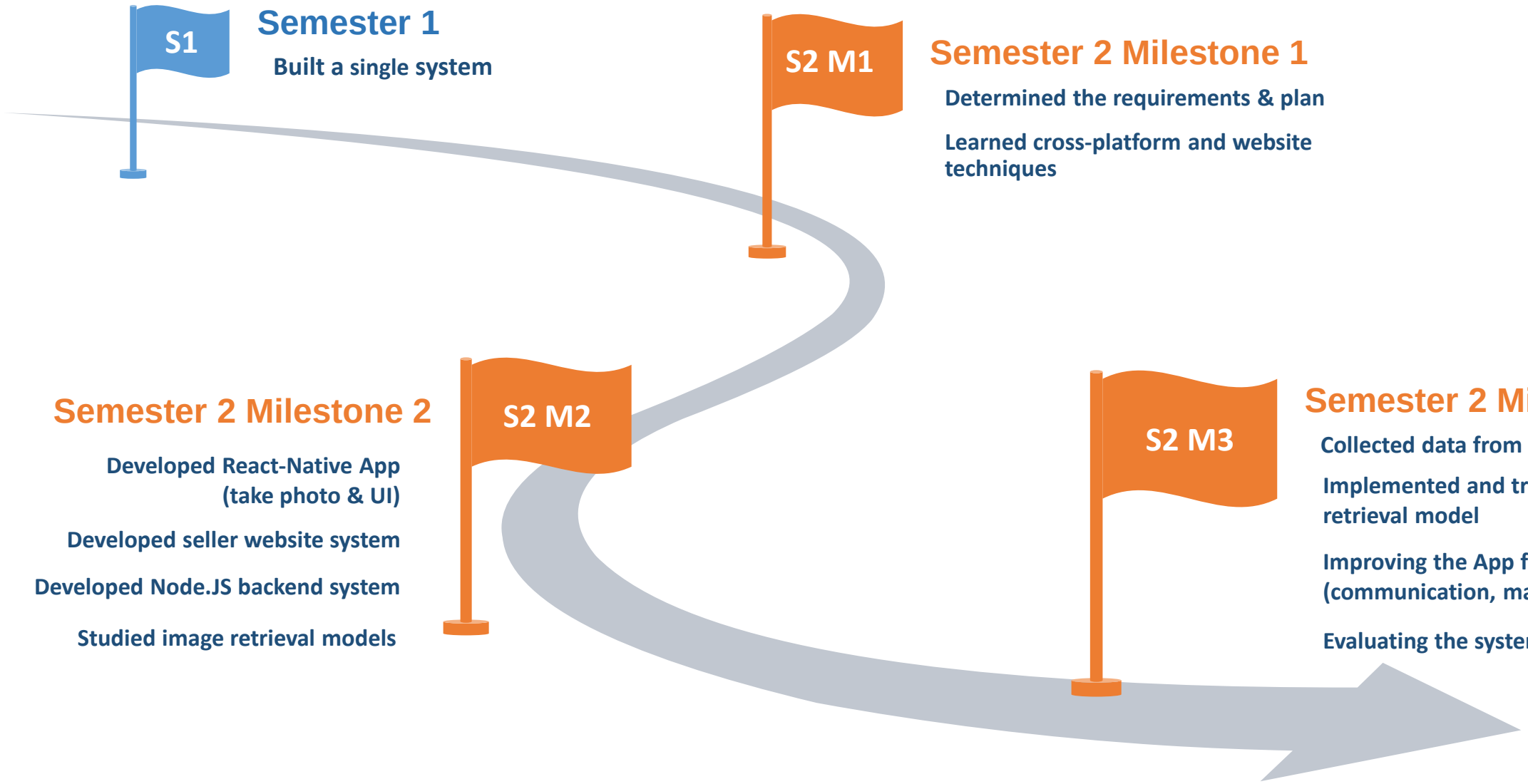
Reconstruct computer vision model

Classification -> Image Retrieval

Platform structure



Timeline

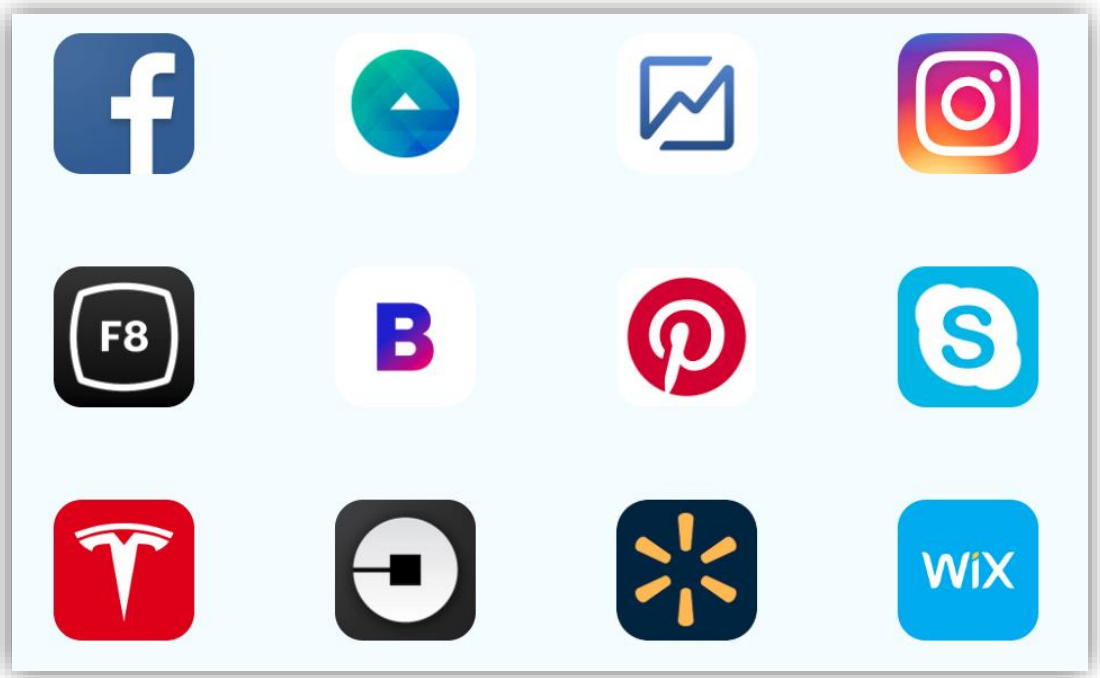

S1
Semester 1
Built a single system
S2 M1
Semester 2 Milestone 1
Determined the requirements & plan
Learned cross-platform and website techniques
S2 M2
Semester 2 Milestone 2
Developed React-Native App (take photo & UI)
Developed seller website system
Developed Node.JS backend system
Studied image retrieval models
S2 M3
Semester 2 Milestone 3
Collected data from Google Image
Implemented and trained image retrieval model
Improving the App function (communication, map)
Evaluating the system

React Native App

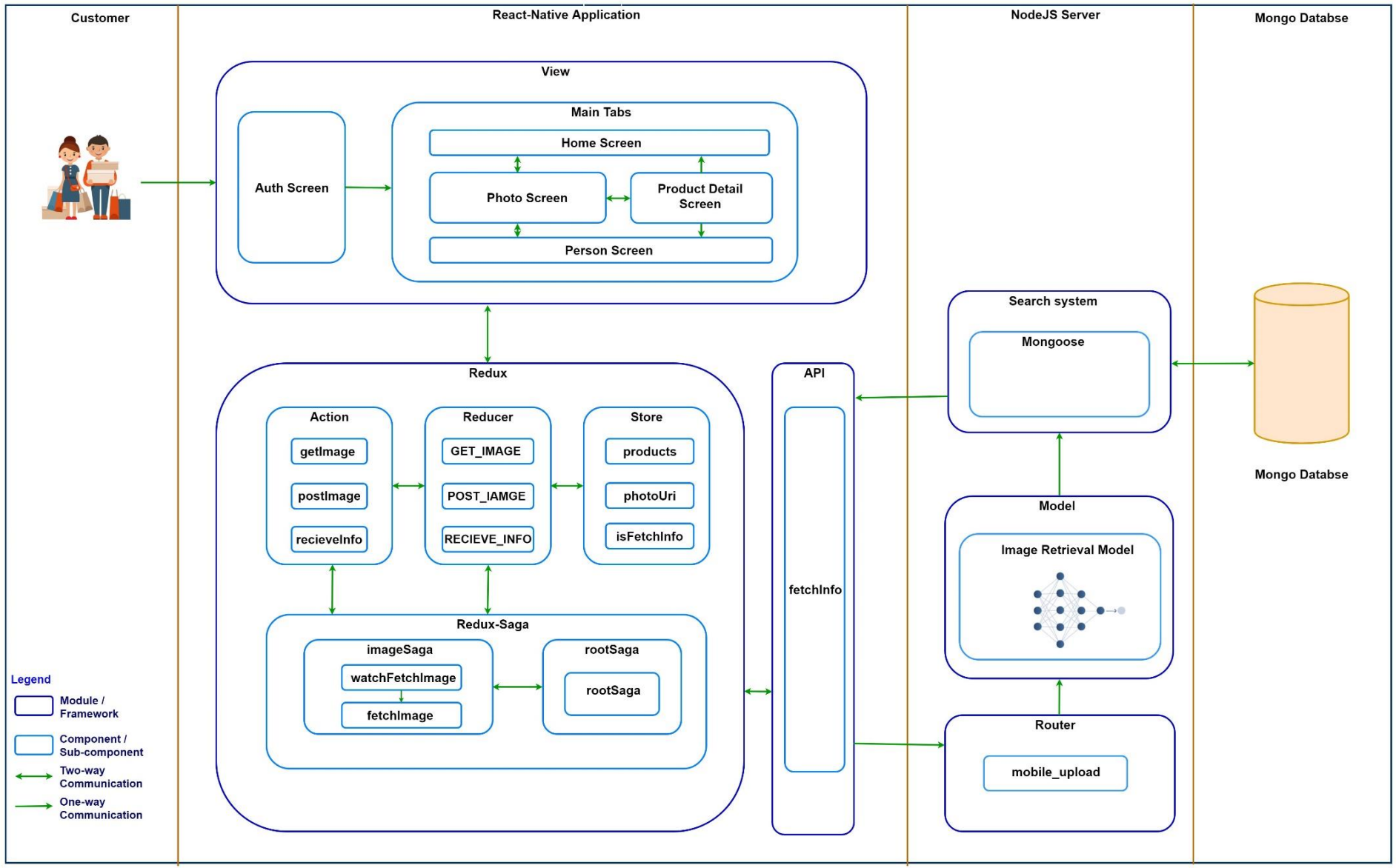
Cross-platform

Advance speed

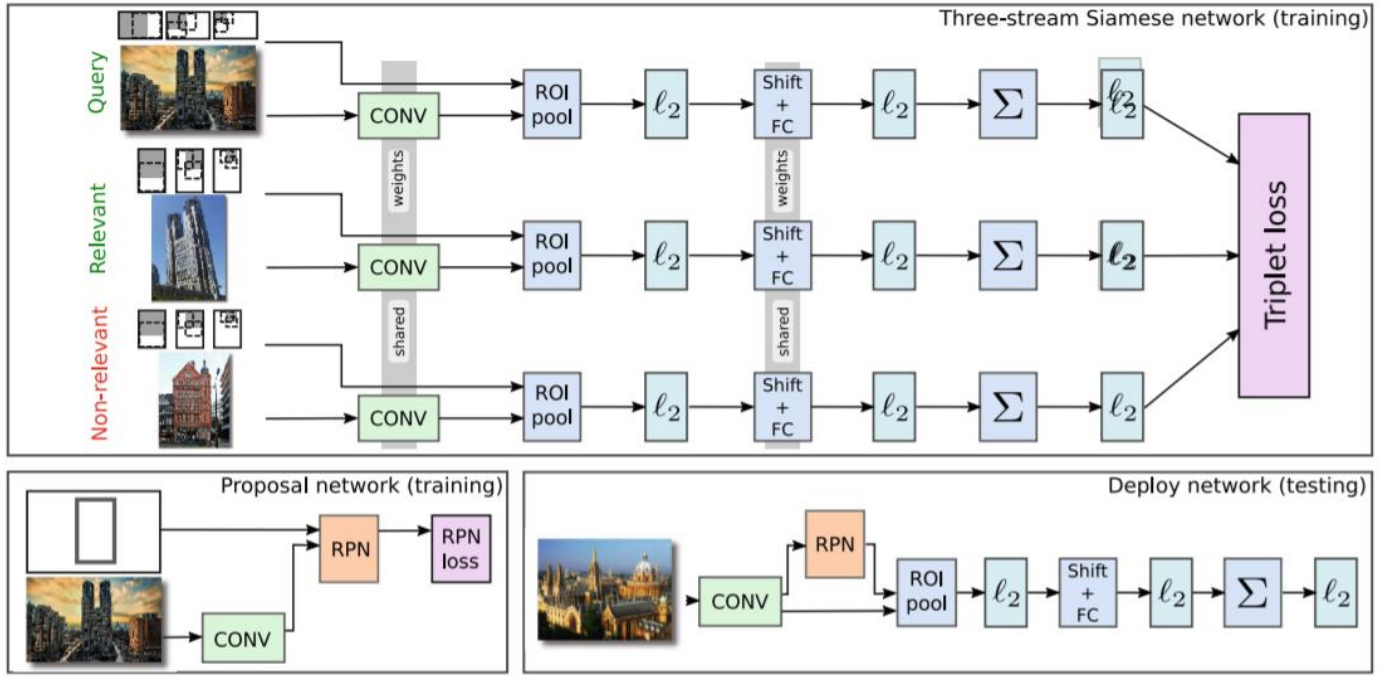
High Performance



App Architecture



Research - Content Based Image Retrieval Model



[1]

10 classes

Over 10,000 classes

Three-stream Siamese network c_n^3 VS **Two channel convolutional neural network** c_n^2 VS **Classification model with KNN** n

[1] Gordo, Albert, et al. "Deep image retrieval: Learning global representations for image search." European Conference on Computer Vision. Springer, Cham, 2016.

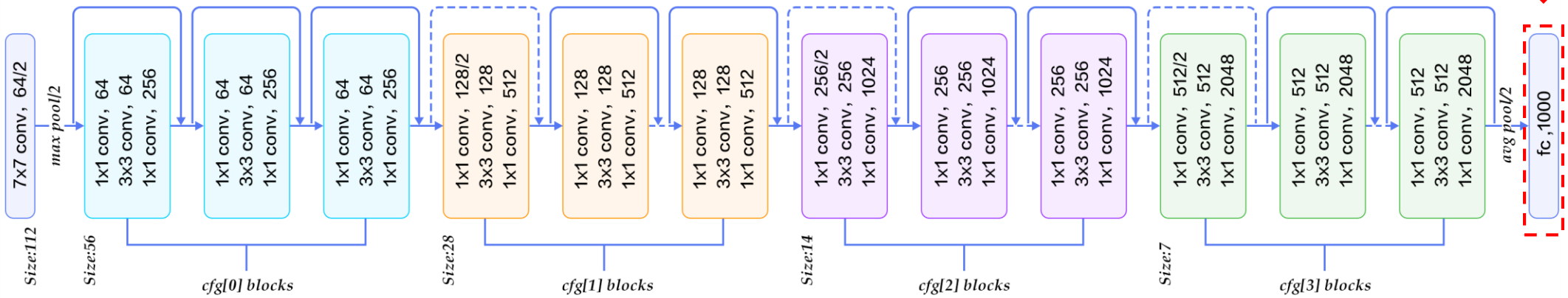
Solution – Select base model

Model	Train Acc	Val Acc	Epoch time (s)	Total time
MobileNet(not pretrained)	94.80%	87.84%	7	00:27:12
VGG 19 (pretrained)	94.24%	92.70%	35	01:00:31
VGG 19 bn (pretrained)	91.93%	93.05%	36	01:01:19
Densenet 161 (pretrained)	95.23%	92.70%	21	00:36:50
Resenet 152 (pretrained)	92.95%	92.36%	24	00:41:06

Solution – Finetune & KNN

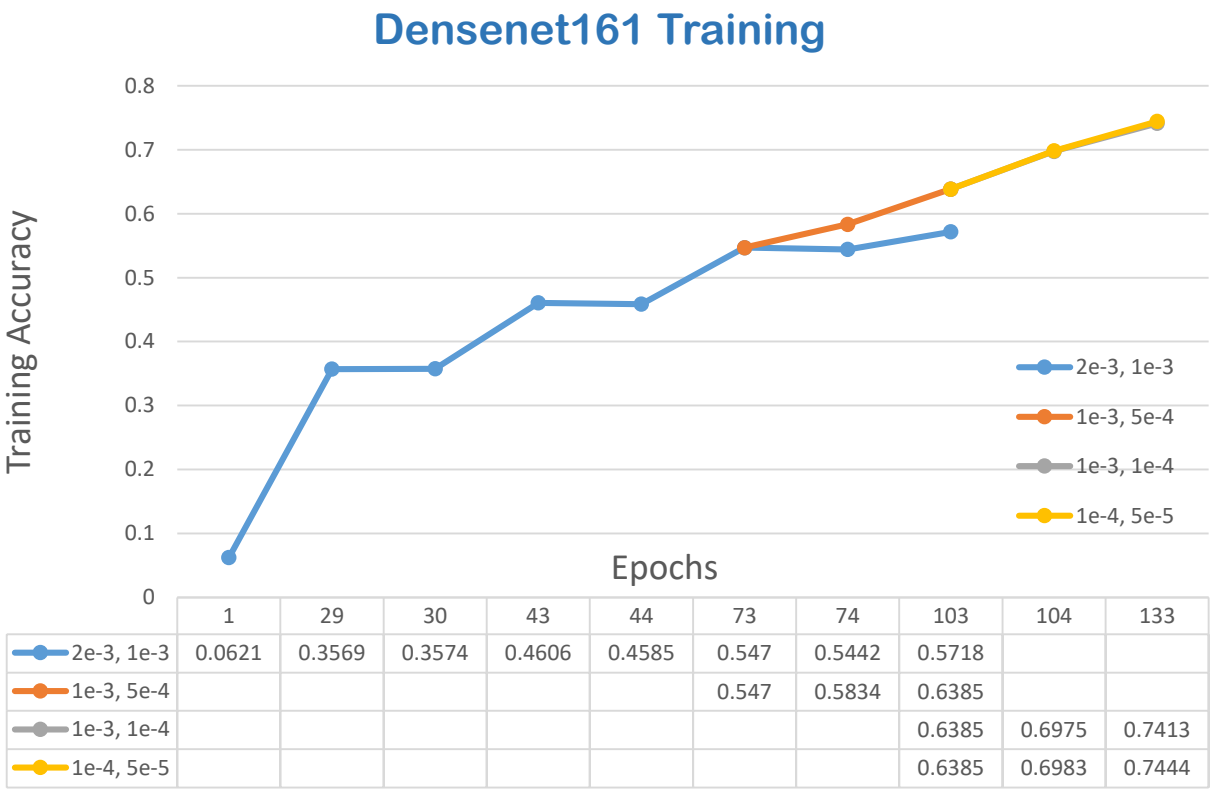
Resnet 152

Remove last layer

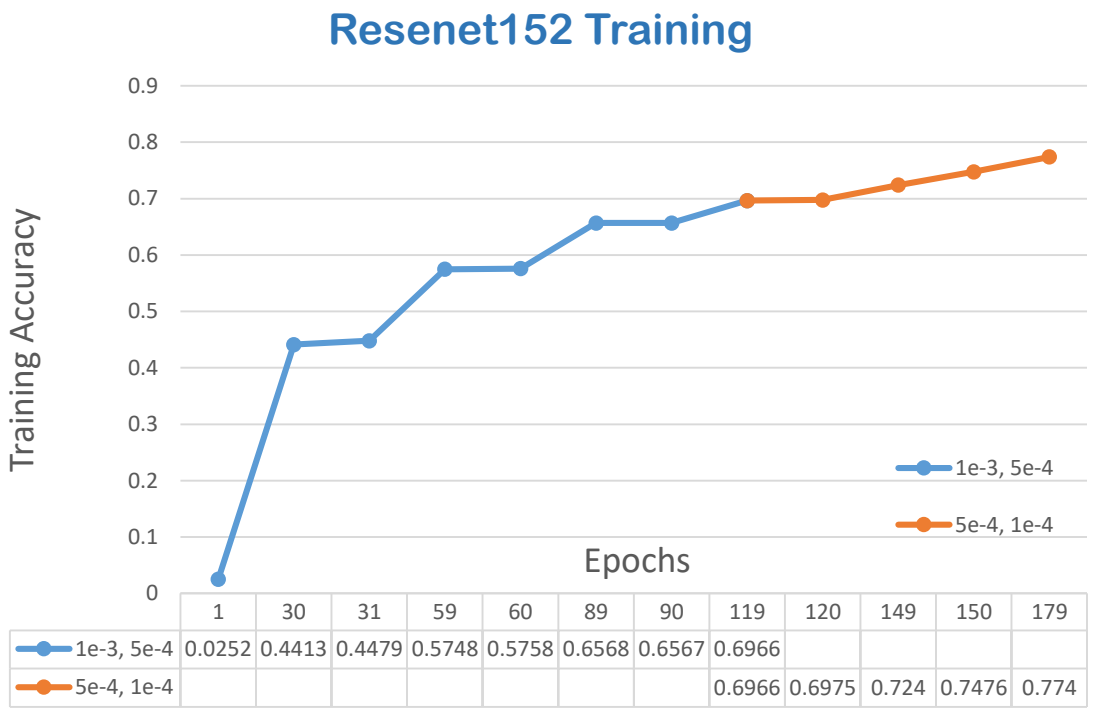


Extract feature + One nearest neighbor + Top 3

Training & Results

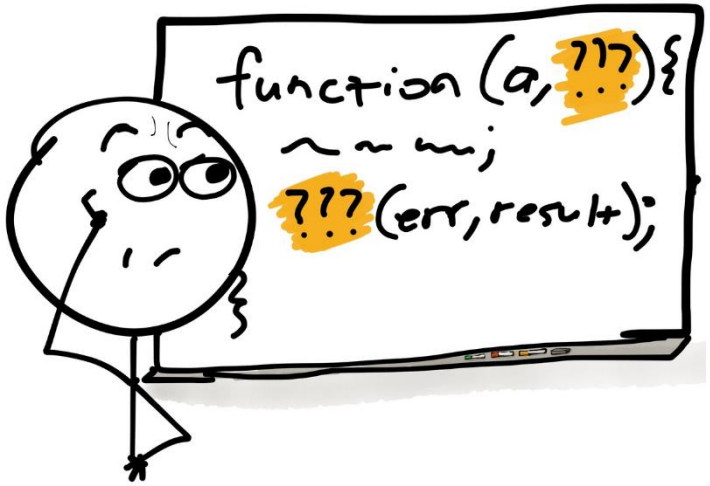


	1	29	30	43	44	73	74	103	104	133
2e-3, 1e-3	0.0621	0.3569	0.3574	0.4606	0.4585	0.547	0.5442	0.5718		
1e-3, 5e-4						0.547	0.5834	0.6385		
1e-3, 1e-4								0.6385	0.6975	0.7413
1e-4, 5e-5								0.6385	0.6983	0.7444

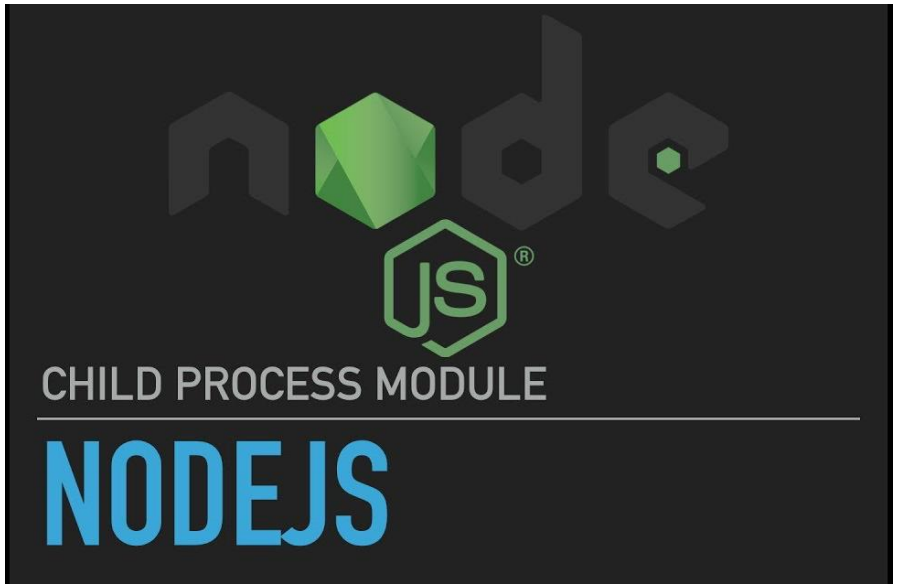


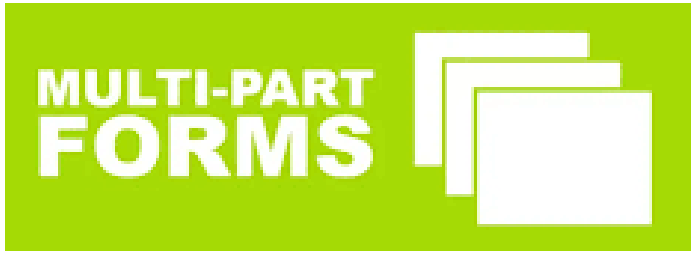
	1	30	31	59	60	89	90	119	120	149	150	179
1e-3, 5e-4	0.0252	0.4413	0.4479	0.5748	0.5758	0.6568	0.6567	0.6966				
5e-4, 1e-4								0.6966	0.6975	0.724	0.7476	0.774

Back-end Server



Callback function





JSON.stringify(result)

Router: /mobile_upload (model)
/upload (seller)
/update (seller)
/search (both)

Database

Database

MongoDB

photo collection

```
{
    "_id" : ObjectId("5b7e0591f01301339d26fc3b"),
    "path" : "files/1534985617051-photo.jpg",
    "caption" : "",
    "__v" : 0
}
```

user collection

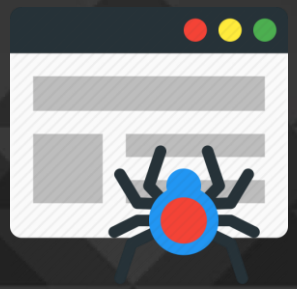
```
{
    "_id" : ObjectId("5b7e02573ccef53314ff3231"),
    "email" : "admin",
    "password" : "123",
    "dateCreated" : ISODate("2018-08-23T00:39:51.461Z")
}
```

products collection

```
{
    "_id" : ObjectId("5b8e0667e9d2fd3e77a562f7"),
    "image" : "https://shop.coles.com.au/wcsstore/Coles-CAS/imag
    "sellerList" : [
        {
            "name" : "Woolworth",
            "location" : "Rundle Mall",
            "price" : 11.9,
            "latitude" : 138.602599,
            "longitude" : -34.92243
        },
        {
            "name" : "Coles",
            "location" : "Rundle Mall",
            "price" : 12.0,
            "latitude" : 138.602862,
            "longitude" : -34.923067
        },
        {
            "name" : "COCOS",
            "location" : "Central Market",
            "price" : 12.0,
            "latitude" : 138.597483,
            "longitude" : -34.929362
        }
    ],
    "productName" : "Purina Total Care Undercoat Rake 1 pack ",
    "productsInfo" : "full of VC and VA"
}
```

embedded

Testing Data Collection



Crawler

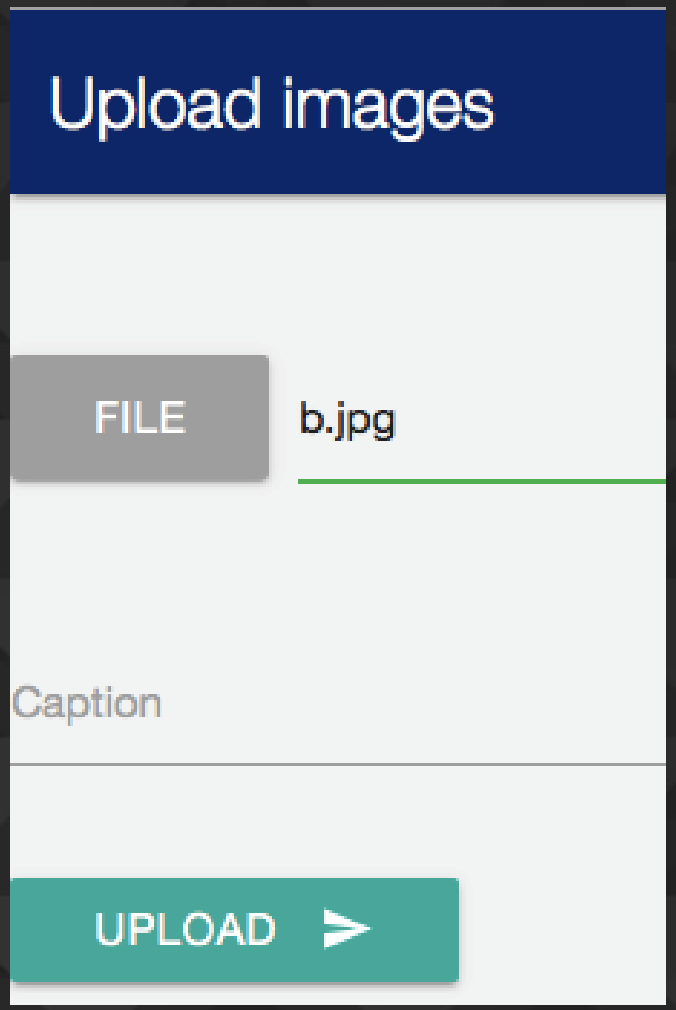
Enormous products

Image retrieval

Scrapy

Redis

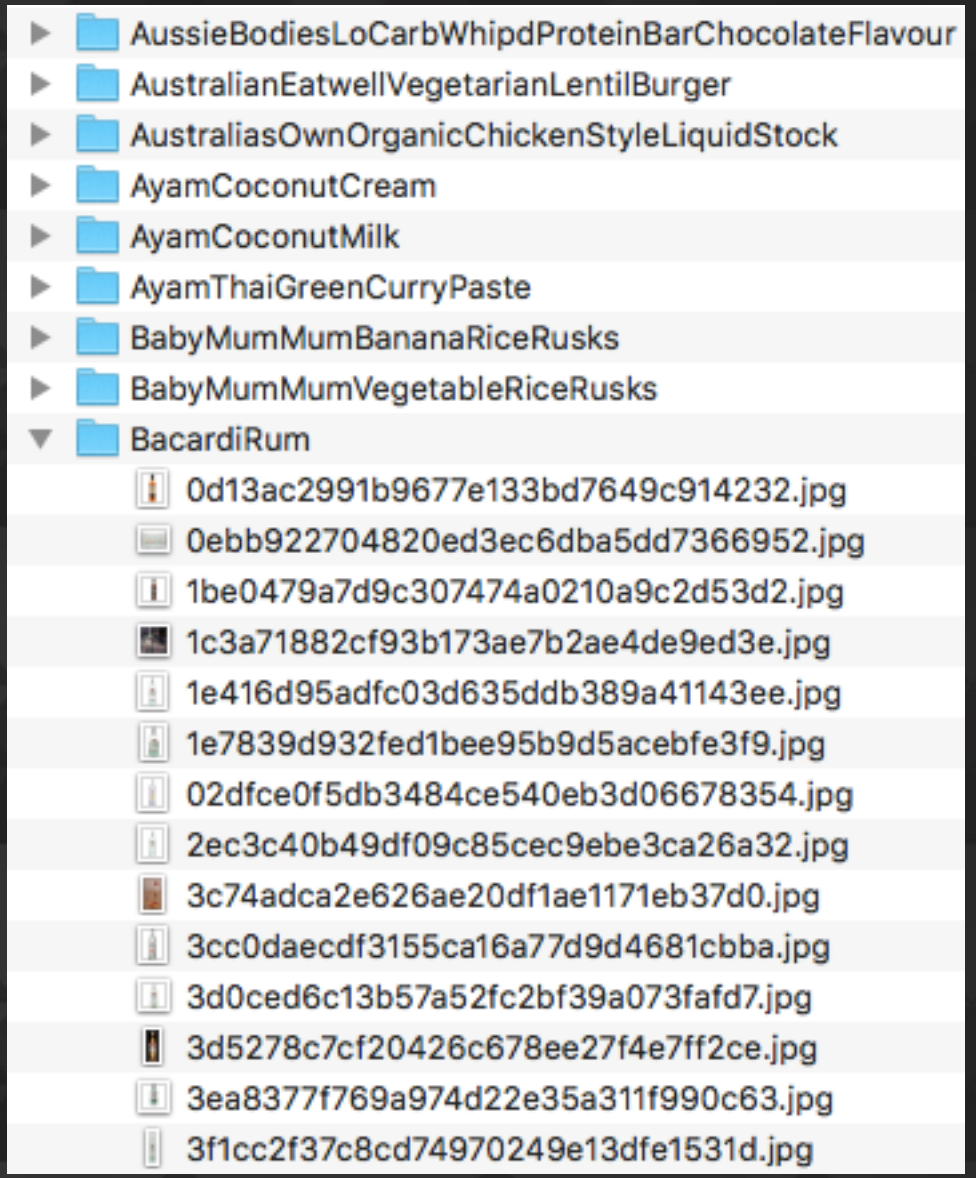
seller upload



Training Data Pretreatment

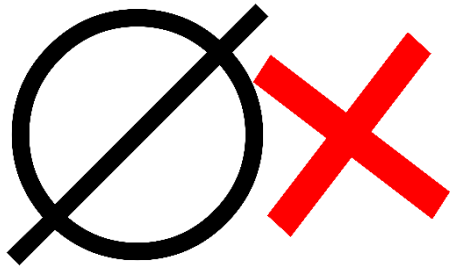
Google

- 90000+ images (noisy images)
- 1047 categories (narrow down each Browse categories)
- each category (100 images)
- 6 computers —> 4 days (stable)
- limitations on anti-crawl and quantity
- forged request header to deceive Google



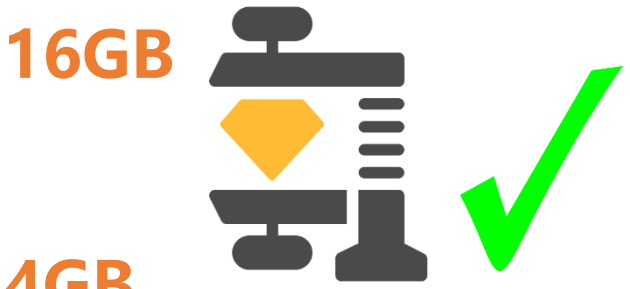
Data Pretreatment

imageDetection



empty dir

alpha channel

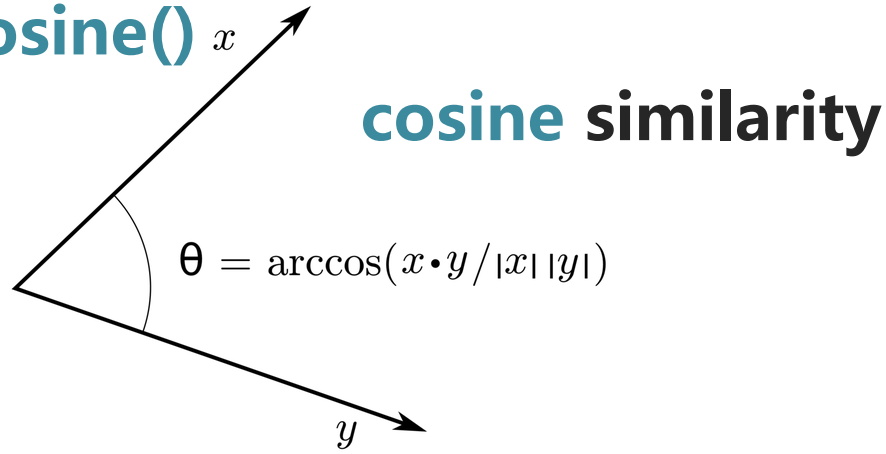


compression

Final Retrieval

nn.CosineSimilarity() >> spatial.distance.cosine()

How to speed up?



cosine similarity

Feature Matrix

10007*2048

Sort

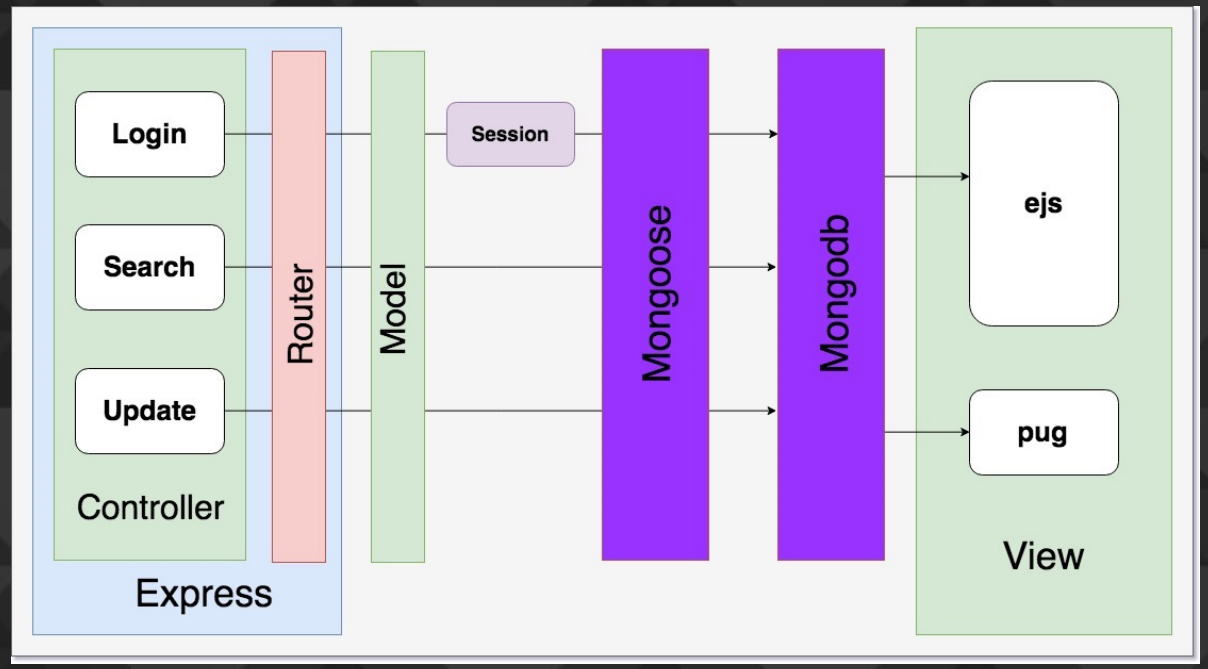
Index

pandas

images + classes

10007*2

Seller Update



Search1.ejs

↓

index.js

↓

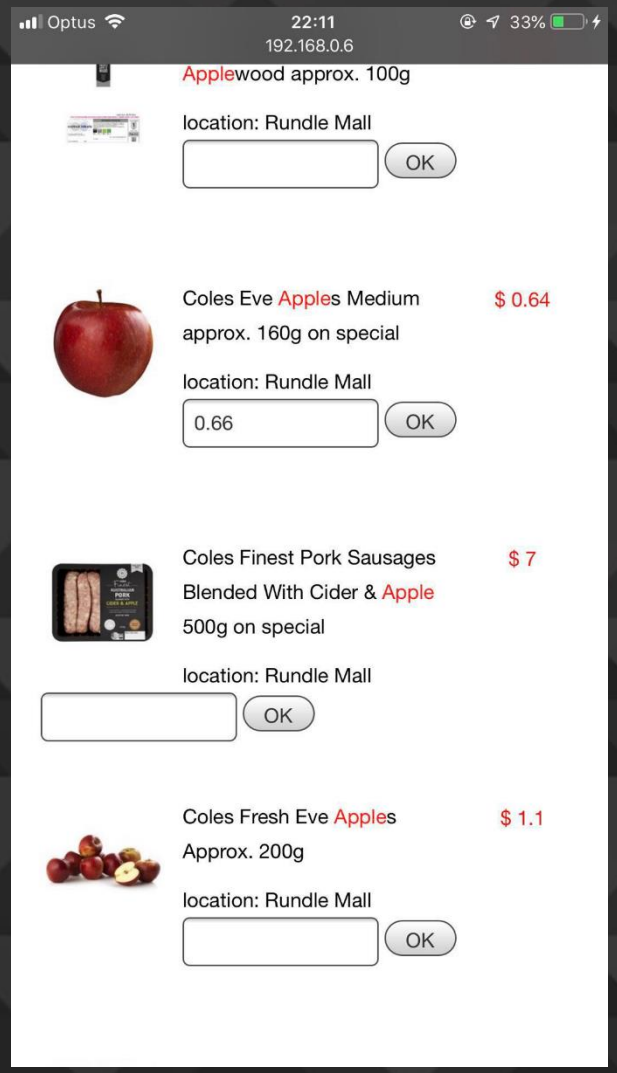
Mongodb

↓

Search1.ejs

Seller Update

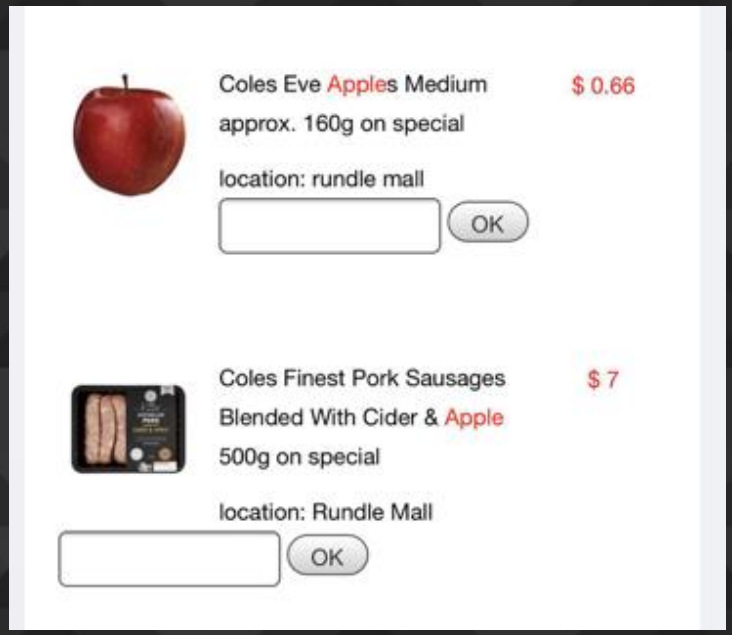
example on phone:



Mongodb Command:

```
Listening to port: 4000
Connected to mongoDB
CommandResult {
  result: { n: 1, nModified: 1, ok: 1 },
```

Output, price updated:



App Map Function

User Current Location

Could get the User's real-time location

Seller Location

Show the related Stores / Sellers location

Connected with Server

Get the location Information from the Server and show on the APP

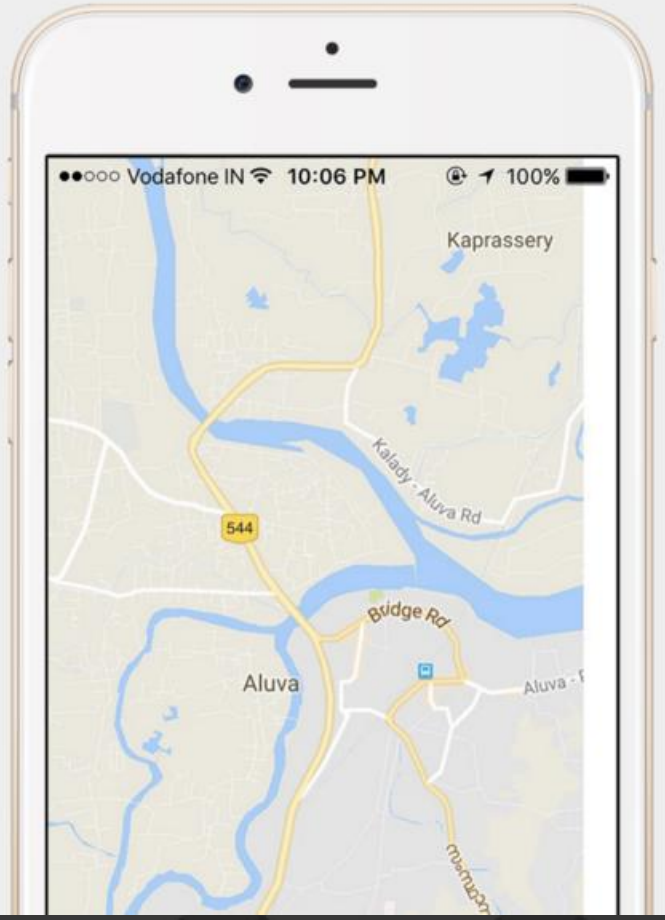
Routes direction

According to the given sellers location and the users location generate the recommendation route

Related function

Animating Map Movement
Zoom in/ Zoom out

APP-Map function



React Native + Google Maps

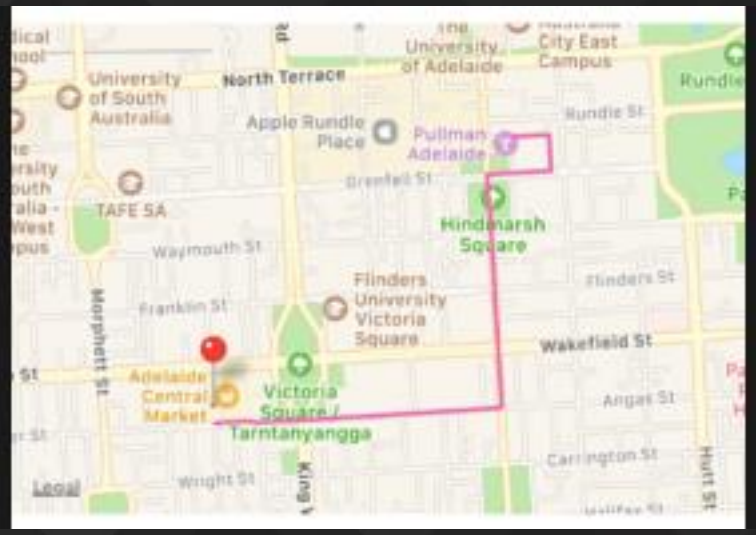
App Map Function

React-native-maps

```
import MapView from 'react-native-maps';
import MapViewDirections from 'react-native-maps-directions';
```

```
state = {
    focusedLocation:{
        latitude: -34.922844,
        longitude: 138.5997243,
        latitudeDelta: 0.0122,
        longitudeDelta:
            Dimensions.get("window").width /
            Dimensions.get("window").height *
            0.0112
    },
    locationChosen: false
}
```

```
const GOOGLE_MAPS_APIKEY = 'AIzaSyB4D1wmYG_vgcyy4F9xujx4qoTYUOs83C0';
```

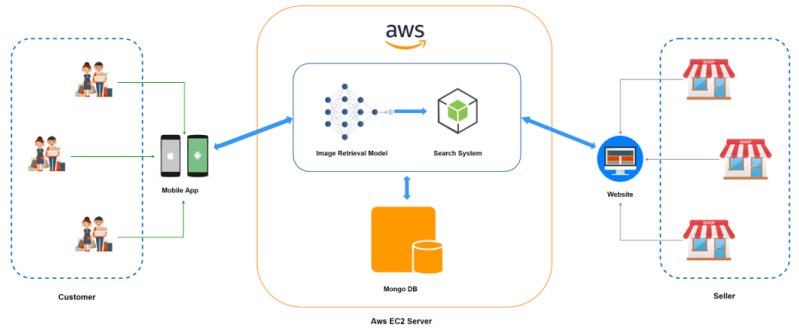



```
<MapView
    initialRegion={this.state.focusedLocation}
    // region={this.state.focusedLocation}
    style={styles.map}
    onPress={this.pickLocationHandler}
    ref={ref => this.map = ref}
    >
    {marker}
    <MapViewDirections
        origin={this.state.focusedLocation}
        destination={destination}
        apikey={GOOGLE_MAPS_APIKEY}
        strokeWidth={3}
        strokeColor="hotpink"
    />
</MapView>
```

App Progress - Demo

Summary

Platform structure

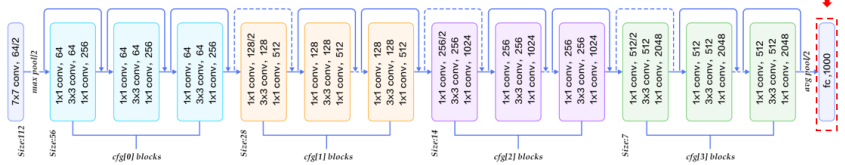


a1728768 4

Solution – Finetune & KNN

Resnet 152

Remove last layer



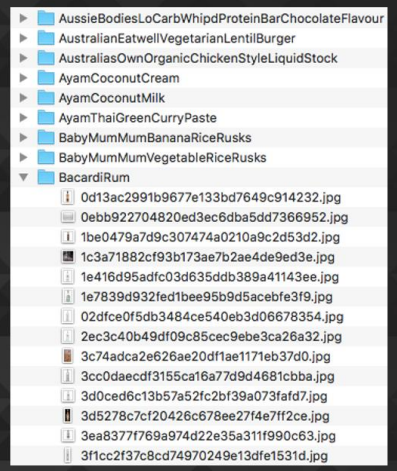
Extract feature + One nearest neighbor + Top 3

a1728768 10

Training Data Pretreatment

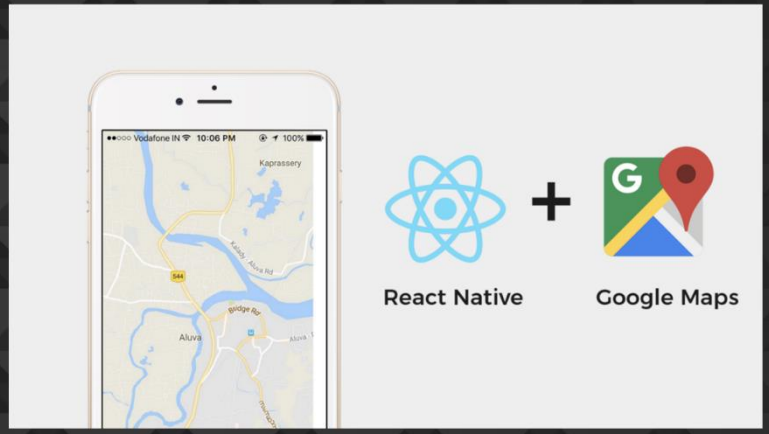
Google

- 90000+ images (noisy images)
- 1047 categories (narrow down each Browse categories)
- each category (100 images)
- 6 computers —> 4 days (stable)
- limitations on anti-crawl and quantity
- forged request header to deceive Google



a1699631 15

APP-Map function



a1710640 21