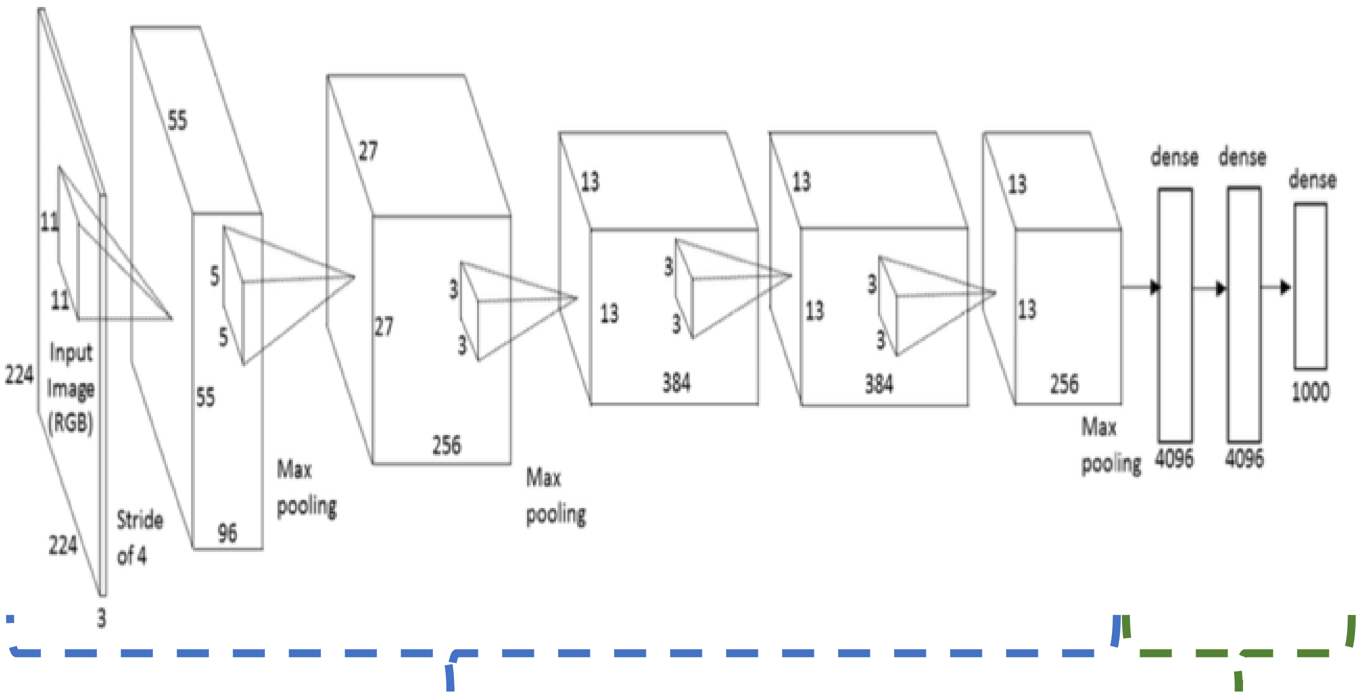
Convolutional Neural Network Classifier of Instagram City Photos

Agenda

- What I wanted to do -> Transfer Learning
- Why this dataset?
- InstaCities1M
- Architecture of the experiments
- Experiment 1 – small data
- Experiment 2 – bigger data
  - Original Experiment
  - Top 100 experiment
  - Random 100 experiment
- Experiment 2 – deep dive
  - Lowest probability correctly classified images
  - Error analysis
    - Original experiment
    - Top 100 experiment
  - Landmark analysis
  - Accuracy curve for degraded training data
  - Repeated experiments
  - Errors explanation – LIME
- Ideas for further research

What I wanted to do

Transfer Learning

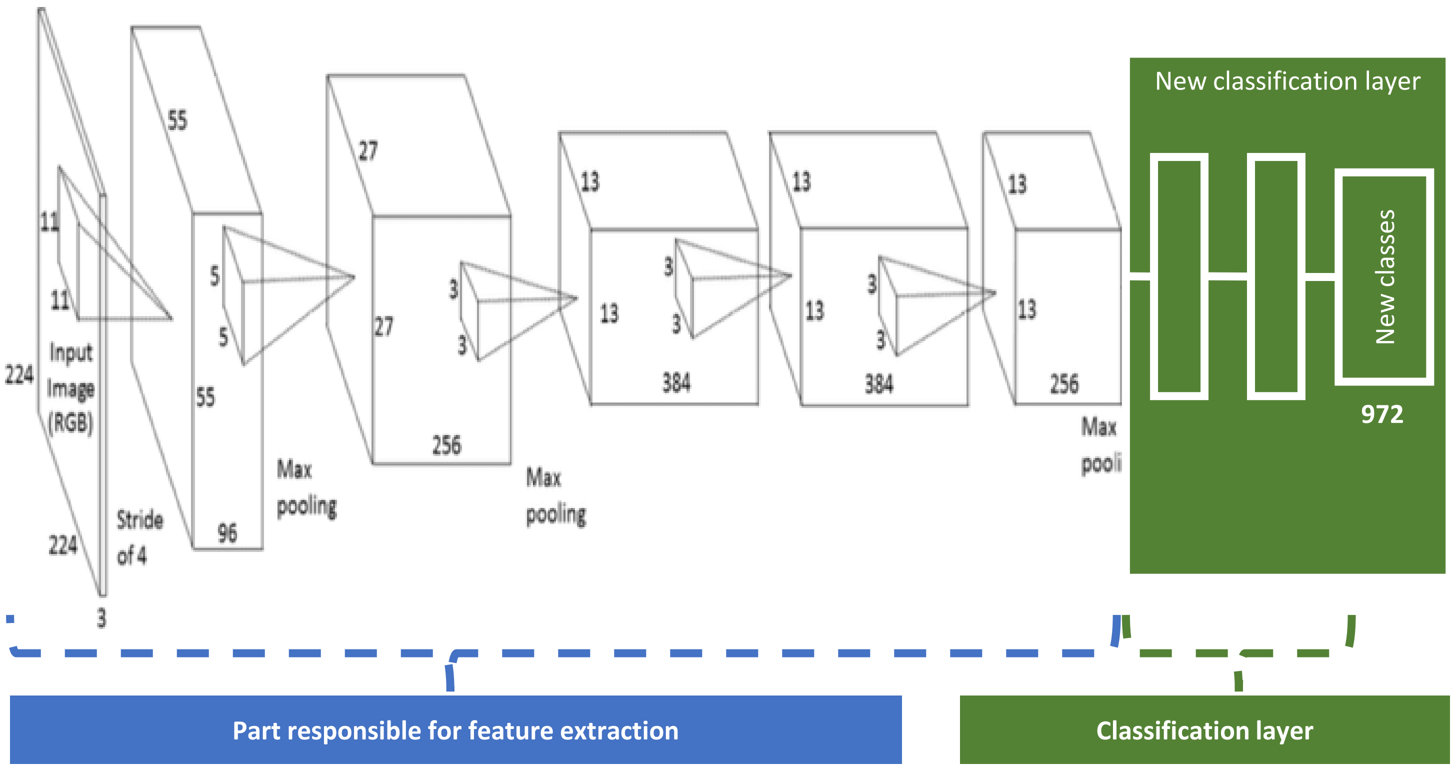


Part responsible for feature extraction
At initial layers the net extracts features like edges and colors. The deeper into the net we will go the more high level feature will be extracted. Examples of high level features: faces, wheels or text.

Categorization layer
This is a problem specific layer that uses the features extracted earlier to solve current problem.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural network. In Advances in Neural Information Processing Systems, 2012.

Transfer Learning

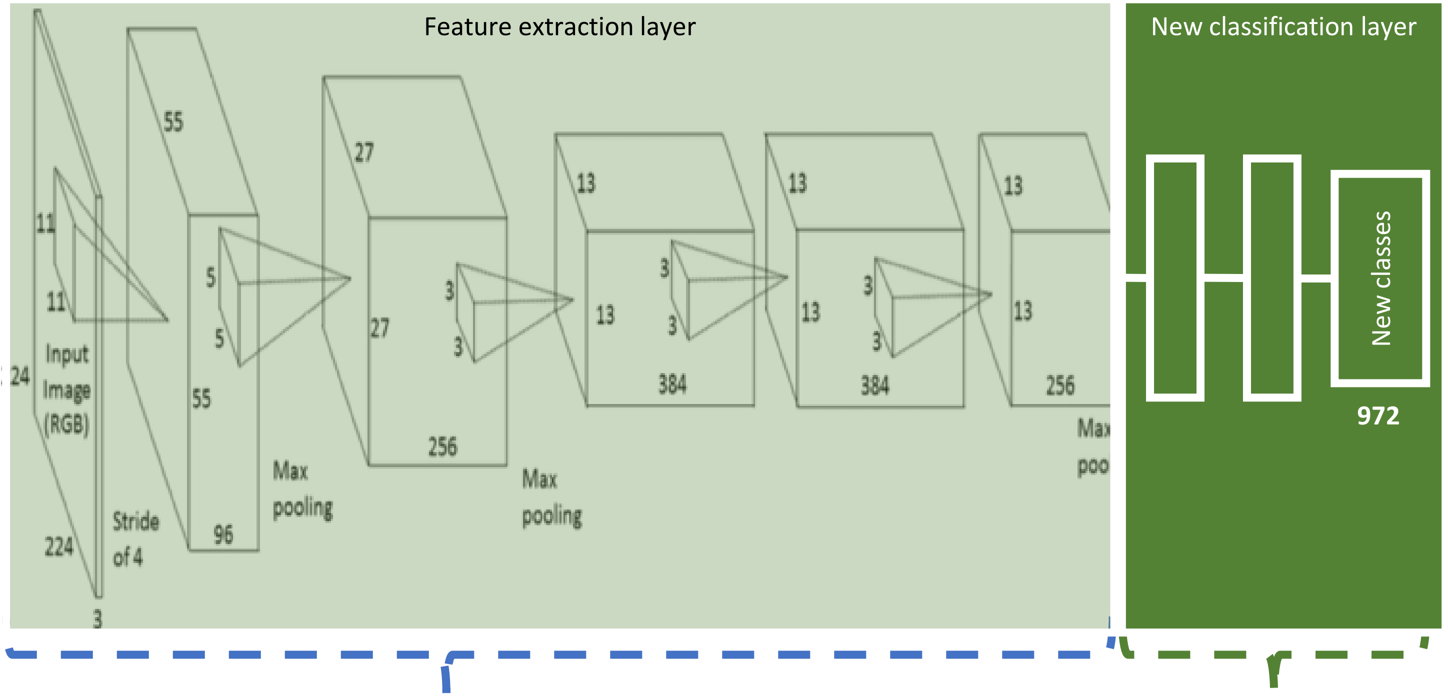


- Using information extracted from millions of images
- Possibility to fine-tune this part to extract specific features connected to problem at hand

- Retrained for the problem at hand

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural network. In Advances in Neural Information Processing Systems, 2012.

Transfer Learning



Part responsible for feature extraction

- Using information extracted from millions of images
- Possibility to fine-tune this part to extract specific features connected to problem at hand

Classification layer

- Retrained for the problem at hand

Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural network. In Advances in Neural Information Processing Systems, 2012.

Why this dataset?

Datasets

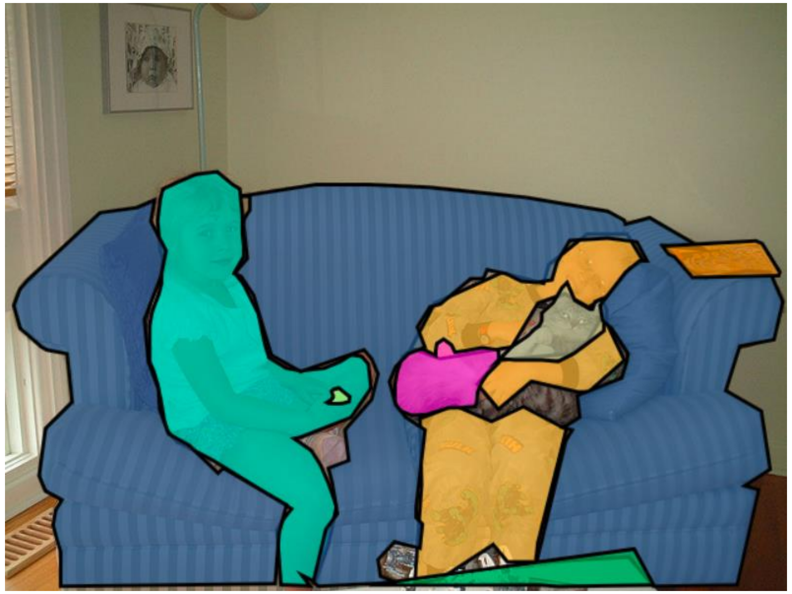
Object detection in images	scenario	#images	categories	avg. #labels/categories	resolution	occlusion labels	year
UIUC [12]	life	1,378	1	739	200×150		2004
INRIA [13]	life	2,273	1	1,774	96×160		2005
ETHZ Pedestrian [14]	life	2,293	1	10.9k	640×480		2007
TUD [15]	life	1,818	1	3,274	640×480		2008
EPFL Multi-View Car [16]	exhibition	2,000	1	2,000	376×250		2009
Caltech Pedestrian [1]	driving	249k	1	347k	640×480	√	2012
KITTI Detection [2]	driving	15.4k	2	80k	1241×376	√	2012
PASCAL VOC2012 [17]	life	22.5k	20	1,373	469×387	√	2012
ImageNet Object Detection [3]	life	456.2k	200	2,007	482×415	√	2013
MS COCO [4]	life	328.0k	91	27.5k	640×640		2014
VEDAI [18]	satellite	1.2k	9	733	1024×1024		2015
COWC [19]	aerial	32.7k	1	32.7k	2048×2048		2016
CARPK [10]	drone	1,448	1	89.8k	1280×720		2017
VisDrone2018	drone	10,209	10	54.2k	2000×1500	√	2018

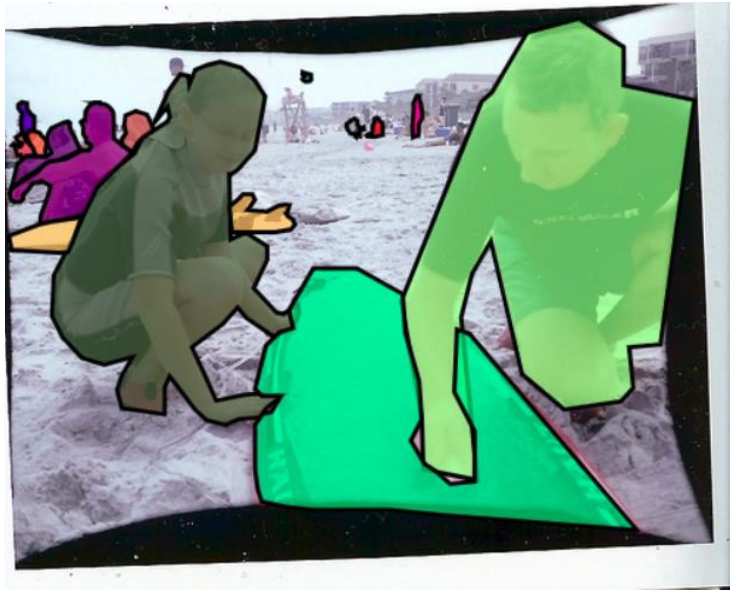
This set is for classification only

Database	Database - link
PIROPO	https://sites.google.com/site/piropodatabase/home
Pascal VOC	http://host.robots.ox.ac.uk/pascal/VOC/
Open Images Dataset V4	https://storage.googleapis.com/openimages/web/index.html
WebVision	https://www.vision.ee.ethz.ch/webvision/2017/download.html
IMAGENET	http://image-net.org/
COCO	http://cocodataset.org/#home
Caltech 256	http://www.vision.caltech.edu/Image_Datasets/Caltech256/intro/
Daimpler - pedestrians & (	http://www.gavrila.net/Datasets/Daimler_Pedestrian_Benchmark_D/daimler_pedestrian_benchmark_d.html
MIT datasets	
HRI Road Traffic dataset	http://www.gepperth.net/alexander/interests.html#carbenchmark
BELGA Logos	http://www-sop.inria.fr/members/Alexis.Joly/BelgaLogos/BelgaLogos.html
TopLogo-10 Dataset	http://www.eecs.qmul.ac.uk/~hs308/qmul_toplogo10.html/
WebLogo-2M Dataset	http://www.eecs.qmul.ac.uk/~hs308/WebLogo-2M.html/
TME Motorway Dataset	http://cmp.felk.cvut.cz/data/motorway/
CVL	http://www.vision.ee.ethz.ch/en/datasets/
Caltech Pedestrian Detect	http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/
Robust Multi-Person Trac	https://data.vision.ee.ethz.ch/cvl/aess/dataset/
CCPD: Chinese City Parkin	https://github.com/detectRecog/CCPD
VisDrone2018	http://aiskyeye.com/views/index
YFCC100M	http://multimediacommons.org/
InstaCities1M	https://gombru.github.io/2018/08/01/InstaCities1M/
Google Landmark	https://www.kaggle.com/google/google-landmarks-dataset

COCO – Common Object in Context

http://cocodataset.org/





	AP	AP50	AP75	APS	APM	APL	AR1	AR10	AR100	ARS	ARM	ARL	date
COCO Baseline	0.242	0.453	0.235	0.077	0.264	0.371	0.238	0.340	0.346	0.120	0.385	0.544	2016-03-06

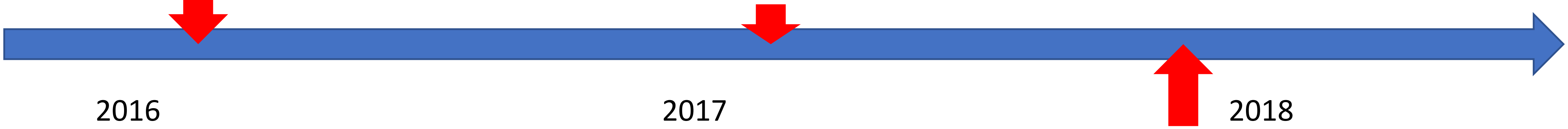
Team: Ross Girshick

Description: Faster R-CNN with end-to-end training (https://github.com/rbgirshick/py-faster-rcnn). This entry uses VGG16 pre-trained on ImageNet-1k and an RPN with 4 anchor scales (those in the NIPS paper plus 64^2).

	AP	AP50	AP75	APS	APM	APL	AR1	AR10	AR100	ARS	ARM	ARL	date
COCO VGG16 Baseline	0.297	0.504	0.313	0.128	0.325	0.421	0.272	0.399	0.409	0.187	0.451	0.591	2017-02-17

Team: Xinlei Chen, Abhinav Gupta

Description: Reimplementation of Faster R-CNN with end-to-end training (https://github.com/endernewton/tf-faster-rcnn). Single VGG16 model on conv5_3 with no extra parameters. Tested with 5000 top RoIs without using NMS for region selection.



	AP	AP^{50}	AP^{75}	AP^{S}	AP^{M}	AP^{L}	AR^{1}	AR^{10}	AR^{100}	AR^{S}	AR^{M}	AR^{L}	date
Megvii (Face++)	0.526	0.730	0.585	0.343	0.556	0.660	0.391	0.645	0.689	0.513	0.727	0.827	2017-10-05

Team: Chao Peng*, Tete Xiao*, Zeming Li*, Yuning Jiang, Xiangyu Zhang, Kai Jia, Gang Yu, Jian Sun (* indicates equal contribution); Megvii Research

Description: We trained a large-batch object detector, MegDet, by Megvii (Face++)'s large-scale deep learning framework called MegBrain, in parallel, on 128 GPUs. Thanks to MegBrain, we were able to finish the whole training within ~1 day and got improved performance due to the large batch size. The design of our detectors follows the idea of FPN[1], whose feature extractors are a series of ReNeXt-like models pre-trained on ImageNet only. GCN modules[2] and instance-blind segmentation supervision[3] were also applied in the detector. We did not use the unlabeled images provided by MSCOCO or other training data. On the test-dev, our best single detector had obtained mAP 50.5, and the ensemble of four detectors had achieved mAP 52.6. [1] Tsung-Yi Lin, Poitr Dollár, Ross Girshick, et al. Feature pyramid networks for object detection. CVPR 2017. [2] Chao Peng, Xiangyu Zhang, Gang Yu, et al. Large Kernel Matters--Improve Semantic Segmentation by Global Convolutional Network. CVPR 2017. [3] Jiayuan Mao, Tete Xiao, Yuning Jiang, et al. What Can Help Pedestrian Detection? CVPR 2017.

http://cocodataset.org/#detection-leaderboard

Other popular datasets

- IMAGENET
- Pascal VOC
- Open Images Dataset V4

All those are extremely popular and have 100+ articles.

TopLogo-10 Dataset

http://www.eecs.qmul.ac.uk/~hs308/qmul_toplogo10.html/

Size	Small (10 categories with 70 images each)
Challenge	Small training set
Application	Product categorization

Possible extensions -
WebLogo-2M Dataset

VisDrone2018

http://aiskyeye.com/views/index

Method	AP[%]	AP_{50}[%]	AP_{75}[%]	AR_{1}[%]	AR_{10}[%]	AR_{100}[%]	AR_{500}[%]
HAL-Retina-Net	**31.88**	46.18	**32.12**	0.97	7.50	34.43	**90.63**
DPNet	30.92	**54.62**	31.17	1.05	8.00	**36.80**	50.48
DE-FPN	27.10	48.72	26.58	0.90	6.97	33.58	40.57
CFE-SSDv2	26.48	47.30	26.08	1.16	**8.76**	33.85	38.94
RD^4MS	22.68	44.85	20.24	1.55	7.45	29.63	38.59
L-H RCNN+	21.34	40.28	20.42	1.08	7.81	28.56	35.41
Faster R-CNN2	21.34	40.18	20.31	1.36	7.47	28.86	37.97
RefineDet+	21.07	40.98	19.65	0.78	6.87	28.25	35.58
DDFPN	21.05	42.39	18.70	0.60	5.67	28.73	36.41
YOLOv3_DP	20.03	44.09	15.77	0.72	6.18	26.53	33.27
MFaster-RCNN	18.08	36.26	16.03	1.39	7.78	26.41	26.41
MSYOLO	16.89	34.75	14.30	0.93	5.98	23.01	26.35
DFS	16.73	31.80	15.83	0.27	2.97	26.48	36.26
FPN2	16.15	33.73	13.88	0.84	6.73	23.32	30.37
YOLOv3+	15.26	33.06	12.50	0.68	5.77	21.15	23.83
IITH DODO	14.04	27.94	12.67	0.82	5.86	21.02	29.00
FPN3	13.94	29.14	11.72	0.81	6.08	22.98	22.98
SODLSY	13.61	28.41	11.66	0.60	5.20	19.26	23.68
FPN*	13.36	27.05	11.81	0.77	5.65	20.54	25.77
FPN+	13.32	26.54	11.90	0.84	5.87	22.20	22.20
AHOD	12.77	26.37	10.93	0.56	4.36	17.49	18.87
DFP	12.58	25.13	11.43	0.88	6.20	19.63	21.27
YOLO-R-CNN	12.06	27.98	8.95	0.50	4.39	19.78	23.05
MMN	10.40	20.66	9.43	0.41	5.22	18.28	19.97
YOLOv3++	10.25	21.56	8.70	0.48	4.31	15.61	15.76
Faster R-CNN+	9.67	18.21	9.54	1.19	6.74	16.40	16.40
R-SSRN	9.49	21.74	7.29	0.36	3.27	17.07	21.63
JNU_Faster RCNN	8.72	15.56	8.98	1.02	6.20	12.18	12.18
SOD	8.27	20.02	5.80	0.39	3.78	14.12	17.19
Keras-Retina-Net	7.72	12.37	8.68	0.62	5.65	10.76	10.80
MMF	7.54	16.53	6.03	1.28	5.91	14.28	14.36
R-FCN*	7.20	15.17	6.38	0.88	5.35	12.04	13.95
RetinaNet2	5.21	10.02	4.94	0.38	3.54	11.55	14.25
CERTH-ODI	5.04	10.94	4.12	**1.65**	5.93	9.05	9.05
Faster R-CNN3	3.65	7.20	3.39	0.64	2.41	10.08	21.85
Faster R-CNN*	3.55	8.75	2.43	0.66	3.49	6.51	6.53
MSCNN	2.89	5.30	2.89	0.59	2.18	9.33	15.38
SSD*	2.52	4.78	2.47	0.58	2.81	4.51	6.41

Size	Moderate
Challenge	Drone
Application	Drone image analysis

InstaCities1M description

InstaCities1M

https://gombru.github.io/2018/08/01/InstaCities1M/

Size	Big
Challenge	Web data – very noisy
Application	

This dataset was used for learning common embedding for image description and images. Task: image retrieval based on text search.

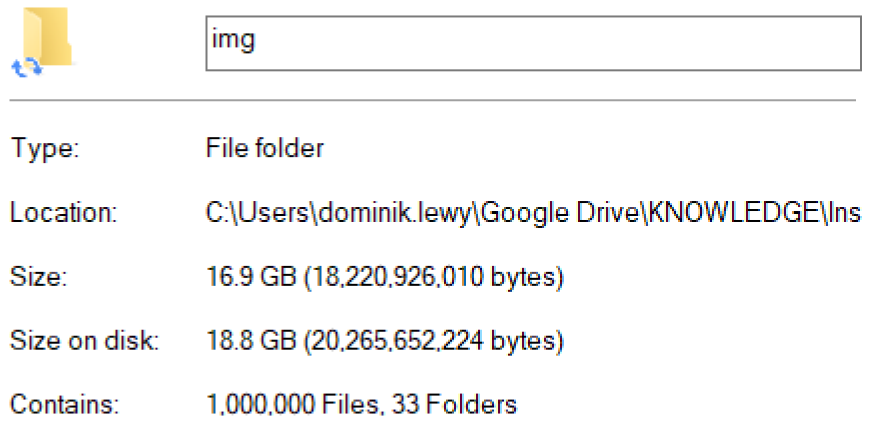
Possible extensions - Google Landmark ??

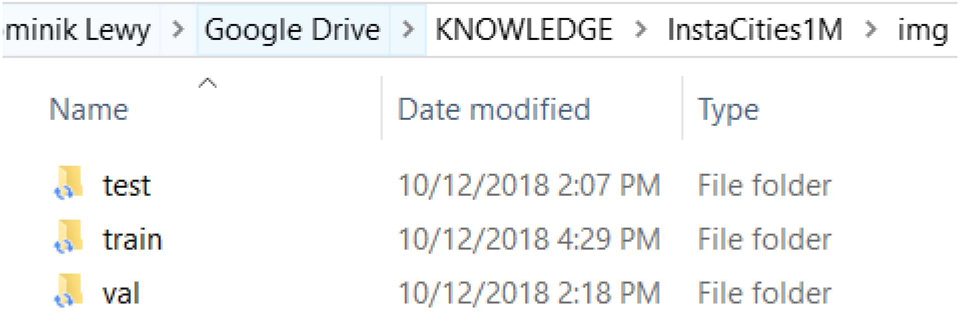
InstaCities1M – exploration

Exemplary images for London category (first 8 from the train folder):

InstaCities1M – facts

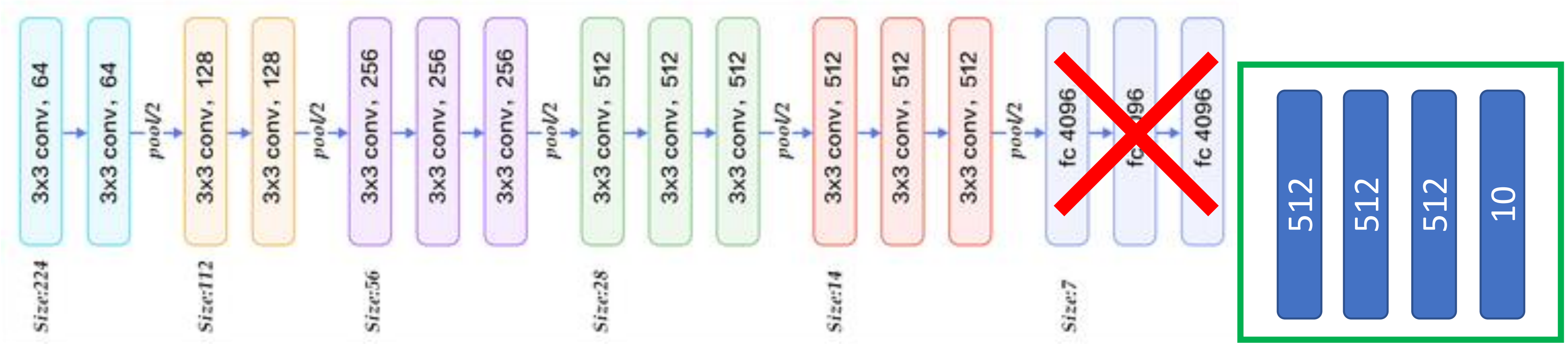
- The size of the data set: 17GB (1M of images from 10 categories)
- The data is already divided into:
  - Train – 800k images
  - Validation – 50k images
  - Test – 150k images





Architecture

Architecture 2 – VGG16



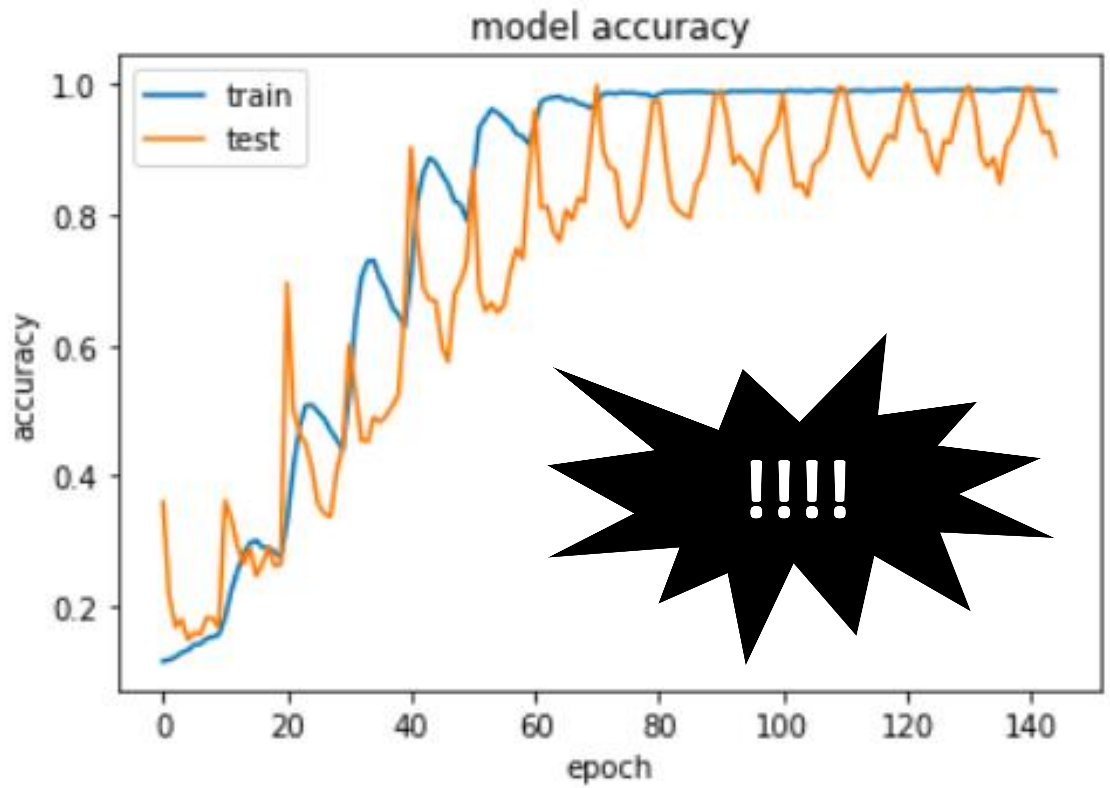
Original network had 3 FC layers at the end: 2 of those had 4096 neurons and the third last layer had as much layers as the number of categories.

My net has 4 FC layers at the end: 3 of those have 512 neurons and the last has 10 as the number of cities in the dataset.

Problem 1	Not enough memory to train network
Solution	Decreasing the batch size from 16 images to 4 at once. Decreasing the size of FC layers from 4096 to 512.
Problem 2	Net not learning to predict which city was the picture taken in
Solution	Gradually decreasing the learning rate which was finally set at 0.0001.

Experiments 1 – small data

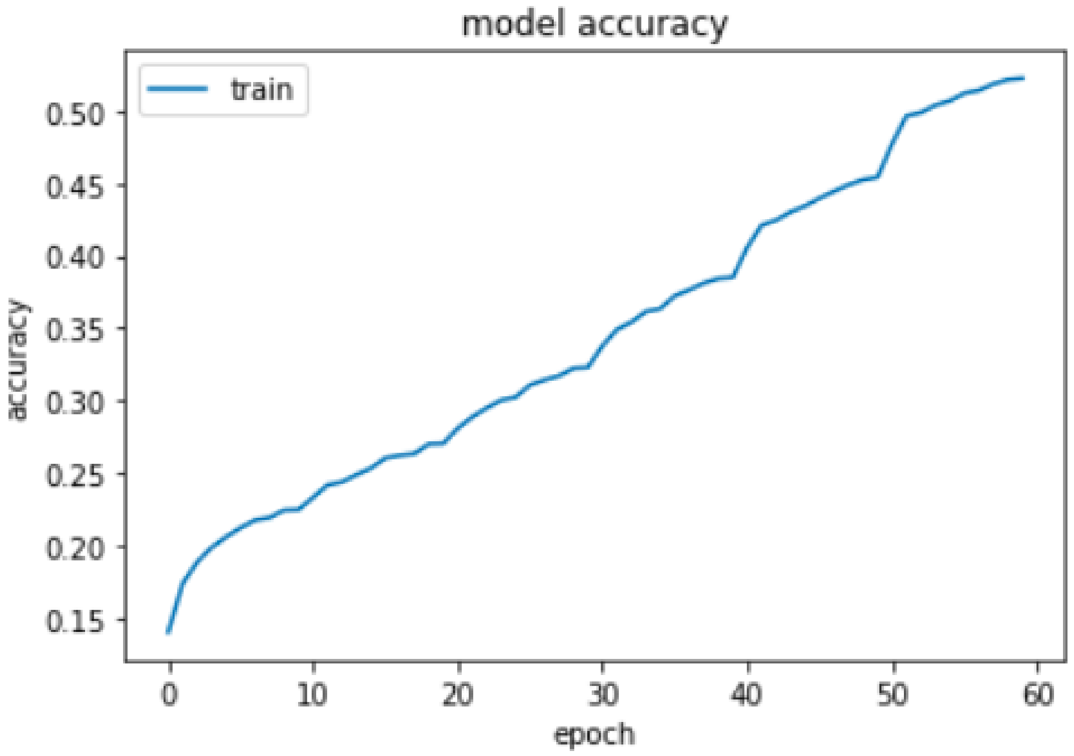
Experiment – Architecture 2 (VGG16) – small data



- The results on the right are for small data (100K images)
- Train – full 100k data
- Test – only **10k first observations from training data!!!**

Result	I was able to train the model to predict each observation from training data with 95%+ accuracy
Observations/questions	• We could probably achieve 100% accuracy by further training at lower learning rate • What is the best approach here, when to stop training? (Normally we have validation set – but now as everything is in train)

Experiment – Architecture 2 (VGG16) – big data



- This is the state of training for 22.10 6:31 pm.
- The model have seen the entire data 6 times
- The accuracy gain seems to be linear except for first few epochs

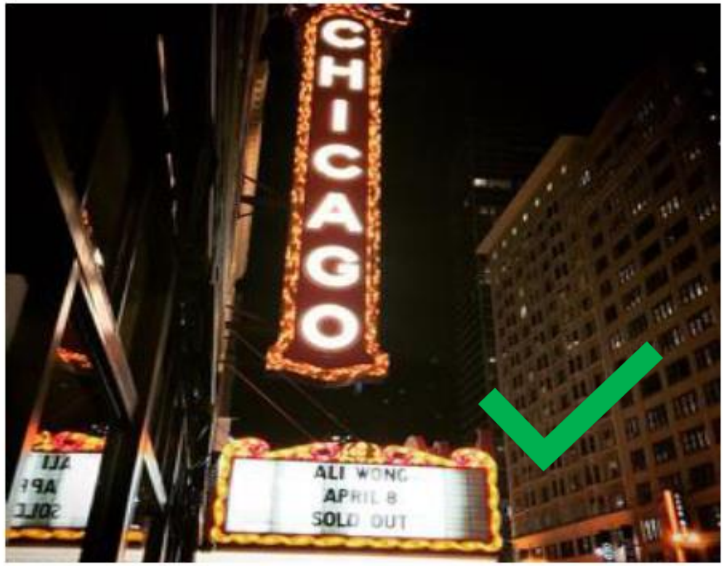
Result	Training in progress, the results keep improving the longer we train the model. Time already elapsed: ~60h (each epoch takes around 55 minutes)
Observations/questions	• When to stop training?

Experiment – Architecture 2 (VGG16) – small data – deep dive

- Is a baseball team from Denver which could play matches in either of the USA cities

- Hard to tell from food photo

Experiment – Architecture 2 (VGG16) – small data – deep dive



- This is the Chicago Theater - correct

- Hard to tell from food photo

Experiment – Architecture 2 (VGG16) – small data – deep dive

- Hard to tell from this photo

- This is the Millennium Park in Chicago

Experiment – Architecture 2 (VGG16) – small data – deep dive

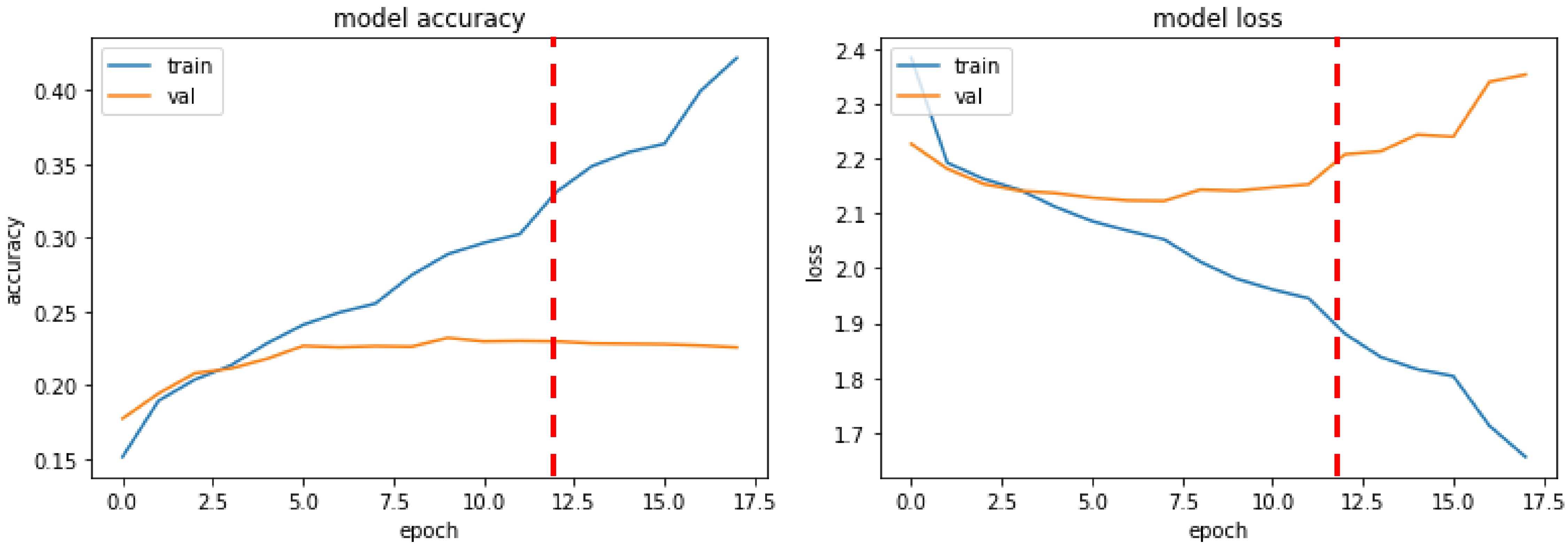
- Hard to tell from this photo

- Normally Starry Night is in NY but it could be on trip in Chicago?

Experiments 2 – bigger data

Original Experiment

Experiment 2 – VGG16 – training on one set validation and early stopping on other



- Train – full 800k data
- Val – full 50k data
- The split was done according to author proposition

- The training was conducted for 17 epochs (200k images each) so the net saw the training data 4 times
- For inference net after 12 epochs was used as after that the validation accuracy and error started decreasing

Experiment 2 – top 100 predictions for each class

```
## check if all top 100 predictions are correct
print("### CORRECT ###")
for i in range(0,10):
    temp = select_top_n_from_city(data, 100, str(i))
    print(dict_num_cities[i],": ",np.sum(temp["correct"]))
```

```
### CORRECT ###
chicago :  100
london :  100
losangeles :  99
melbourne :  99
miami :  98
newyork :  100
sanfrancisco :  100
singapore :  100
sydney :  99
toronto :  100
```

- Vast majority of the top 100 predictions for each class were correct predictions

Experiment 2 – per class deep dive – LONDON

What can be seen on photos:

- London Eye
- Tower Bridge
- Elizabeth Tower (Big Ben)

Experiment 2 – per class deep dive – LOS ANGELES

What can be seen on photos:

- Palm trees

Experiment 2 – per class deep dive – MELBOURNE

What can be seen on photos:

- Flinders Street Station

Experiment 2 – per class deep dive – NEW YORK

What can be seen on photos:

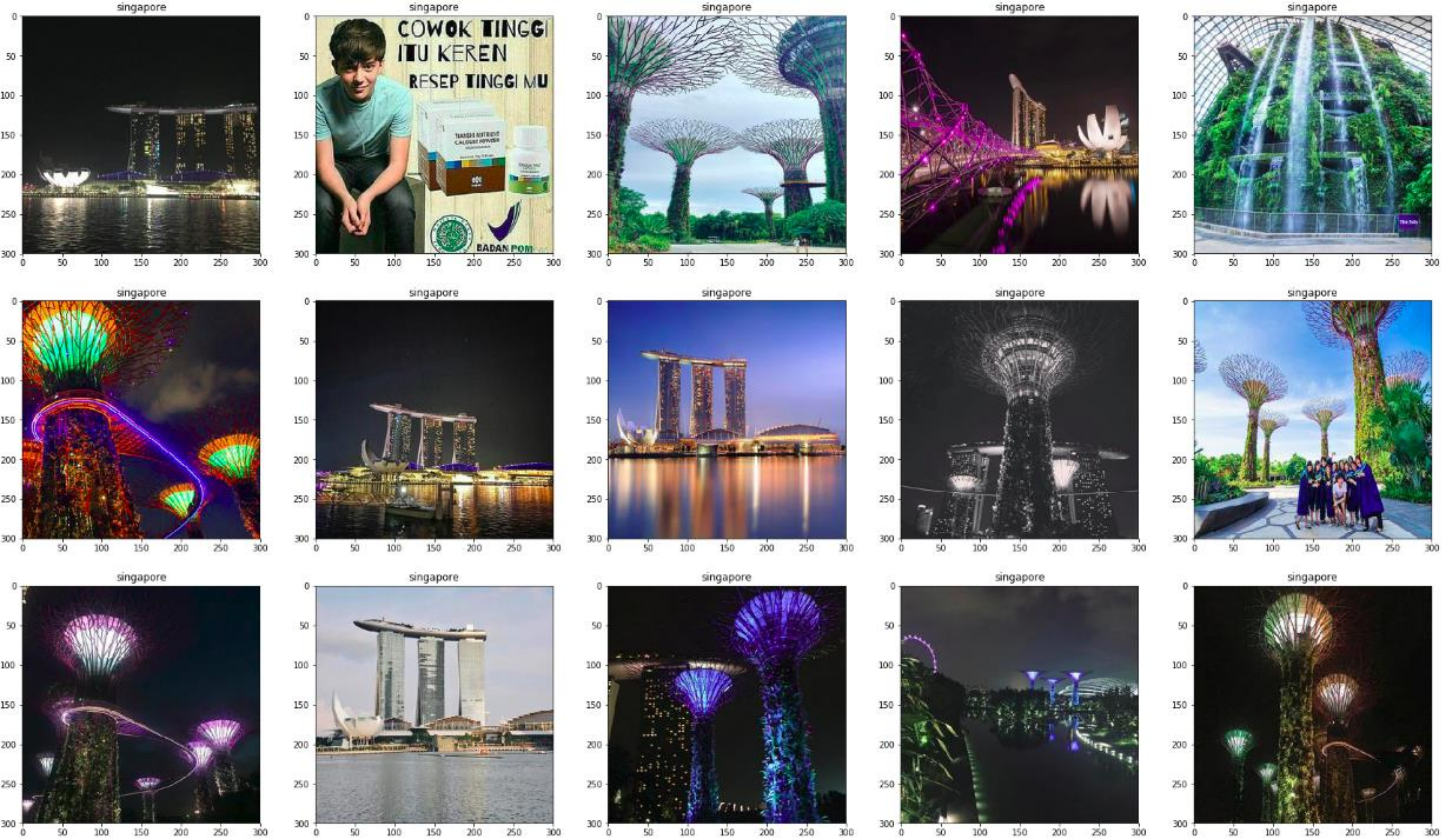
- Times Square
- Brooklyn Bridge
- Empire State Building

Experiment 2 – per class deep dive – SAN FRANCISCO

What can be seen on photos:

- Golden Gate Bridge

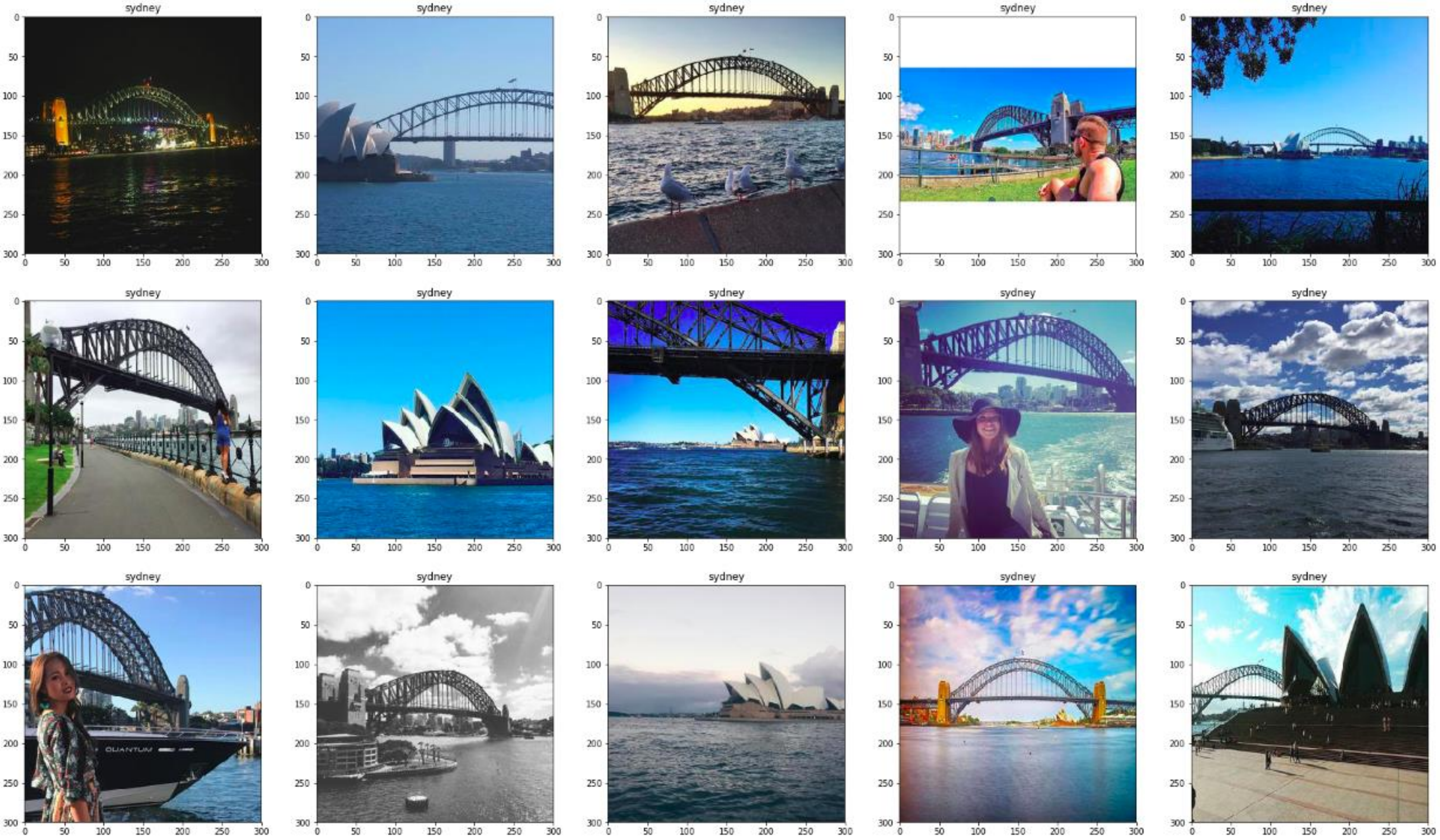
Experiment 2 – per class deep dive – SINGAPORE



What can be seen on photos:

- Marina Bay Sands
- Gardens by the bay
- Leaflet not in English

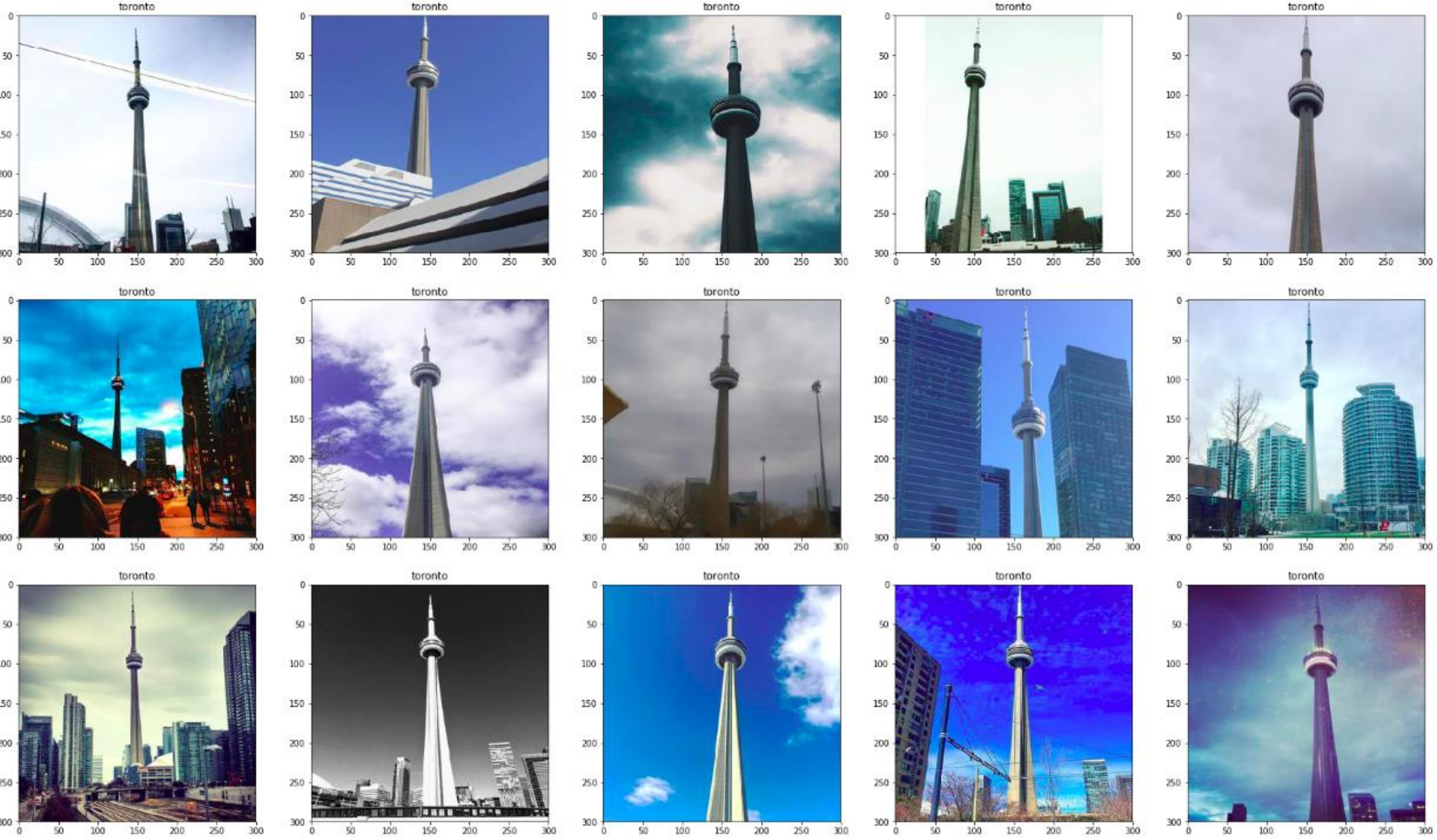
Experiment 2 – per class deep dive – SYDNEY



What can be seen on photos:

- Sydney Harbor Bridge
- Sydney Opera

Experiment 2 – per class deep dive – TORONTO



What can be seen on photos:

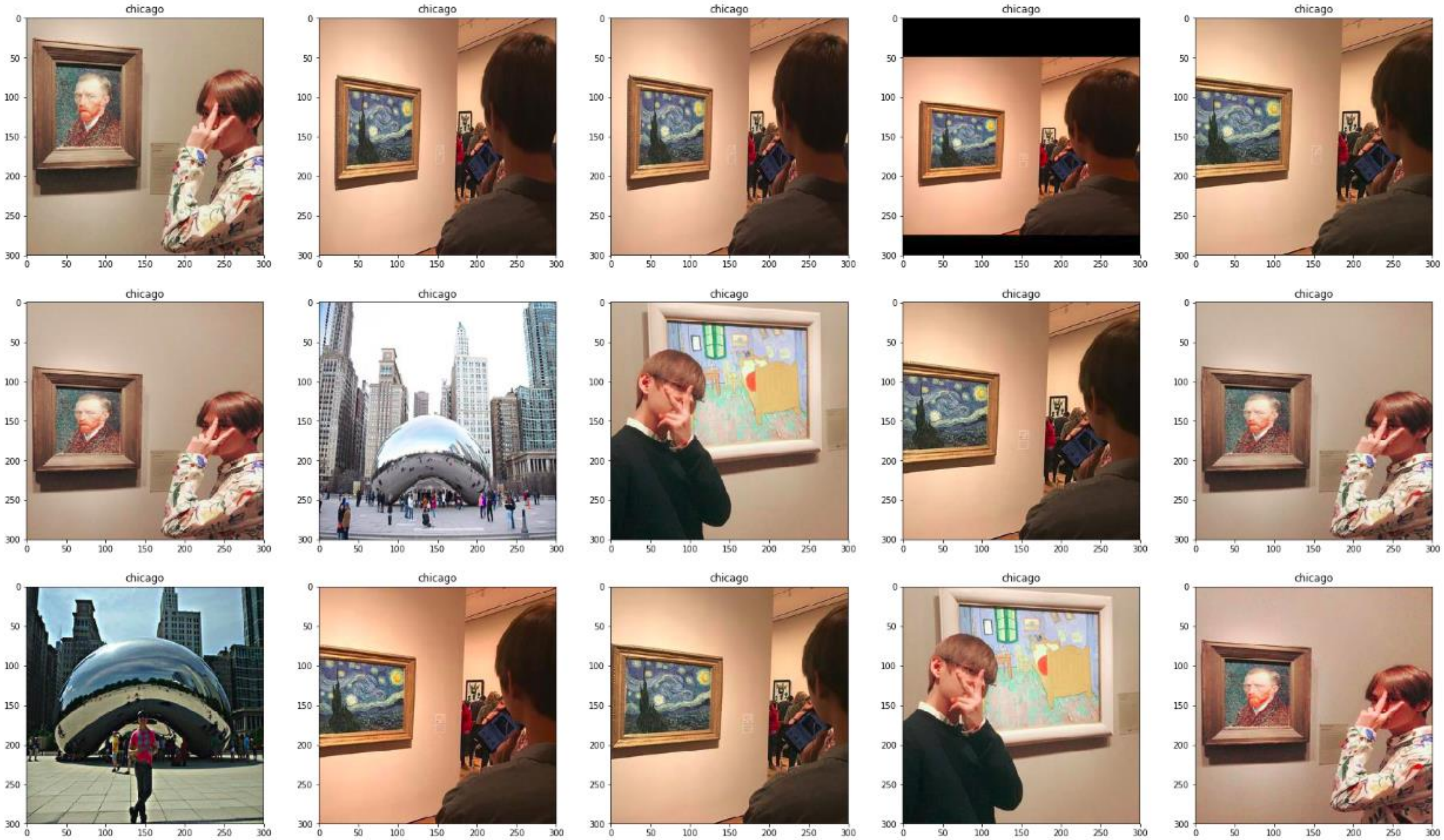
- Canada's National Tower (CN Tower)

Experiment 2 – per class deep dive – MIAMI

What can be seen on photos:

- https://www.miamimusicweek.com/artist/alesso
- A music event was taking place in Miami with the main star Alesso

Experiment 2 – per class deep dive – CHICAGO

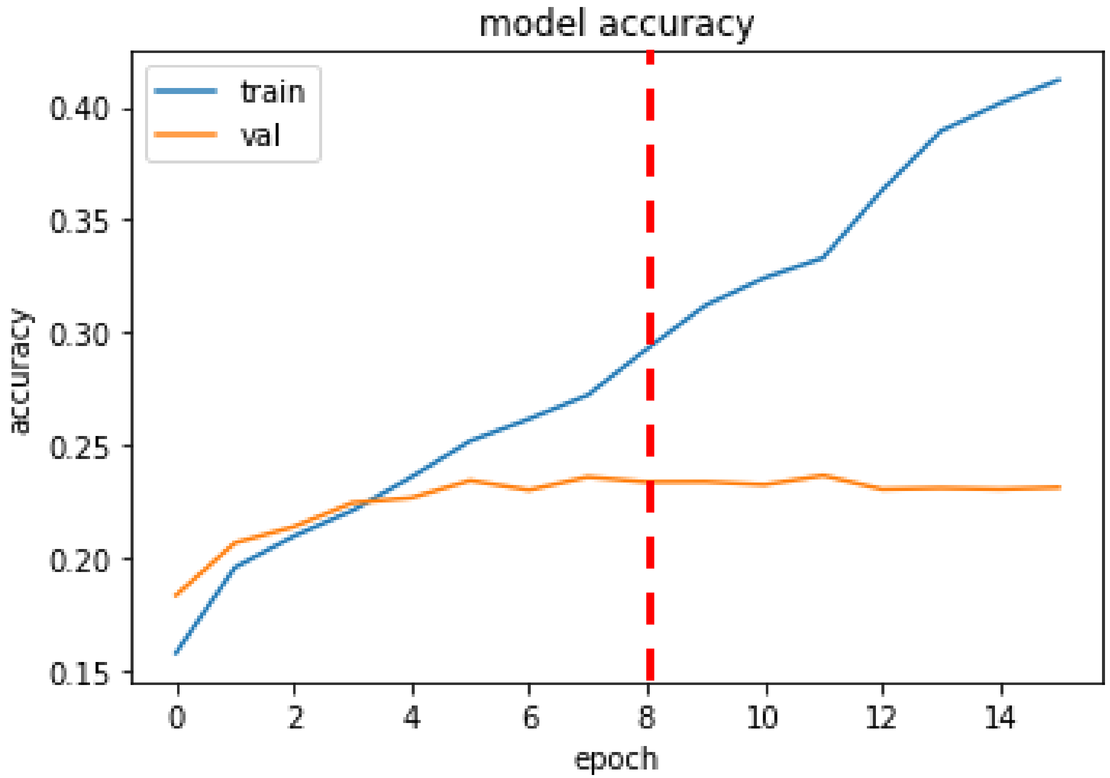


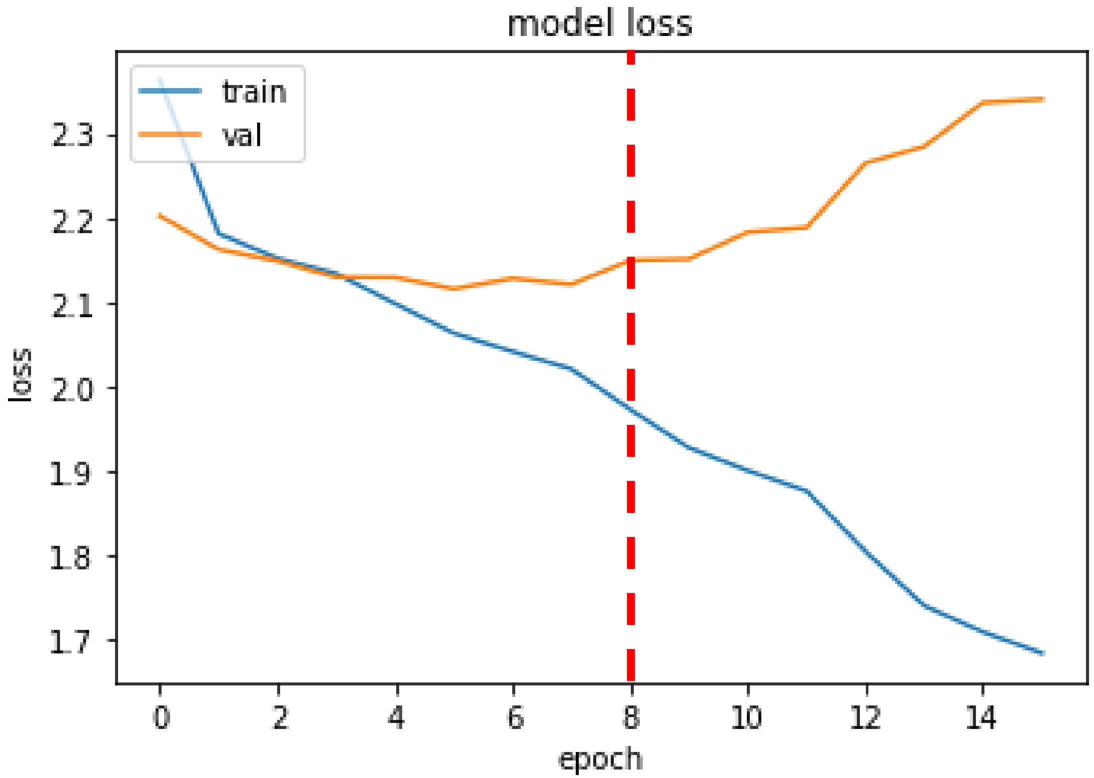
What can be seen on photos:

- Van Gogh autoportret
- Starry Night
- Millennium Park

Top 100 experiment

Experiment 3 – VGG16 – training on same data minus top 100 (same set for validation)





- Train – full 799 data
- Val – full 50k data
- Test – top 100 from each class (1k images not used for training)

- For inference net after 8 epochs was used as after that the validation accuracy started decreasing and error increased

Experiment 3 – VGG16 – training on same data minus top 100 (same set for validation)

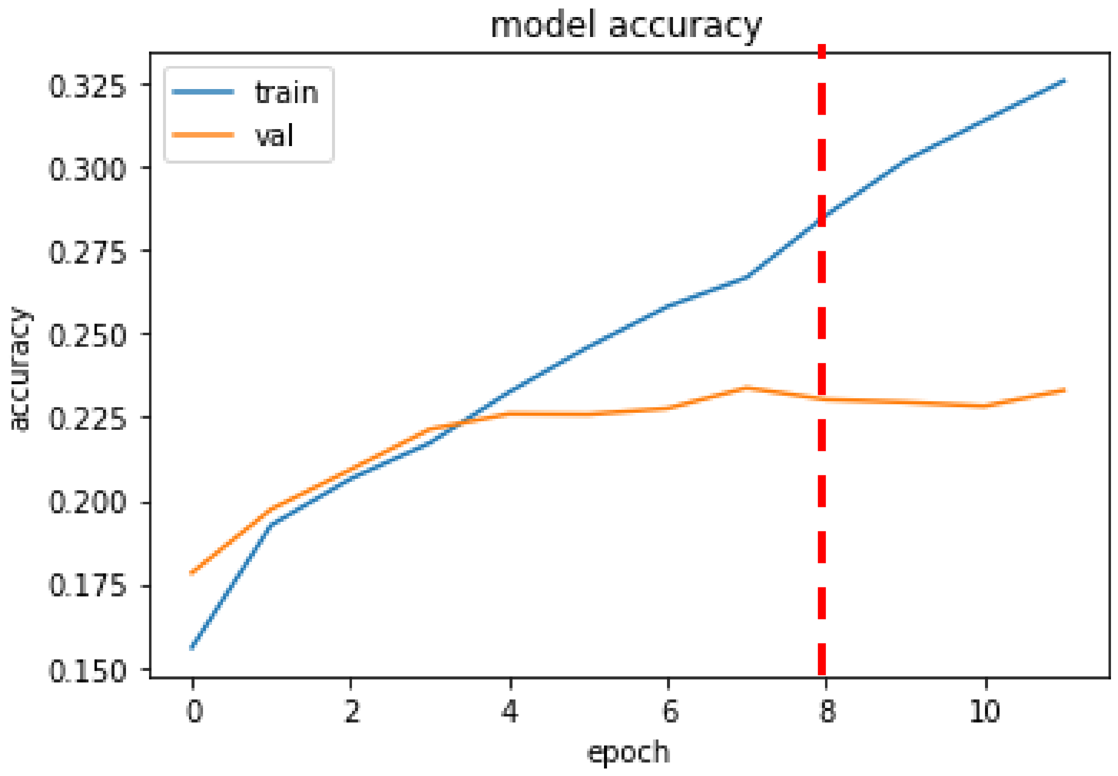
Results:

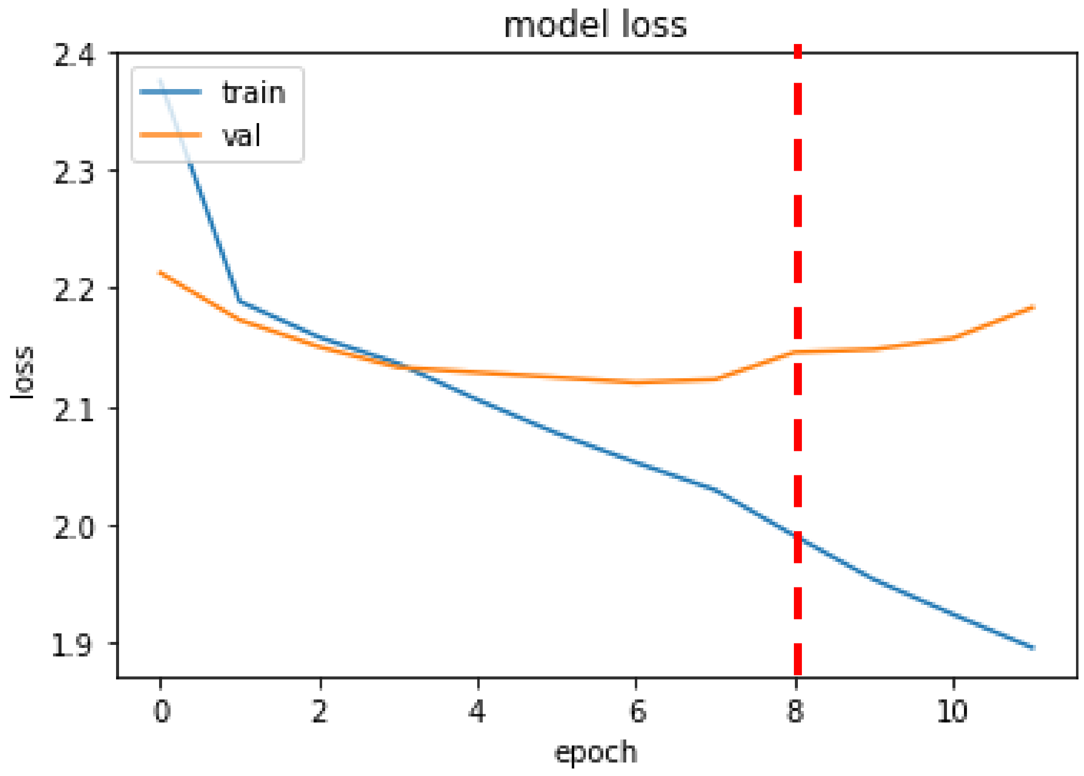
```
test_accuracy = np.mean(test_pred==true_test_y)

print("\nTest accuracy: {} %".format(test_accuracy*100))
```

```
Test accuracy: 95.5 %
```

Random 100 experiment

Experiment 4 – VGG16 – training on same data minus random 100 (same set for validation)





- Train – full 799 data
- Val – full 50k data
- Test – random 100 from each class (1k images not used for training)

- For inference net after 8 epochs was used as after that the validation accuracy started decreasing and error increased

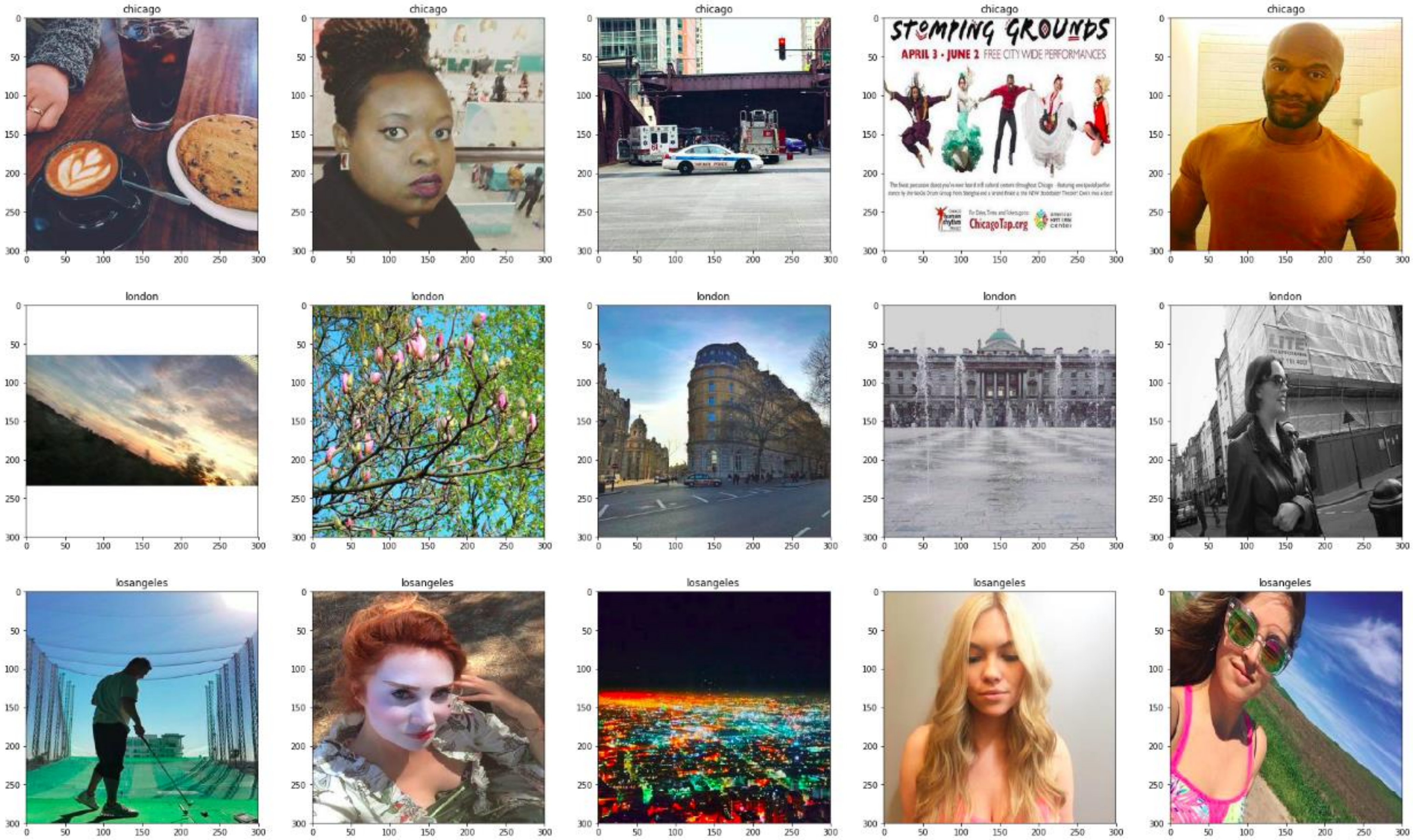
Experiment 4 – VGG16 – training on same data minus random 100 (same set for validation)

Results:

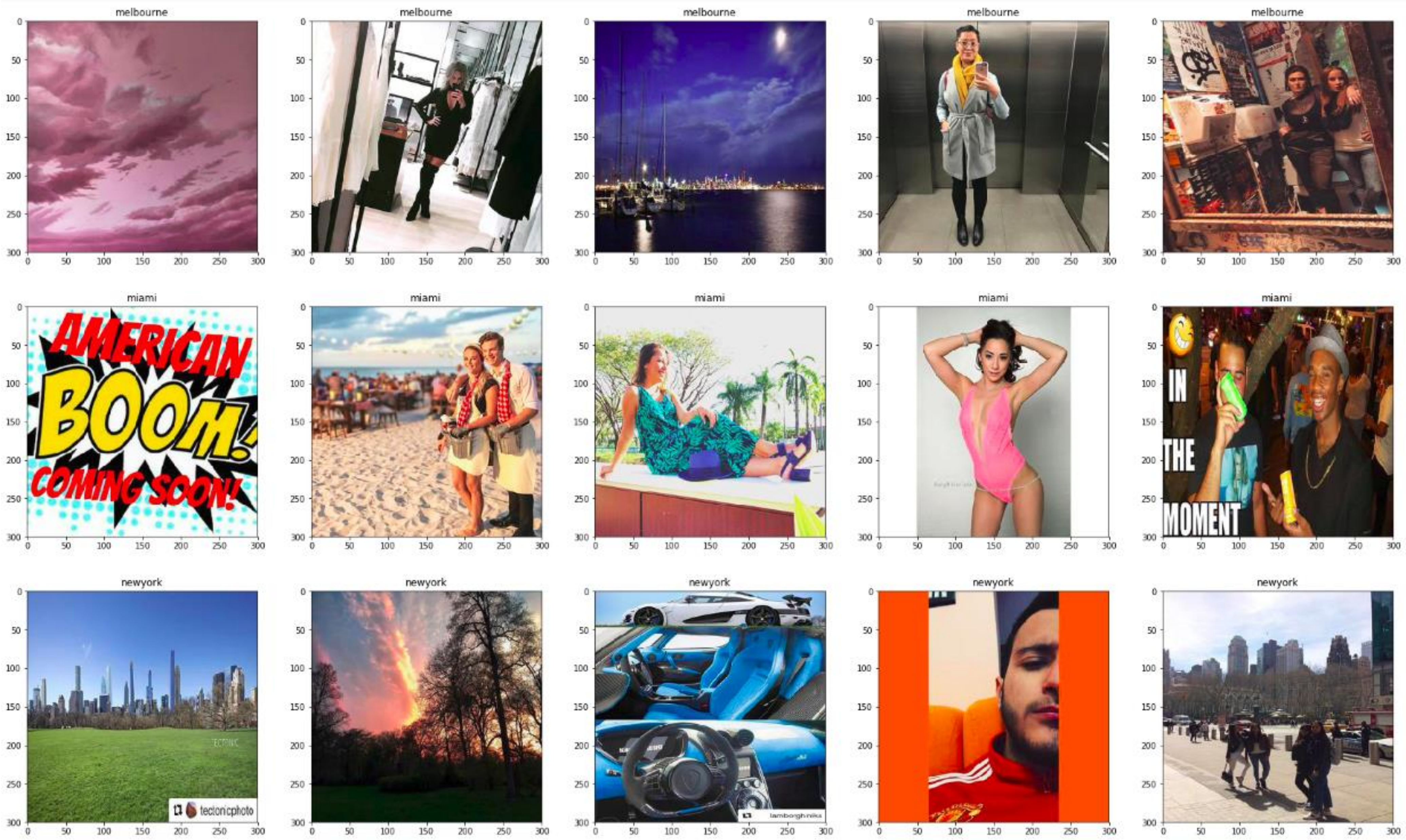
```
test_accuracy = np.mean(test_pred==true_test_y)

print("\nTest accuracy: {} %".format(test_accuracy*100))
```

```
Test accuracy: 44.4 %
```

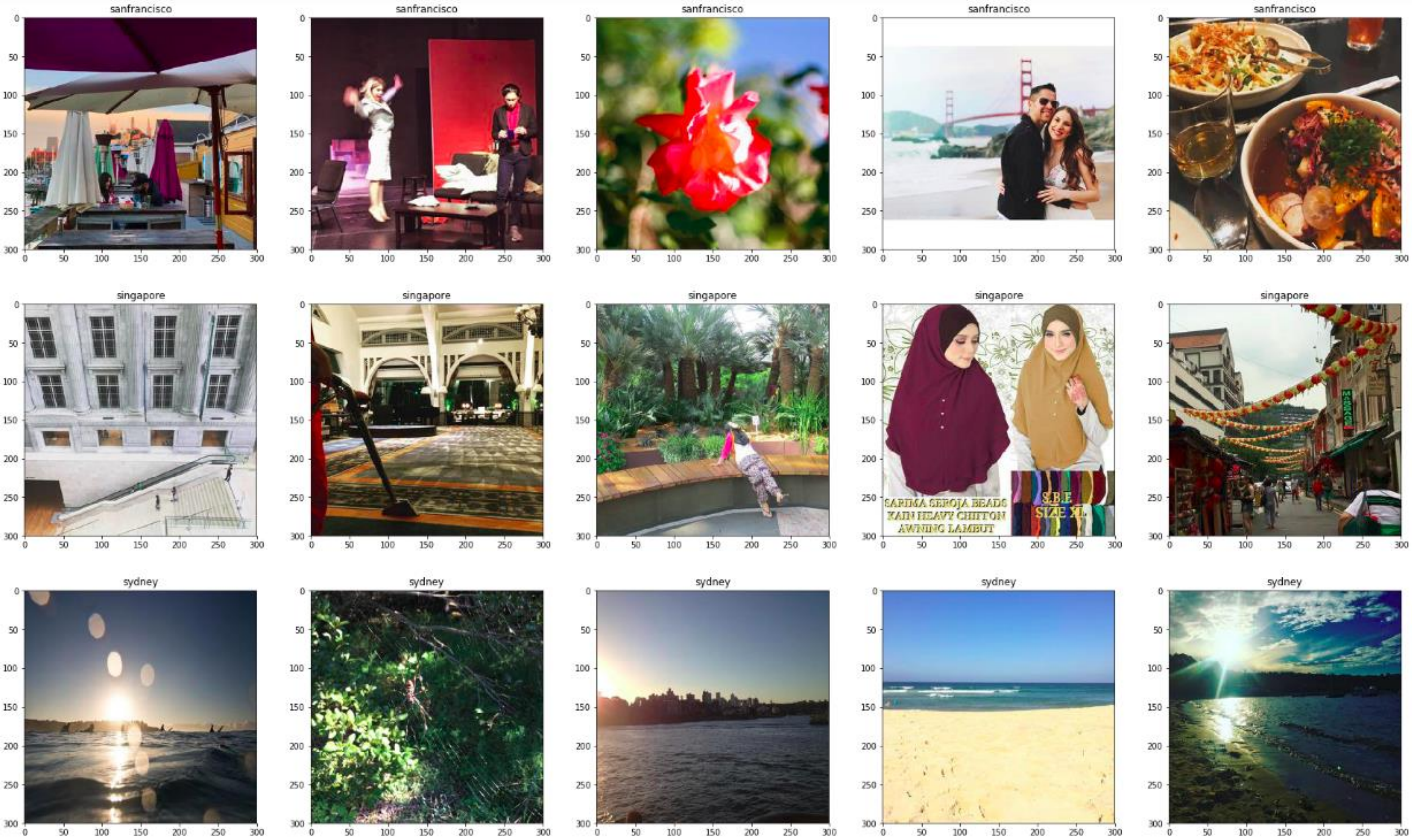
Experiment 4 – VGG16 – training on same data minus random 100 – sample of random images



Experiment 4 – VGG16 – training on same data minus random 100 (same set for validation) – sample images



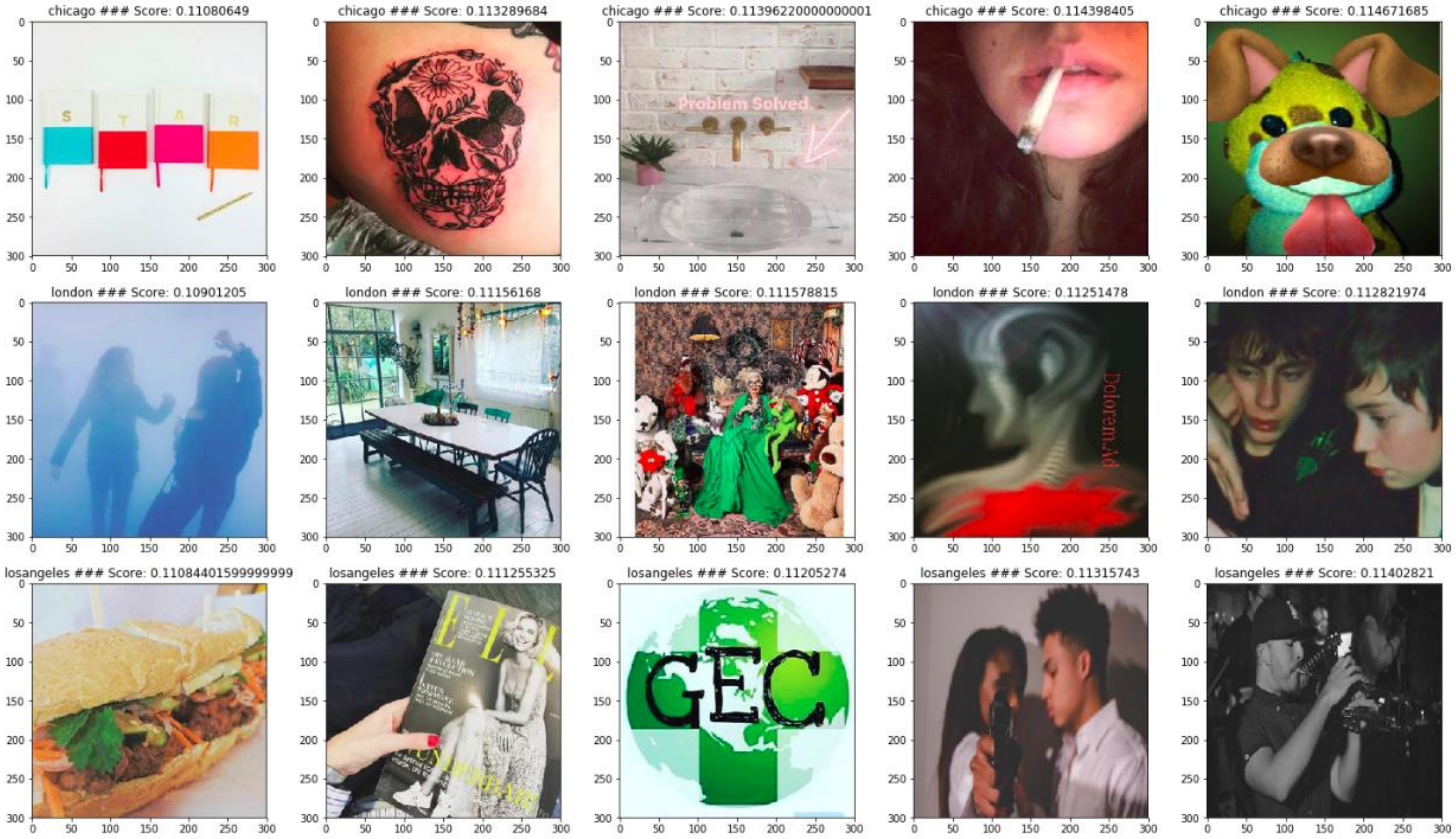
Experiment 4 – VGG16 – training on same data minus random 100 (same set for validation) – sample images



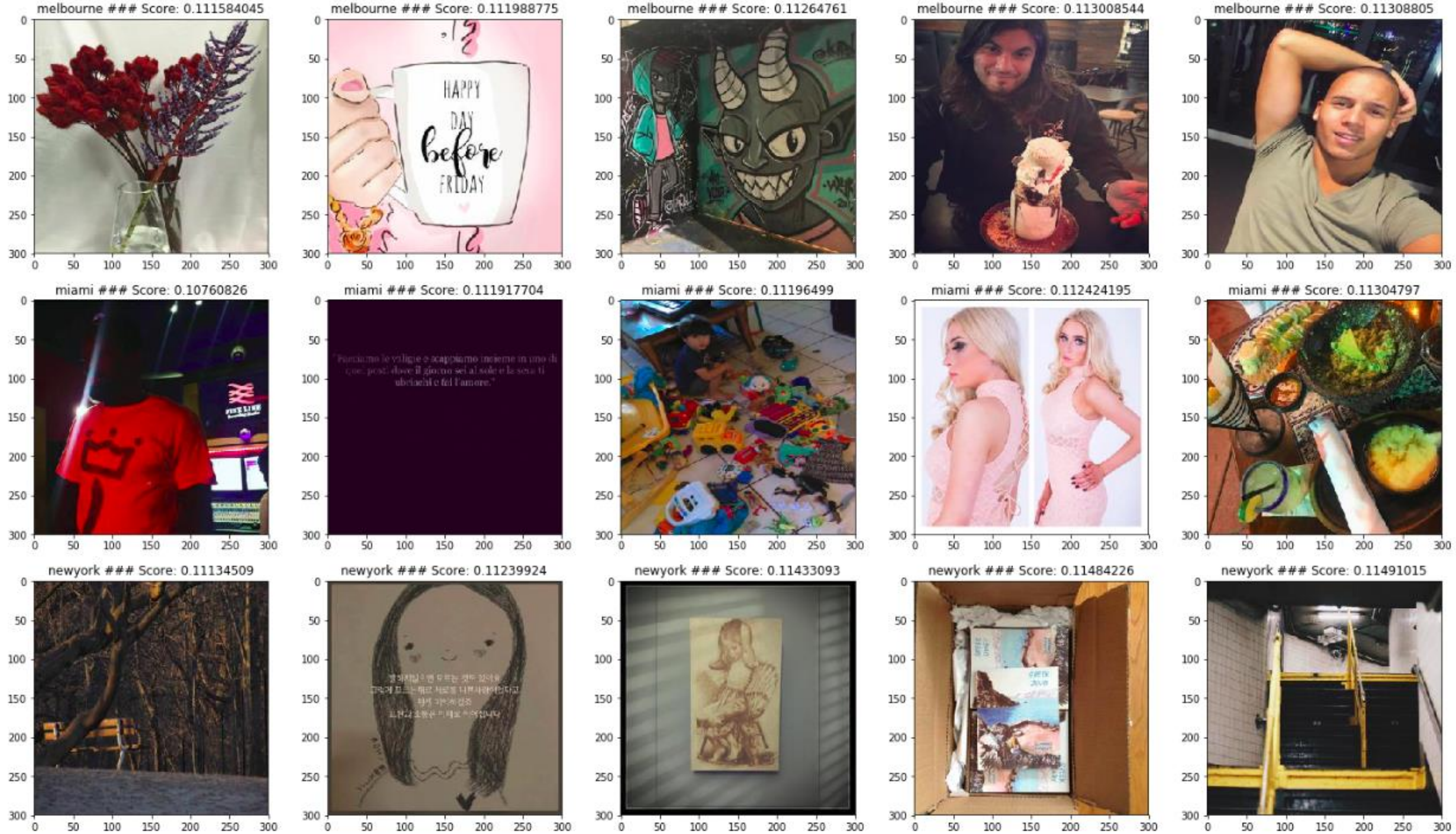
Experiments 2 – deep dive

Lowest probability correctly classified images

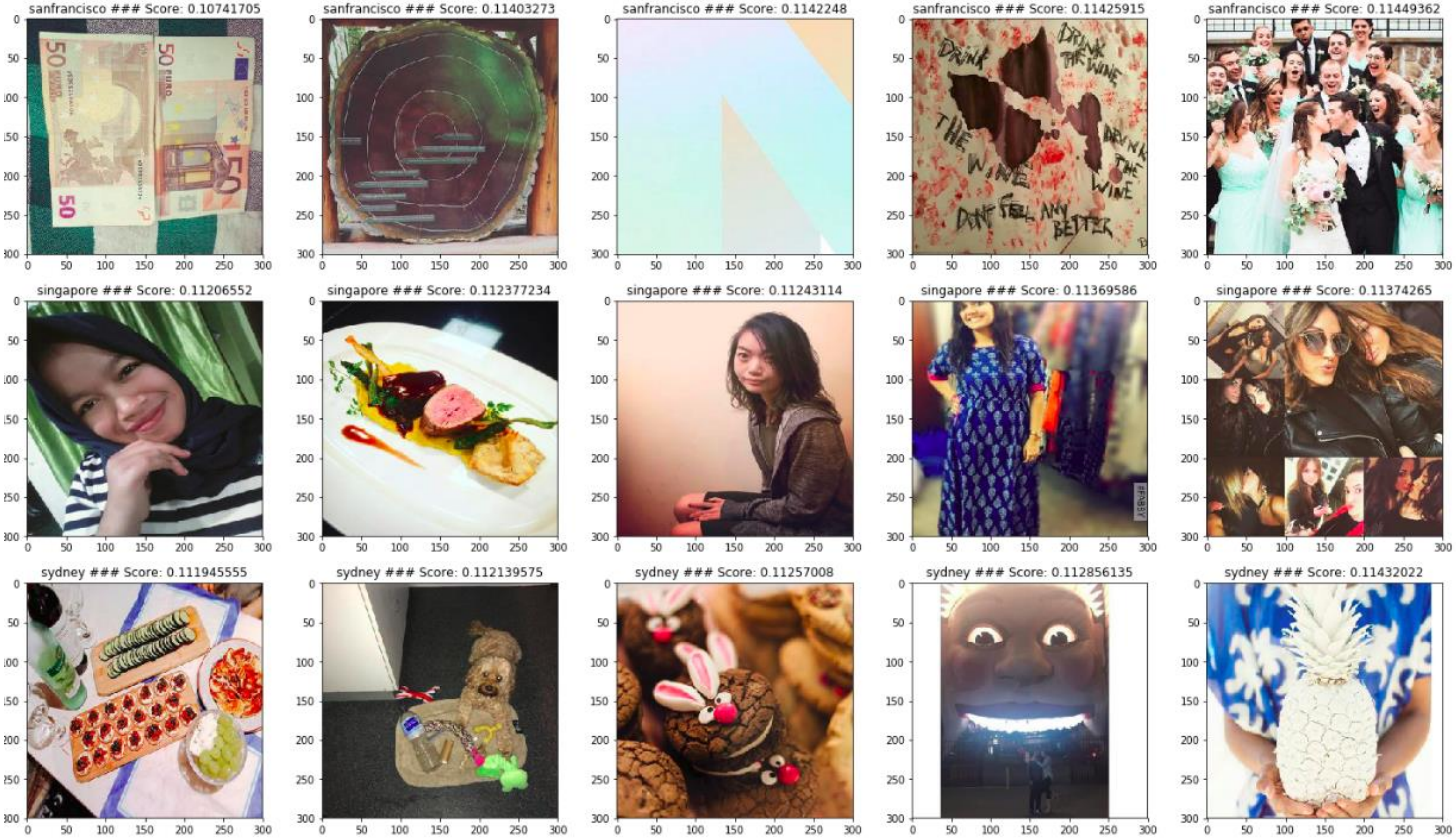
Lowest probability correctly classified images



Lowest probability correctly classified images

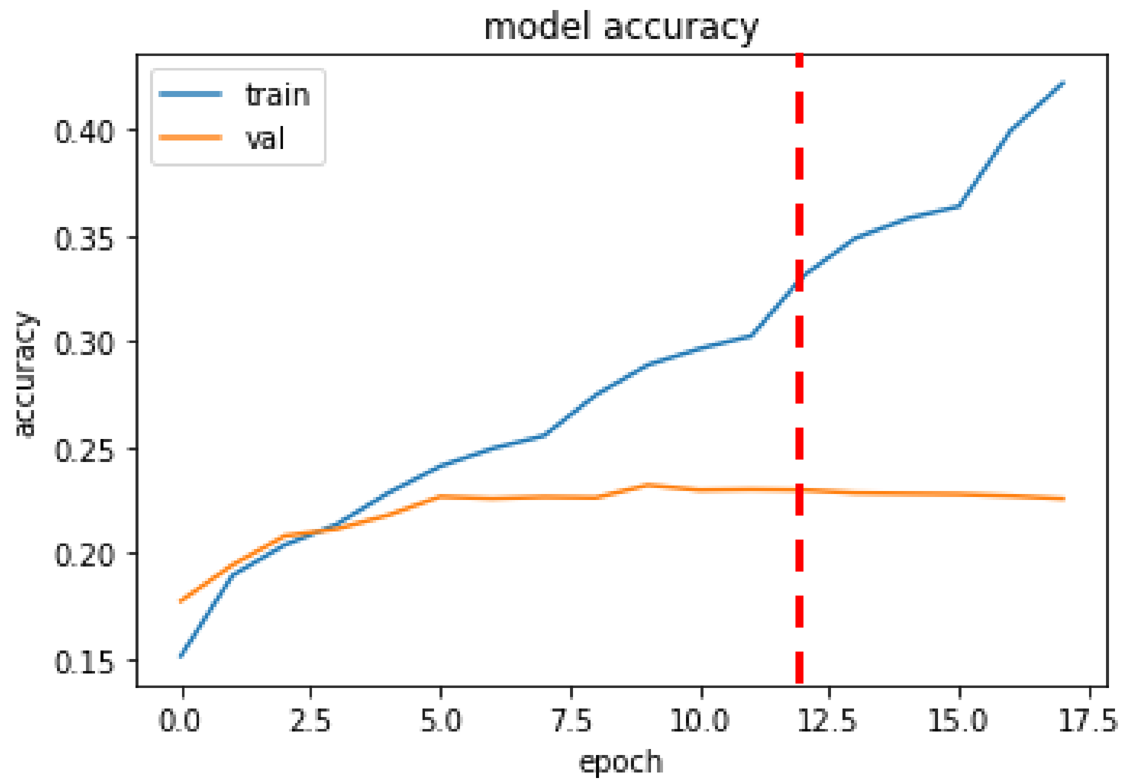


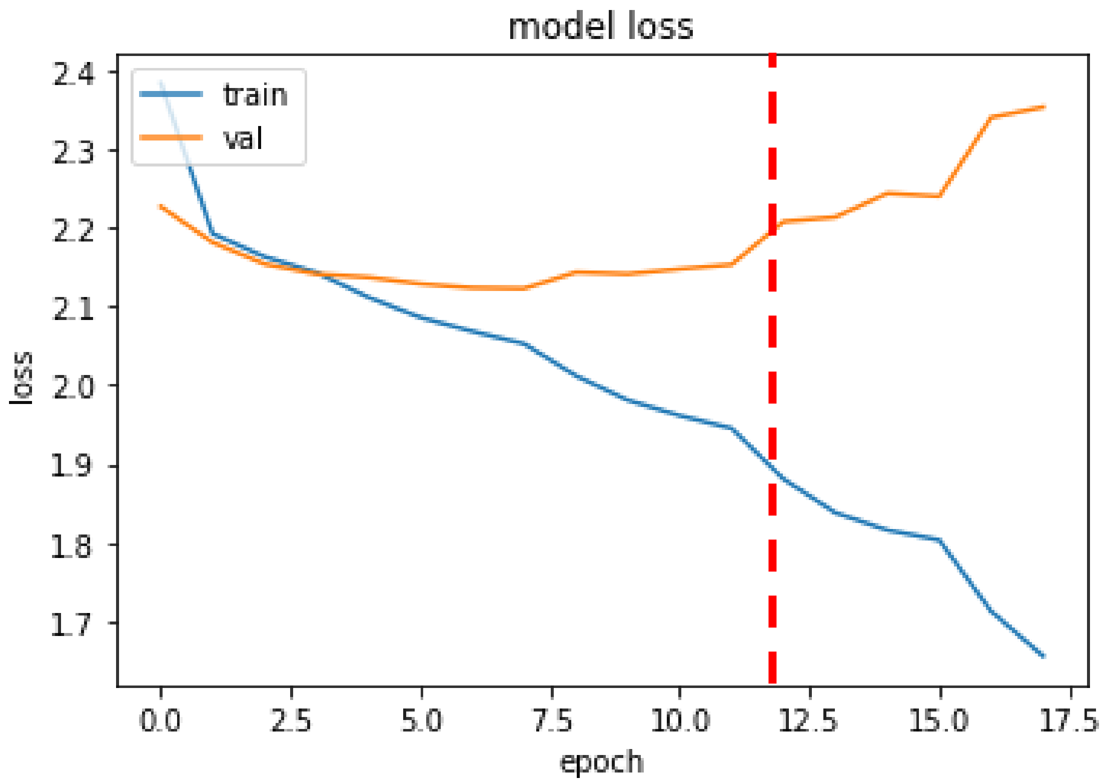
Lowest probability correctly classified images



Error analysis – original experiment

Experiment 2 – VGG16 – training on one set validation and early stopping on other





- Train – full 800k data
- Val – full 50k data
- The split was done according to author proposition

- The training was conducted for 17 epochs (200k images each) so the net saw the training data 4 times
- For inference net after 12 epochs was used as after that the validation accuracy and error started decreasing

Experiment 2 – top 100 predictions for each class

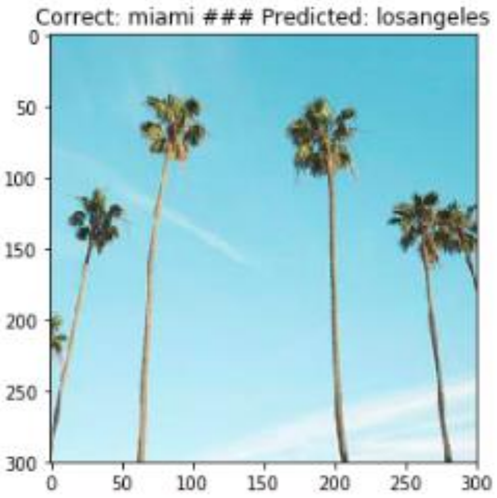
```
## check if all top 100 predictions are correct
print("### CORRECT ###")
for i in range(0,10):
    temp = select_top_n_from_city(data, 100, str(i))
    print(dict_num_cities[i],": ",np.sum(temp["correct"]))
```

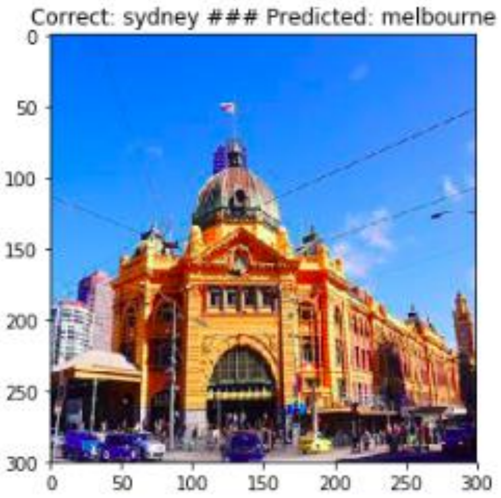
```
### CORRECT ###
chicago :  100
london :  100
losangeles :  99
melbourne :  99
miami :  98
newyork :  100
sanfrancisco :  100
singapore :  100
sydney :  99
toronto :  100
```

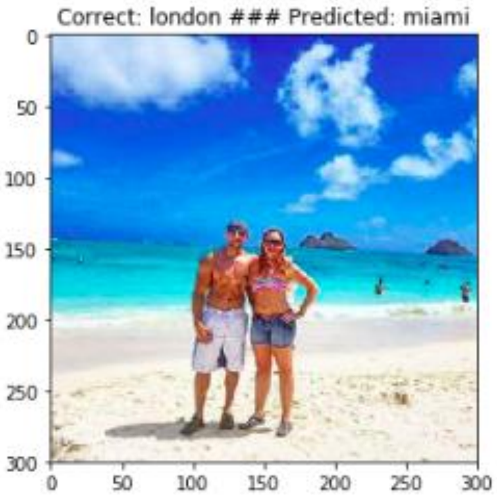
- Vast majority of the top 100 predictions for each class were correct predictions

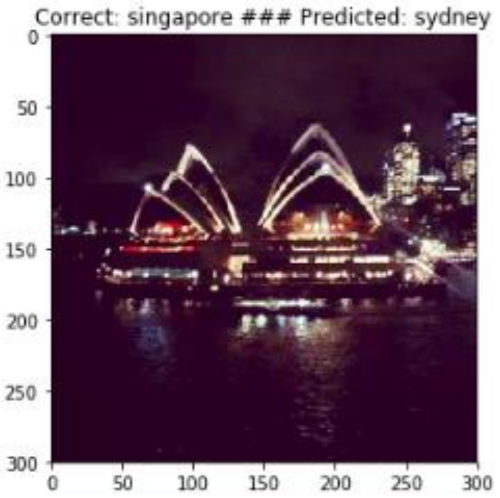
Experiment 2 – top 100 predictions for each class – error analysis

Erroneous images:



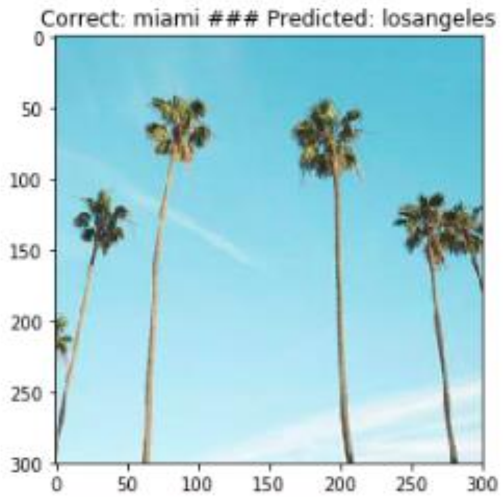






Experiment 2 – top 100 predictions for each class – error analysis

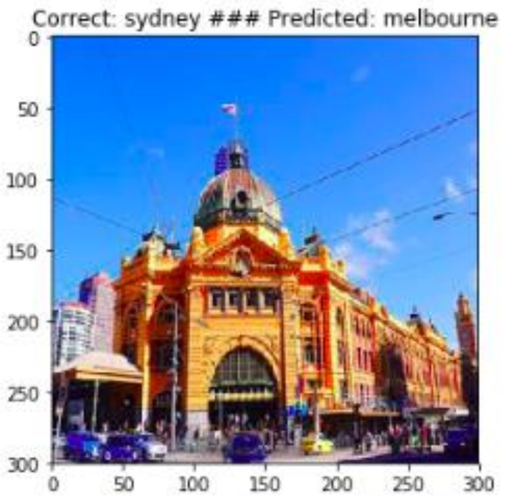
Analyzed image:



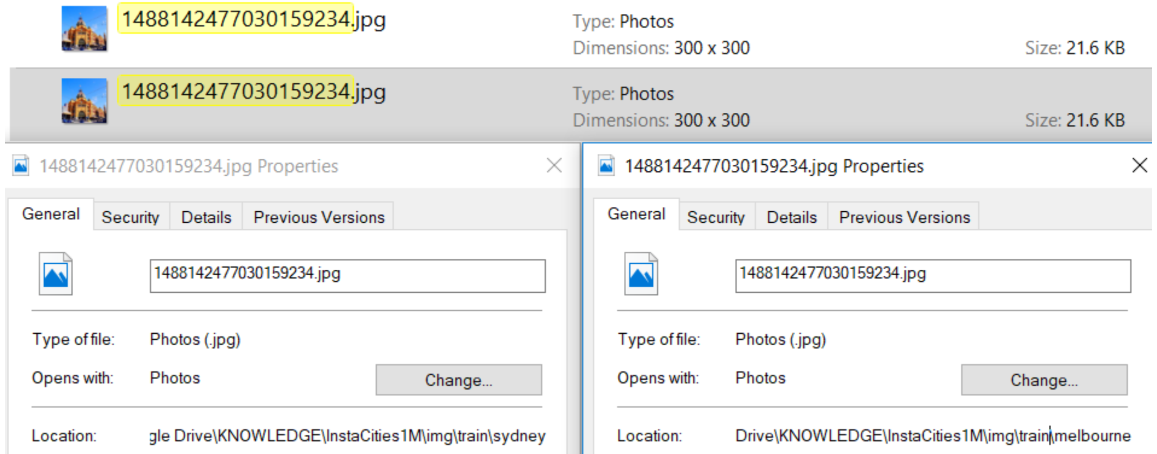
The correct label is Miami but the model predicted Los Angeles. From the analysis of top 15 predictions from LA we can see that the model learned to classify LA based on the appearance of palm trees in the image. There are palm trees in other cities so this is **INCORRECT**.

Experiment 2 – top 100 predictions for each class – error analysis

Analyzed image:



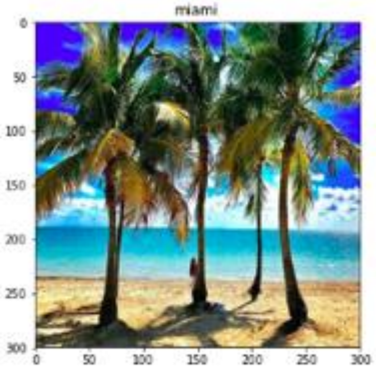
The correct label is Sydney but the model predicted Melbourne. From the analysis of top 15 predictions from Melbourne we can see that the image depicts the Main railway station in Melbourne so it is indeed **CORRECT**. This is actually a dataset error because the same image appears both in Sydney as in Melbourne sub catalog of the training data.



Experiment 2 – top 100 predictions for each class – error analysis

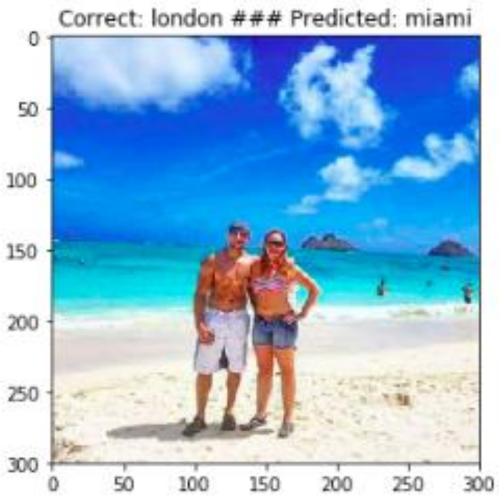
Analyzed image:

The correct label is New York but the model predicted Miami. For sure beach is more characteristic for Miami yet there are some beaches in NYC. Hard to tell whether it is correct or not.



Experiment 2 – top 100 predictions for each class – error analysis

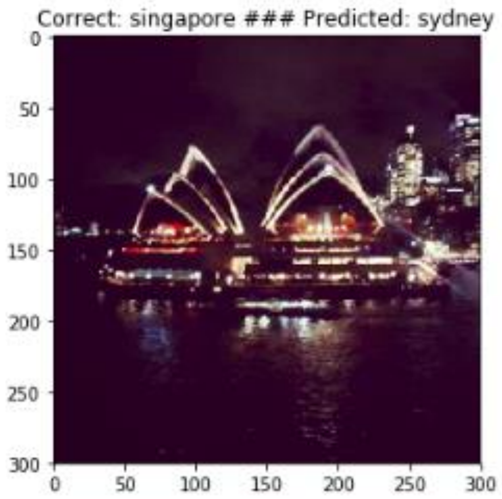
Analyzed image:



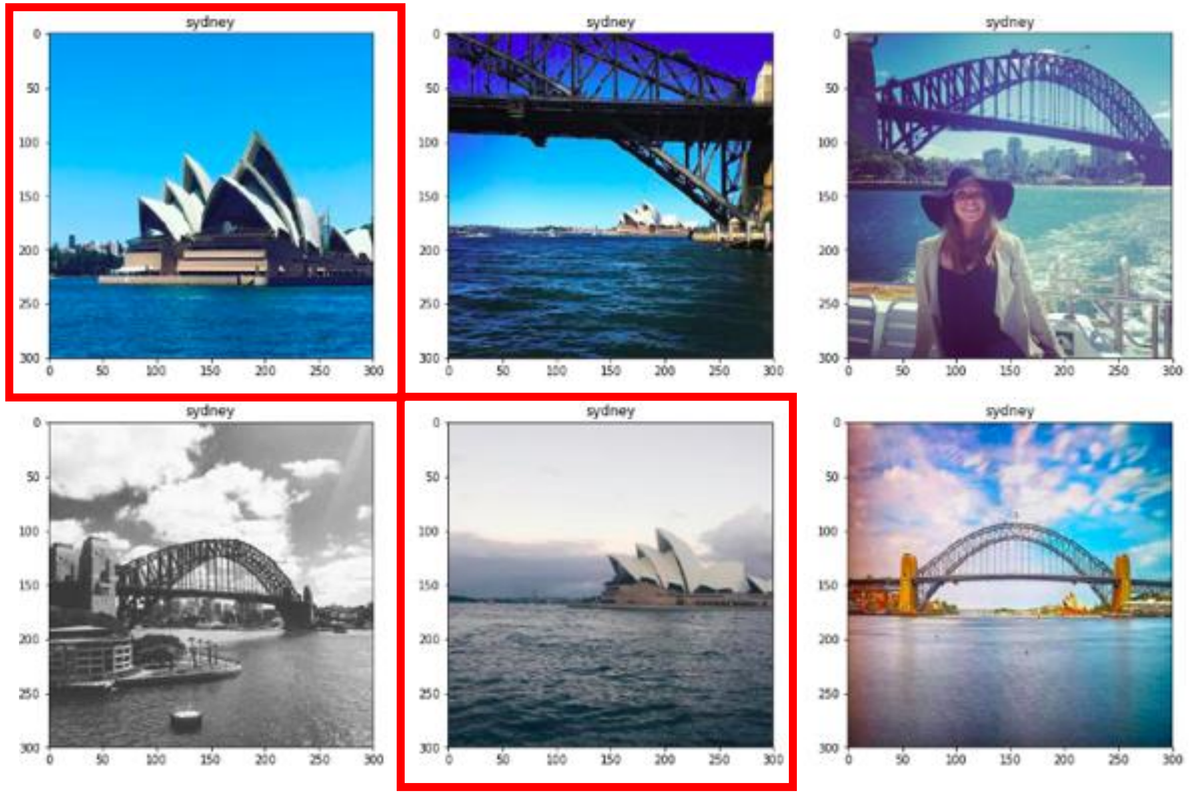
The correct label is London but the model predicted Miami. For sure there are no such beaches in London ☺.

Experiment 2 – top 100 predictions for each class – error analysis

Analyzed image:

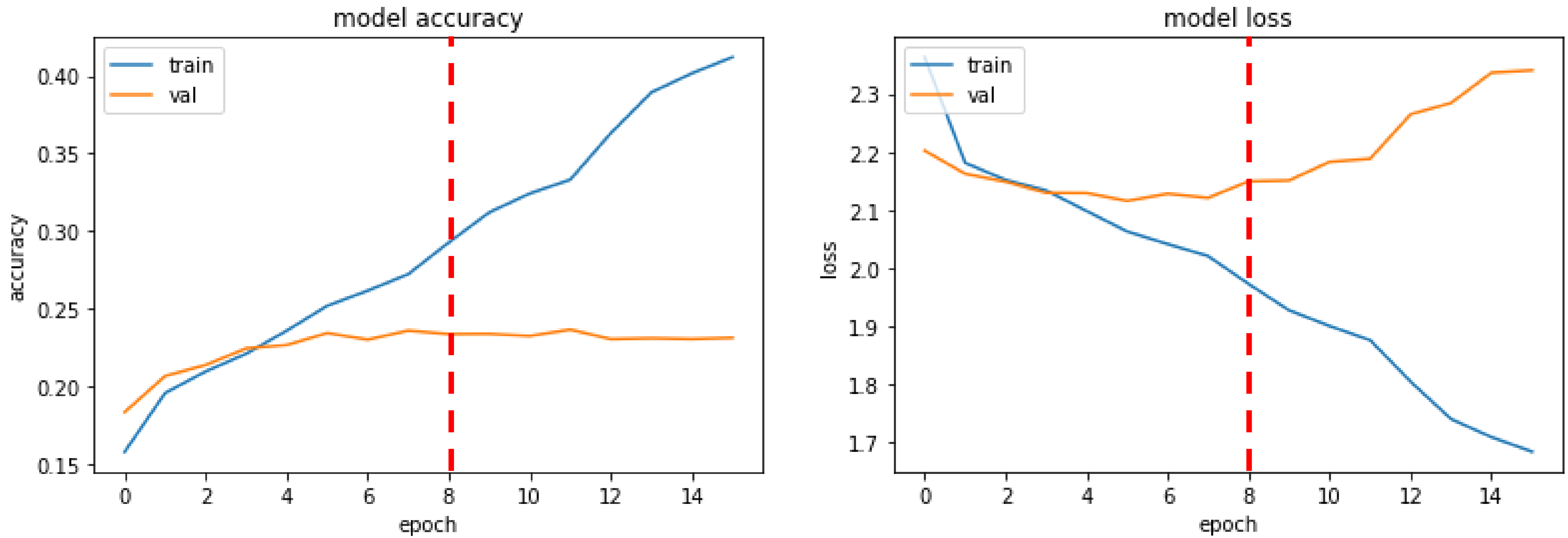


The correct label is Singapore but the model predicted Sydney. From the analysis of top 15 predictions from Sydney we can see that the image depicts the Sydney Opera so it is indeed **CORRECT**. This is actually either a dataset error or a very similar building in Singapore.



Error analysis – top 100 images from each category

Experiment 3 – VGG16 – training on same data minus top 100 (same set for validation)



- Train – full 799 data
- Val – full 50k data
- Test – top 100 from each class (1k images not used for training)

- For inference net after 8 epochs was used as after that the validation accuracy started decreasing and error increased

Experiment 3 – VGG16 – training on same data minus top 100 (same set for validation)

Results:

```
test_accuracy = np.mean(full_test_top_100.Predictions==full_test_top_100.True_class)

print("\nTest accuracy: {} %".format(test_accuracy*100))
```

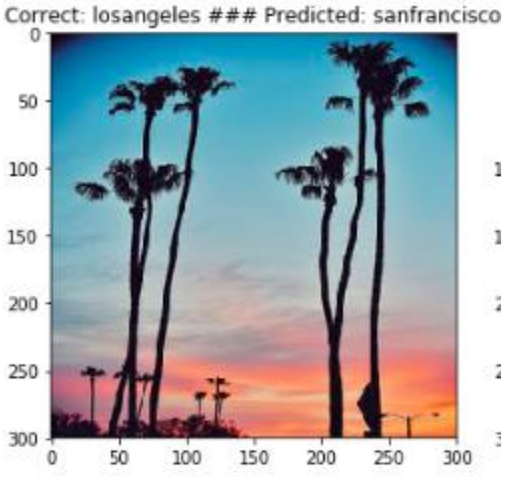
```
Test accuracy: 95.89999999999999 %
```

This means that out of the 1000 images (100 top correctly classified images from each category) there are 41 images that are incorrectly classified.

- 1 in LA
- 9 in Melbourne
- And the rest in Miami

Experiment 3 – top 100 predictions for each class – error analysis – Los Angeles

Analyzed image:



The correct label is LA but the model predicted SF. From the analysis of top 15 predictions from LA we can see that the images mostly depict palm trees so this is wired that the model made mistake.

Experiment 3 – top 100 predictions for each class – error analysis – Melbourne

Analyzed images:

Experiment 3 – top 100 predictions for each class – error analysis – Melbourne

Analyzed images:

There are two misclassification cases in Melbourne:

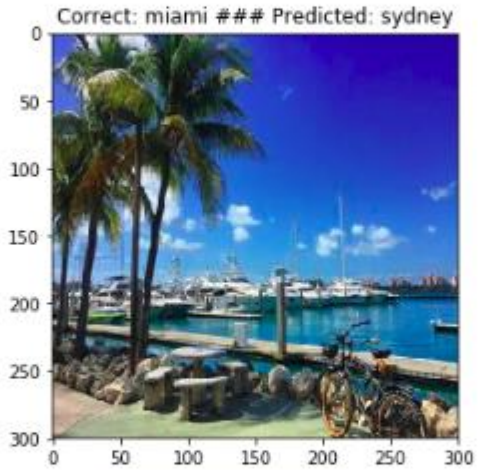
- Flinders railway station being classified as other city (London, SF, Sydney, Singapore)
- Skyline classified as Singapore (this is constant, no randomness as in previous error)

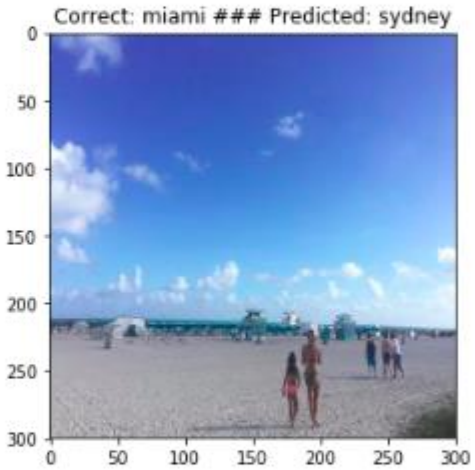
Experiment 3 – top 100 predictions for each class – error analysis – Miami

Analyzed images:

Out of the 31 misclassifications:

- 28 look like the images on the right
- The only 3 different are presented below





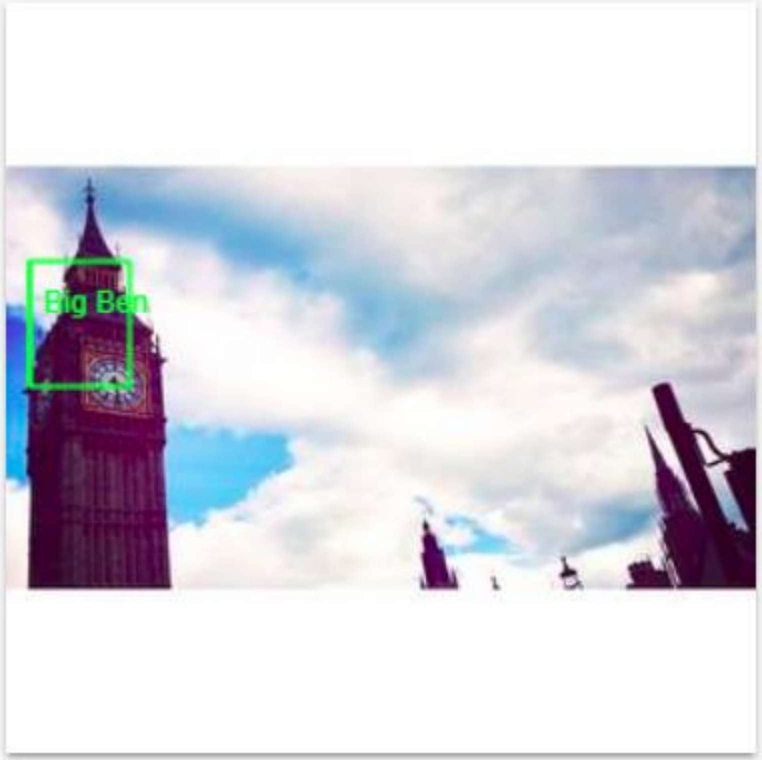
Comments from left:

- Harbor classified as Sydney
- Skyline classified as Singapore
- And the last one is a mystery (beach classified as Sydney)

Landmark analysis

Landmark analysis – Method/process of getting data

Exemplary detection:



1481039023504972877.jpg

Pricing:

		PRICE PER 1,000 UNITS, BY MONTHLY U
FEATURE	1–1,000 UNITS/MONTH	1001–5,000,000 UNITS/MONTH
Label Detection	Free	$1.50
Text Detection	Free	$1.50
Safe Search (explicit content) Detection	Free	Free with Label Detection, or $1.50
Facial Detection	Free	$1.50
Landmark Detection	Free	$1.50
Logo Detection	Free	$1.50

The automated categorization was done using the Google Vision API.
Process:

- An image is send to the API
- Landmark detection are returned

The speed: 6 images per second
Total cost if we were to run detection on all images: $1500
For now I have run the detection on top 50k of images.

Landmark analysis – Limitations of Automated approach

As this is an automated approach we do not really know what is the quality of the landmark detection. Some example of incorrect landmark detection or lack of detection.

Incorrect landmark detection:

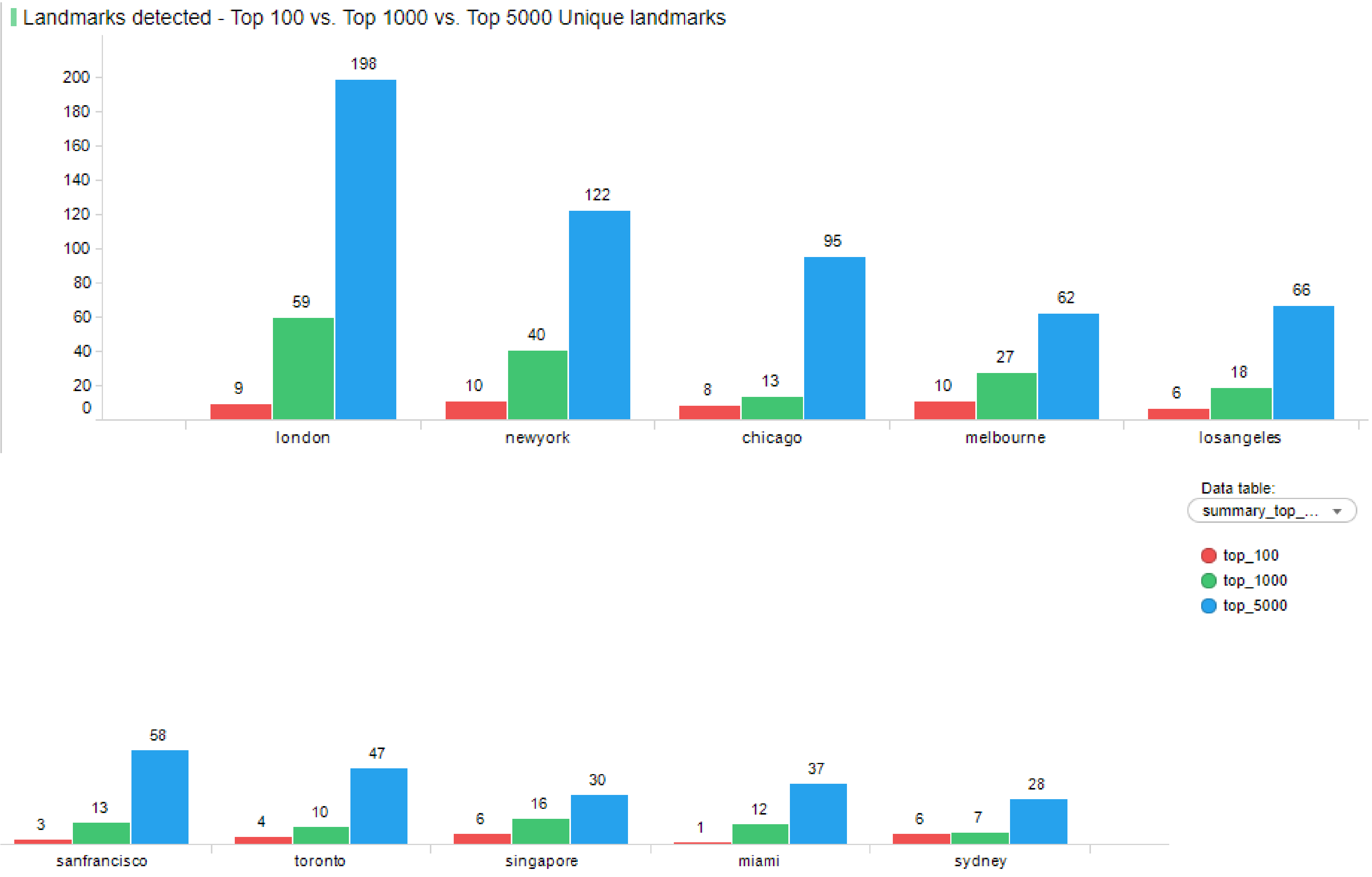
1481039473386755277.jpg

No landmark detection for Big Ben:

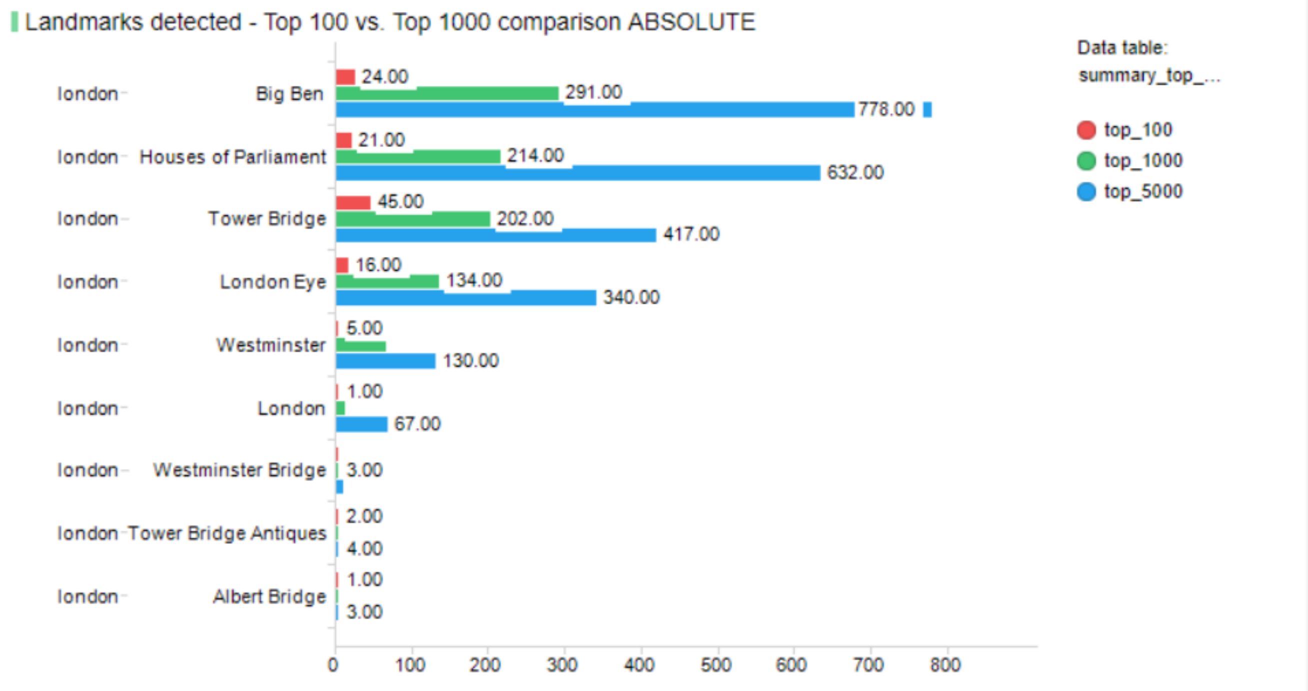
1481039131692802060.jpg

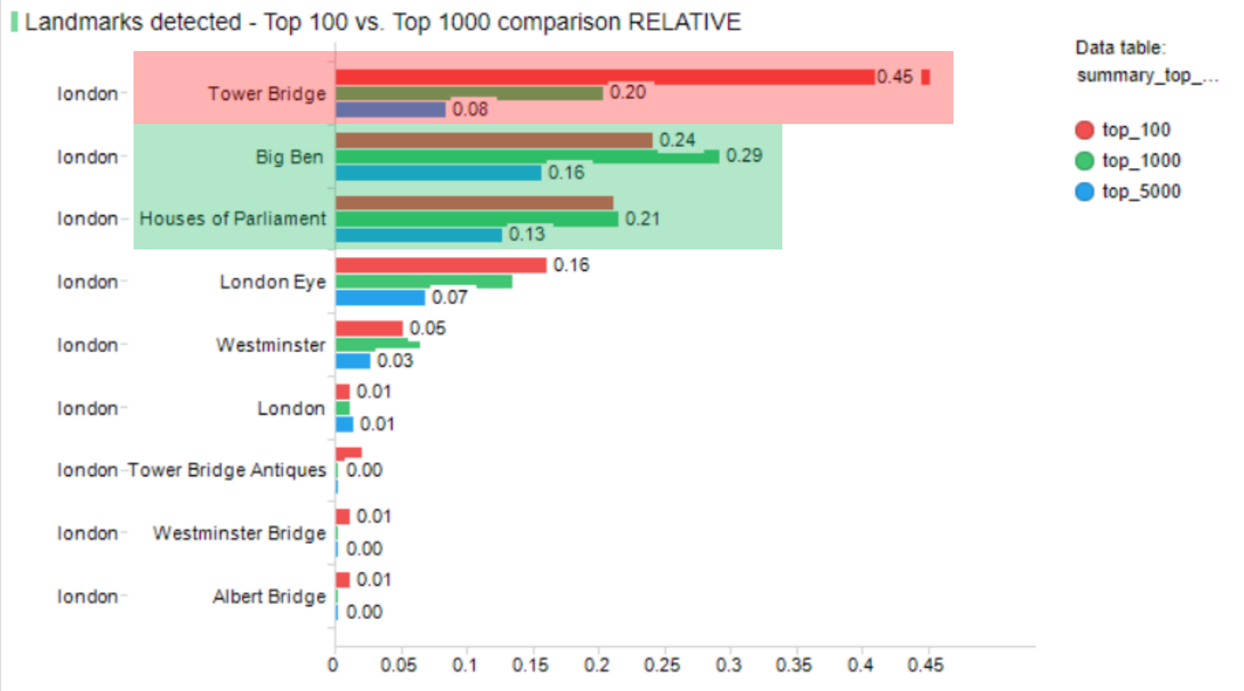
For sure, with increase of dataset size the problem grows.

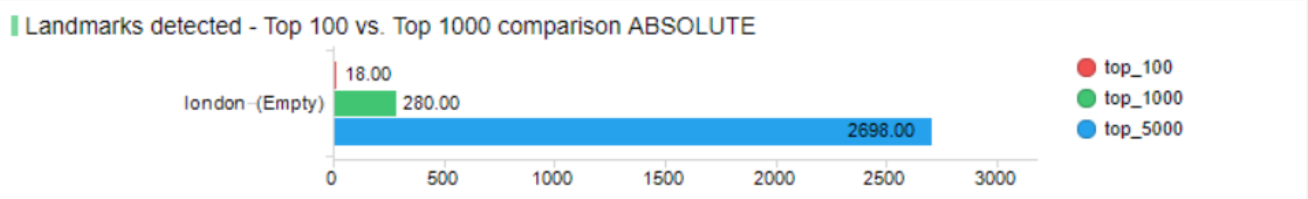
Landmark analysis – Diversity of landmarks per city

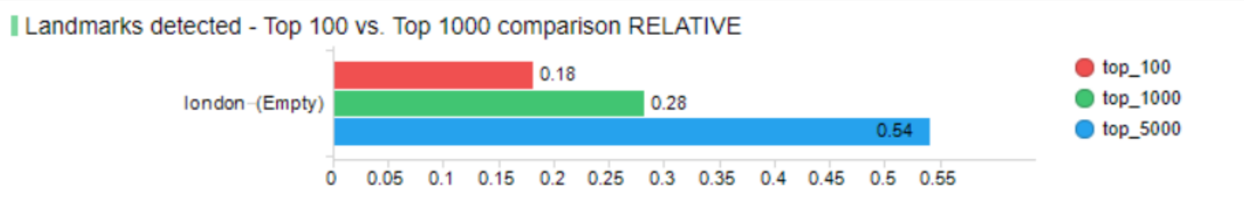


Landmark analysis – In depth analysis – London (from top 100 perspective)









- Landmarks: The absolute # of most frequently appearing landmarks grows but the relative number of those falls
- Lack of landmark: grows with the size of the dataset

Landmark analysis – Diversity of landmarks per city – In depth analysis – London

Frequent landmarks

top_5000 ▲	City	Landmark
28	london	Westminster Abbey
33	london	The Westminster
41	london	National Gallery
52	london	Trafalgar Square
66	london	"St Pauls Cathedral"
67	london	London
111	london	Buckingham Palace
130	london	Westminster
167	london	"St. Pauls Cathedral"
340	london	London Eye
417	london	Tower Bridge
632	london	Houses of Parliament
778	london	Big Ben

Infrequent landmarks

top_5000 ▲	City	Landmark
1	london	Hungerford Bridge and Golden Jubilee Bridges
1	london	Oxford Street
1	london	Arc de Triomphe
1	london	building london
1	london	The Radcliffe Camera
1	london	University of Glasgow
1	london	"Kings College"
1	london	LSE High Holborn
1	london	HMS Belfast
1	london	New Covent Garden Market
1	london	Craven Cottage
1	london	Cathedral of Christ the Saviour
1	london	Primrose Hill
1	london	Brussels-Central Railway Station
1	london	Luxembourg
1	london	St Mary-le-Bow
1	london	Brandenburg Gate
1	london	Hammersmith
1	london	Strand
1	london	Holborn Union Building
1	london	Musical Instrument Museum
1	london	"St Jamess Park Lake"

Landmark analysis – In depth analysis – Summary (from top 100 perspective)

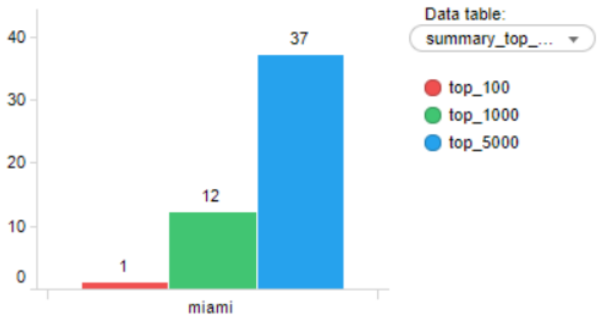
Category	City
• abs # of landmarks grows but the relative % falls • lack of landmarks increases with the dataset size	Melbourne, NYC, London, Chicago, LA, Singapore, Sydney, Toronto, SF, Miami, Sydney, Toronto, SF

Differentiators:

- Melbourne – has just one frequent landmark (Flinders Street station), which is split into 2 labels
- NYC – 7 frequent landmarks, high diversity
- Chicago – impact of Starry Night (suspicious image) disappears almost completely
- LA, Singapore, Miami – very high % of lack of classification
- Sydney – dominated by Harbour Bridge and Opera
- Toronto, SF – has just one frequent landmark

Landmark analysis – In depth analysis – other possibilities

Landmarks diversity



Tells us how the diversity changes with the increase of test dataset size.

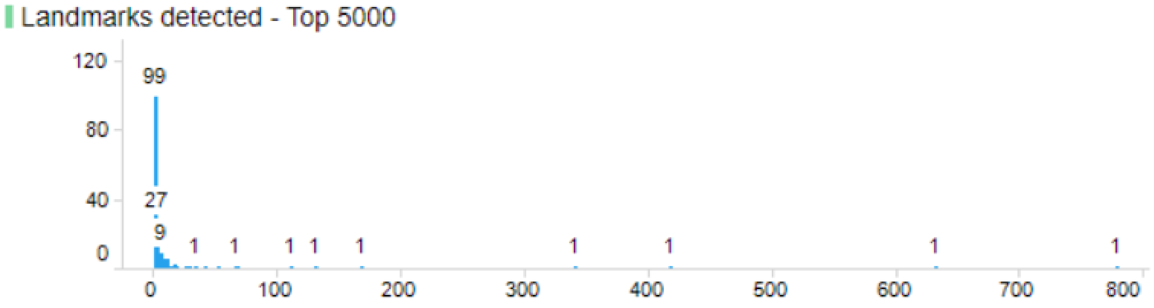
summary_top_5000_landmarks

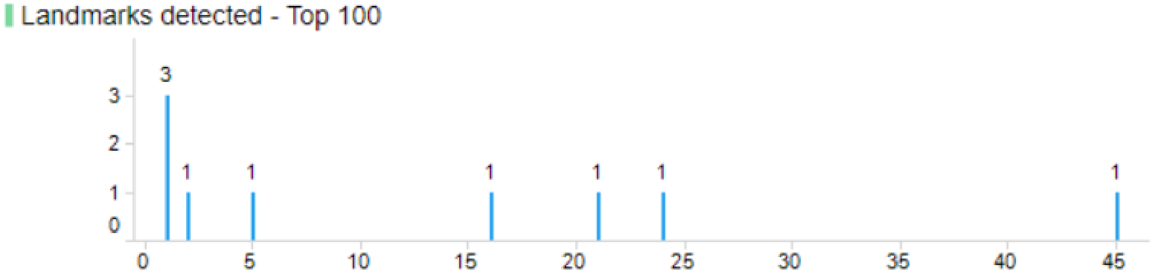
top_5000 ▲	City	Landmark
2	miami	Miami Beach
2	miami	Vizcaya Museum and Gardens
2	miami	"Vlissingens Sea-Side Boulevar
2	miami	South Pointe Park
4	miami	Downtown Miami
5	miami	Miami Seaquarium
9	miami	Miami
11	miami	South Beach

summary_top_100_landmarks

top_100	City	Landmark
1	miami	Expocenter of ...

Landmarks distribution

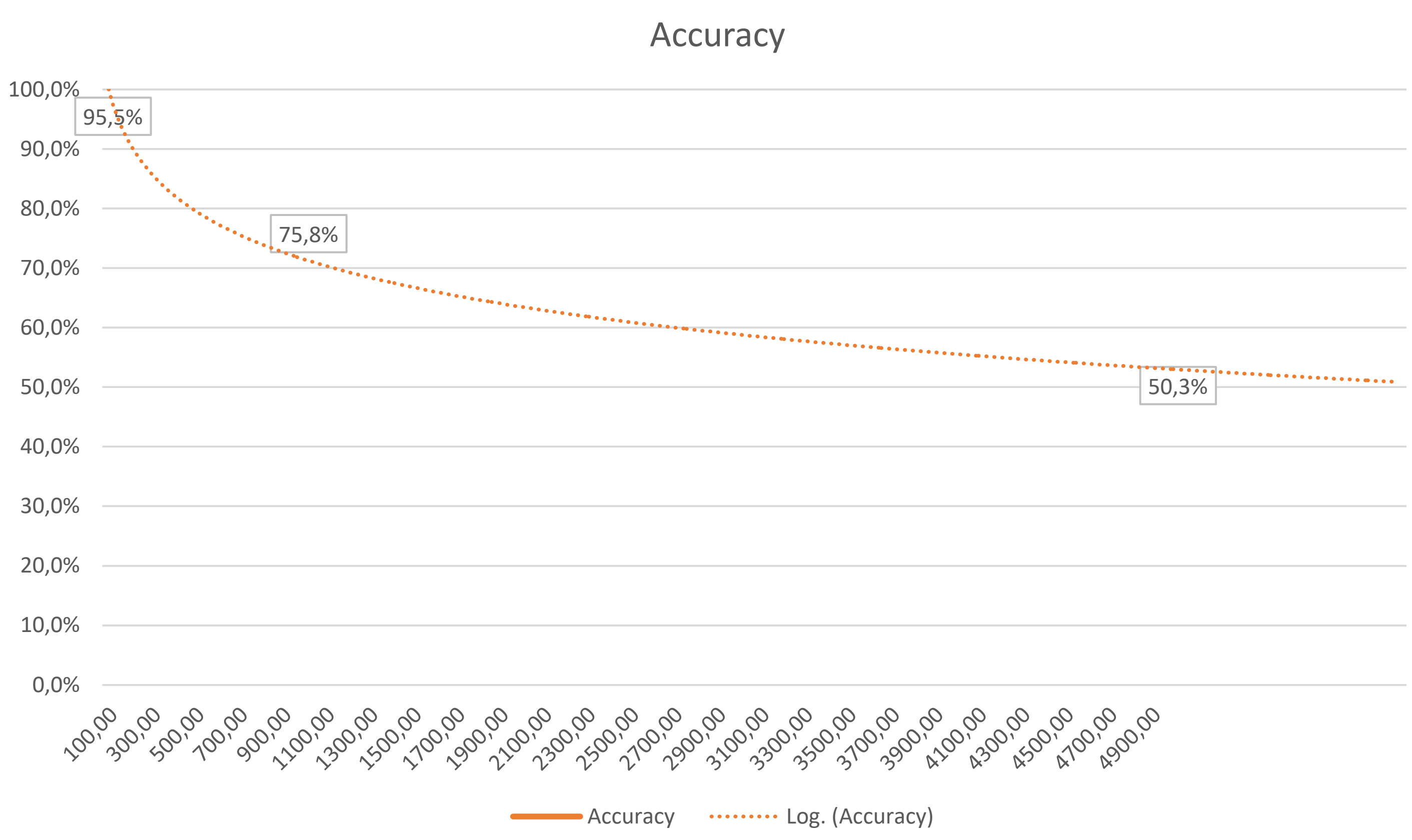




Tells what is the frequency of appearance of each landmark.

Accuracy Curve with decreasing number of training samples

Accuracy Curve for increasing size of dataset



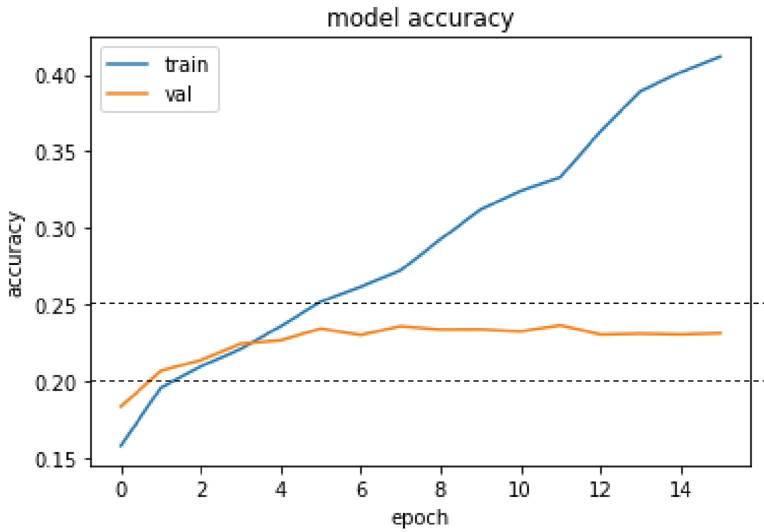
- The dotted curve is a logarithm approximation of the data points
- I have selected the logarithm curve because I expect such a relation but this is a BIASED decision

Top 100 – repeated experiment

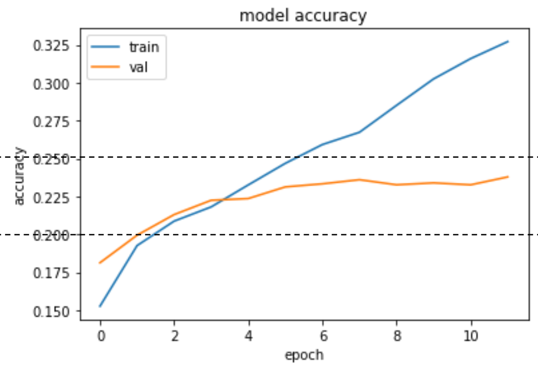
Top 100 – repeated experiment

Introduction: Each experiment was repeated 3 times. For the top 100 experiment the images where the same (100 images from each category that were predicted by the original model with highest probability for this class) but the order of the observations given while training the model was different.

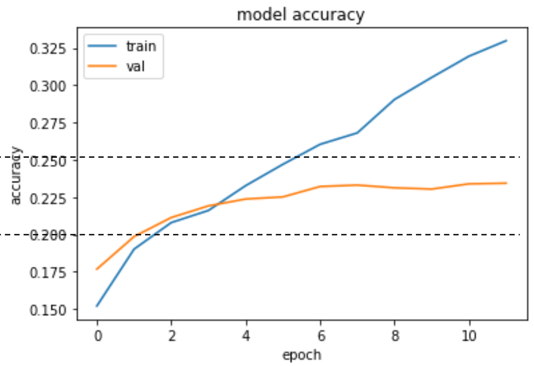
Experiment 1:



Experiment 2:



Experiment 3:



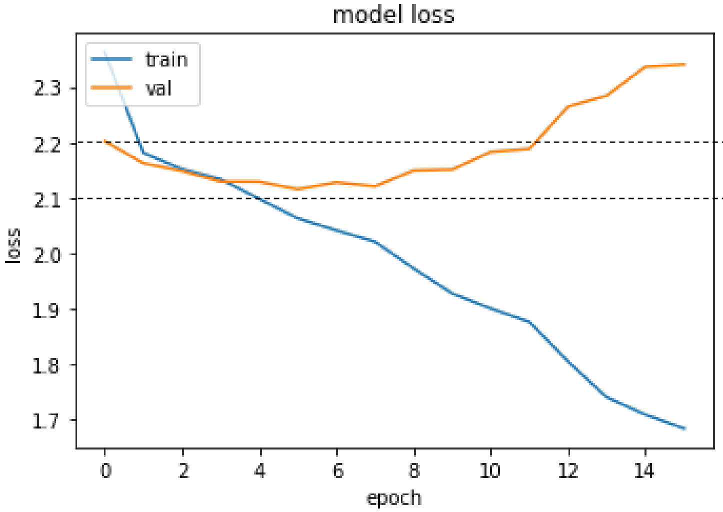
Accuracy:	95.5%	99.8%	94.6%

Which gives an average accuracy of 96.6%.

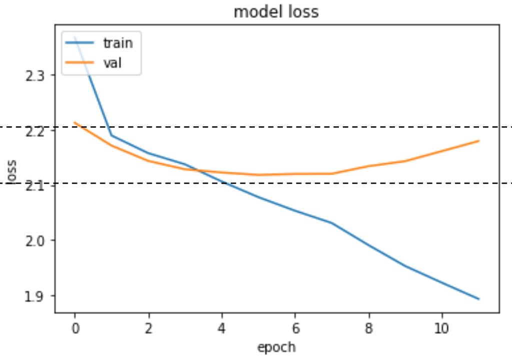
Top 100 – repeated experiment

Result: We can see from the attaches accuracy and loss curves that the training of the network behaves similarly. The accuracy curve for validation set reaches a good performance and plateaus while the training one continues to rise. For model loss we can see that for validation curve it reaches an optimum around 6-8 epoch and rises afterwards while the training one continues to decrease. As for the accuracy on the top 100 images from original net from each category the mean average precision is 96.6%.

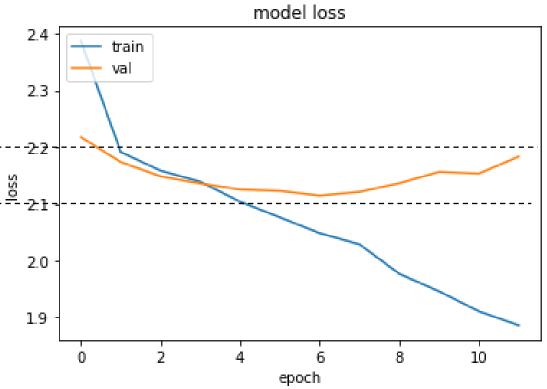
Experiment 1:



Experiment 2:



Experiment 3:

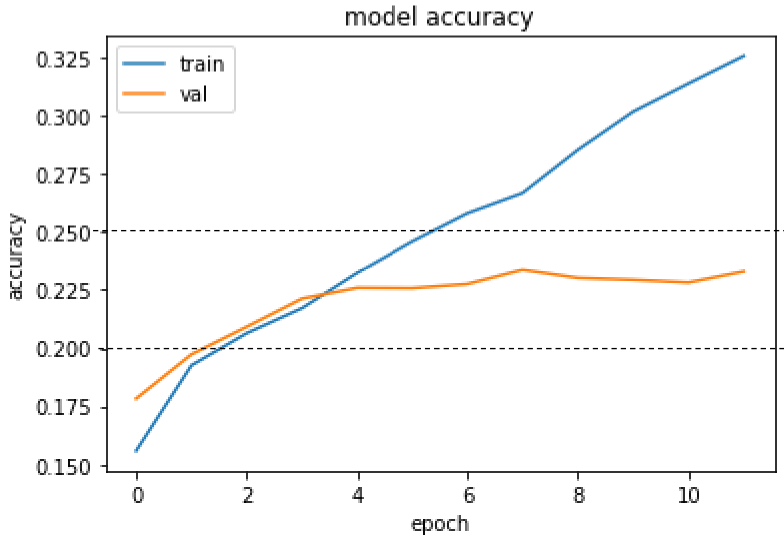


Random 100 – repeated experiment

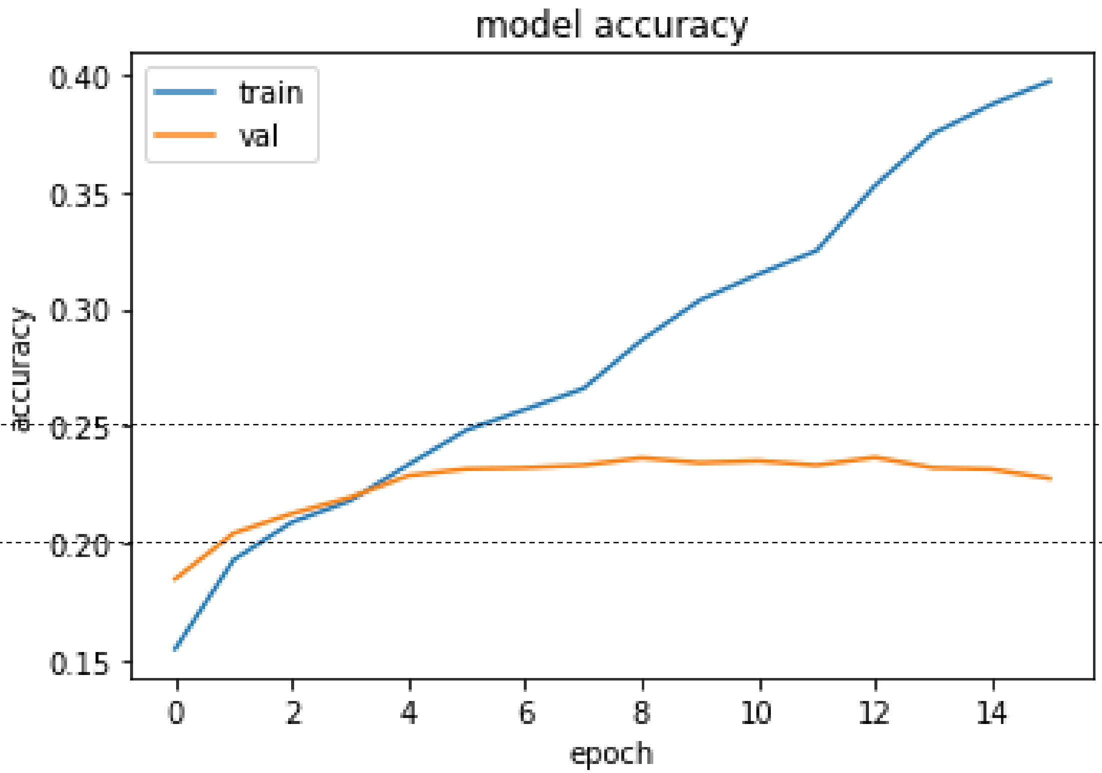
Random 100 – repeated experiment

Introduction: Each experiment was repeated 3 times. For the random 100 experiment the images were random as well as the order of the observations given while training the model.

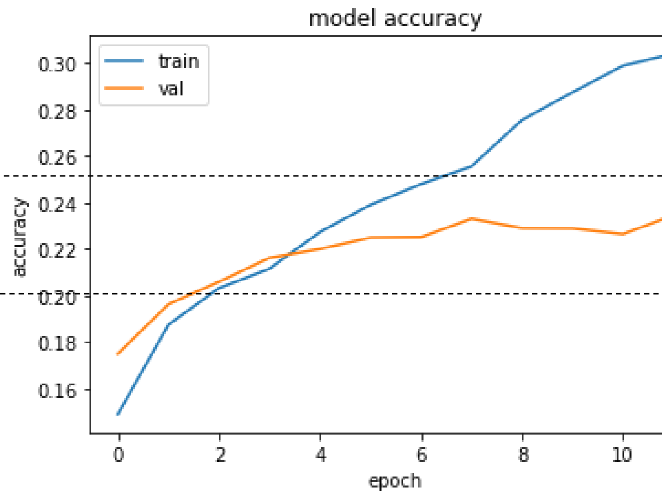
Experiment 1:



Experiment 2:



Experiment 3:



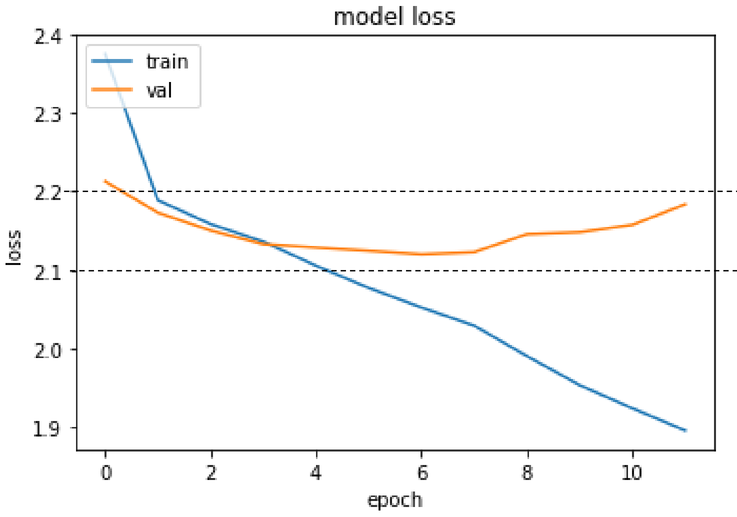
Accuracy:	44.4%	46.5%	45.3 %

Which gives an average accuracy of 45.4 %.

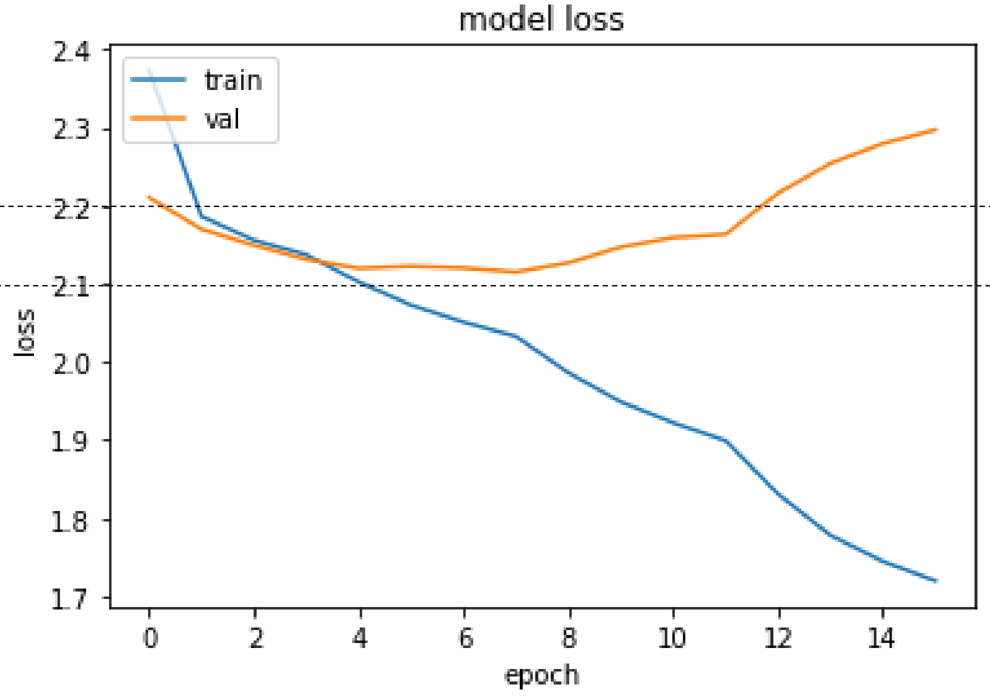
Random 100 – repeated experiment

Result: We can see from the attaches accuracy and loss curves that the training of the network behaves similarly. The accuracy curve for validation set reaches a good performance and plateaus while the training one continues to rise. For model loss we can see that for validation curve it reaches an optimum around 6-8 epoch and rises afterwards while the training one continues to decrease. As for the accuracy on the random 100 images from original net from each category the mean average precision is 45.4%.

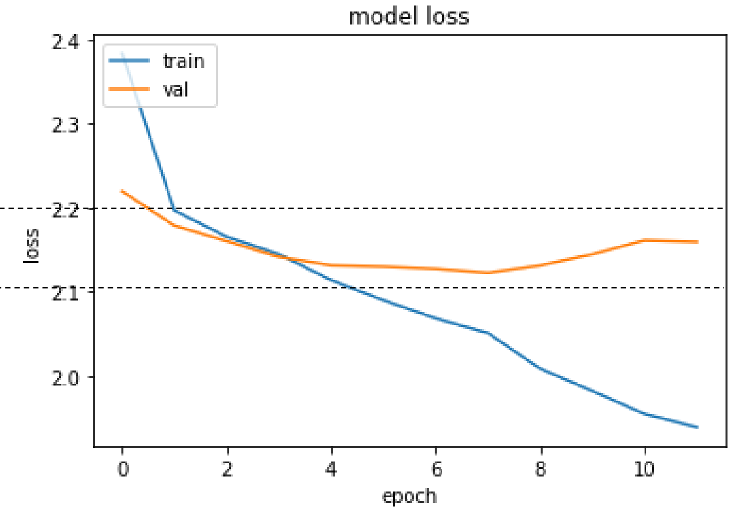
Experiment 1:



Experiment 2:



Experiment 3:

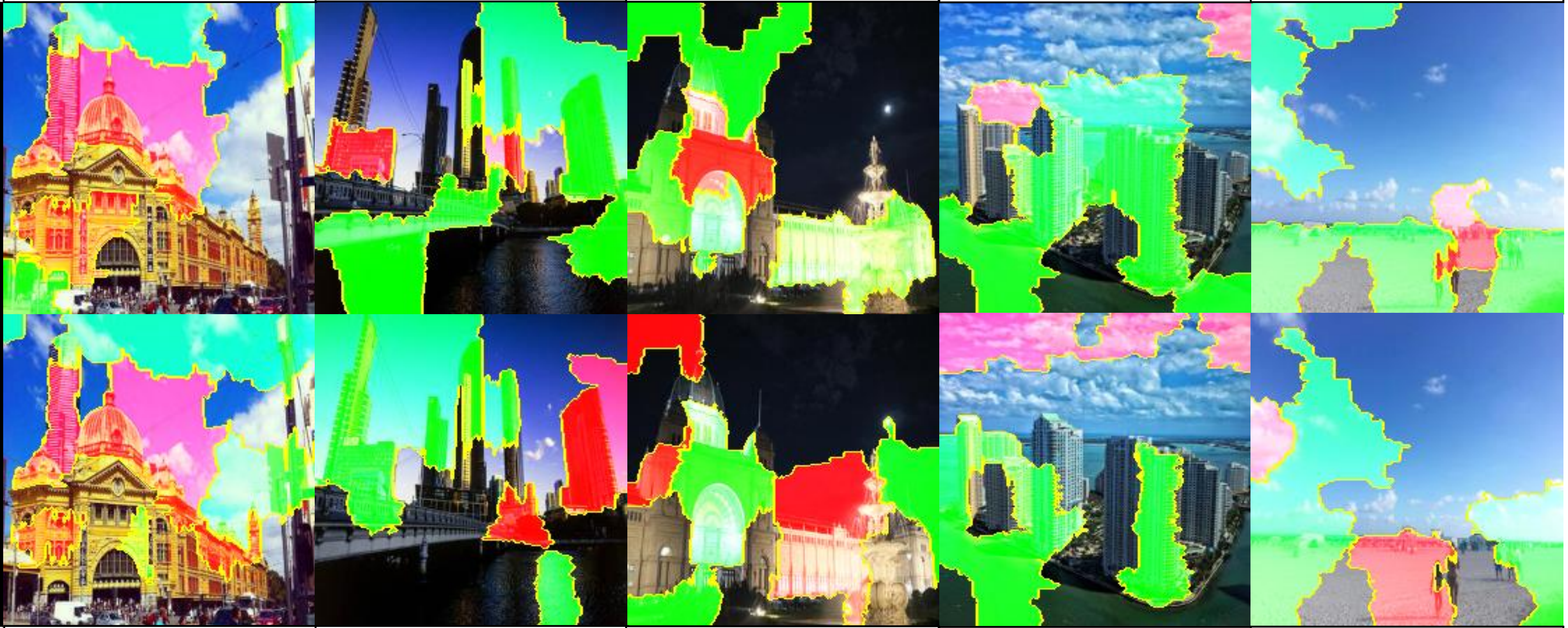


Errors – LIME explanation

Errors – LIME explanation – experiment 1

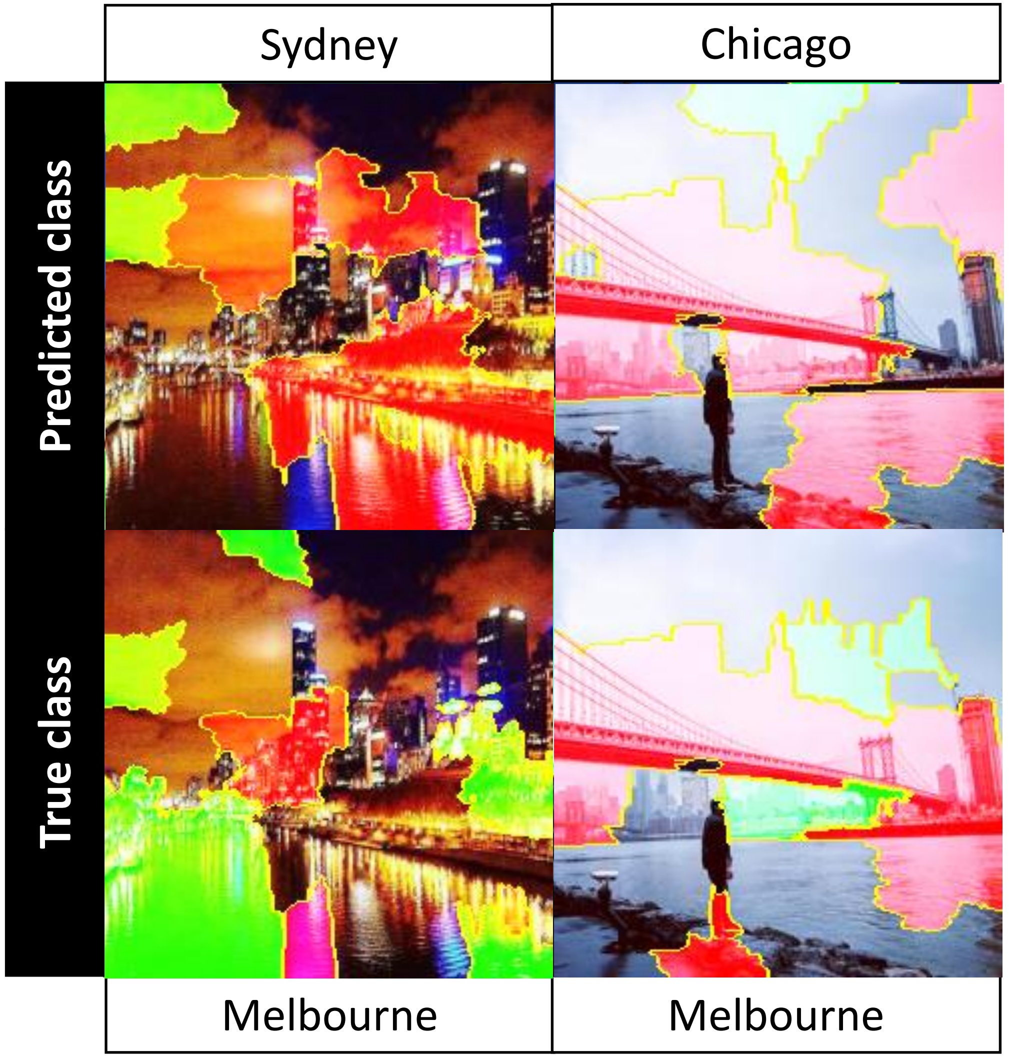
There were 45 errors in total. Only 5 of those were meaningful the other ones depicted Miami festival and dog on pink background.

Predicted class	Sydney	Chicago	London	Singapore	Sydney
True class	Melbourne	Melbourne	Melbourne	Miami	Miami



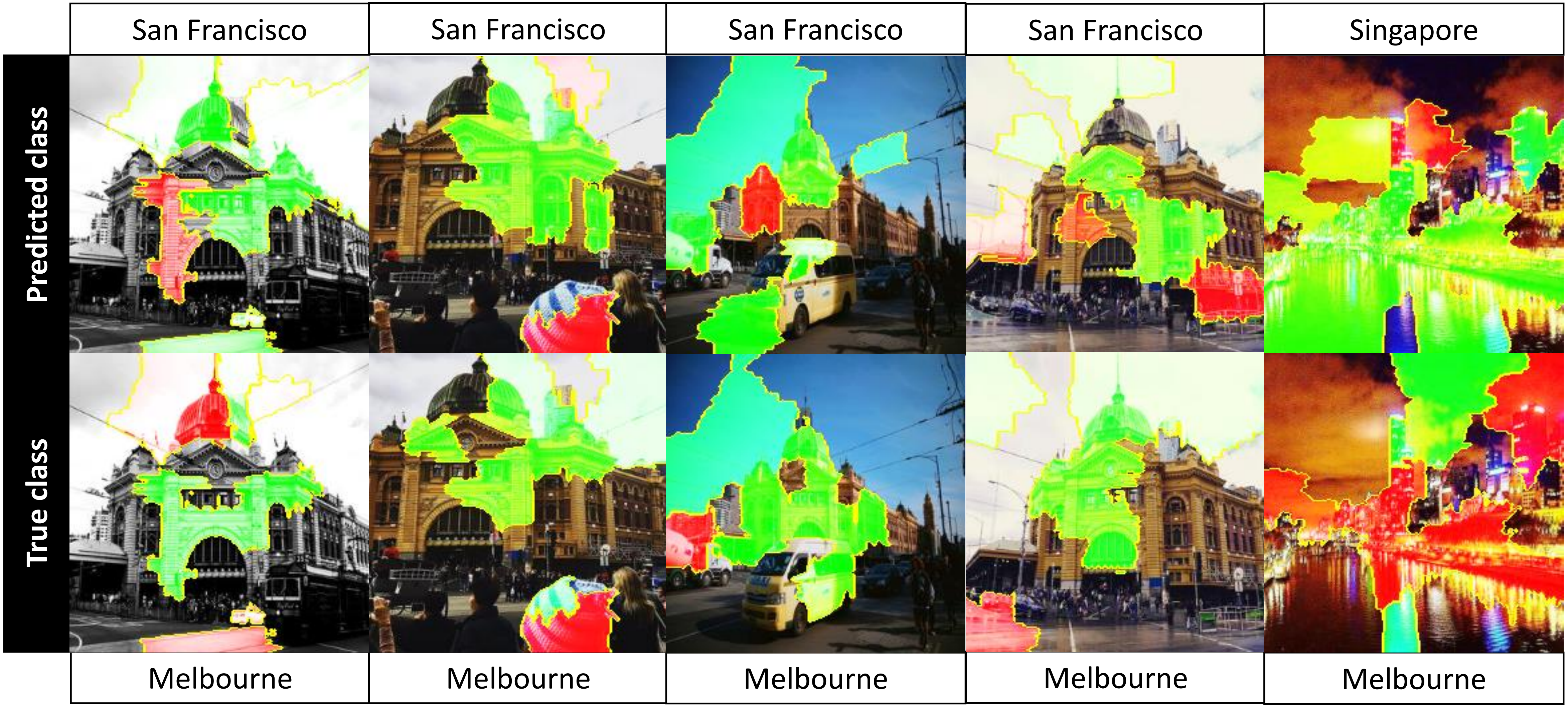
Errors – LIME explanation – experiment 2

There were 2 errors in total and all of those were meaningful.



Errors – LIME explanation – experiment 3

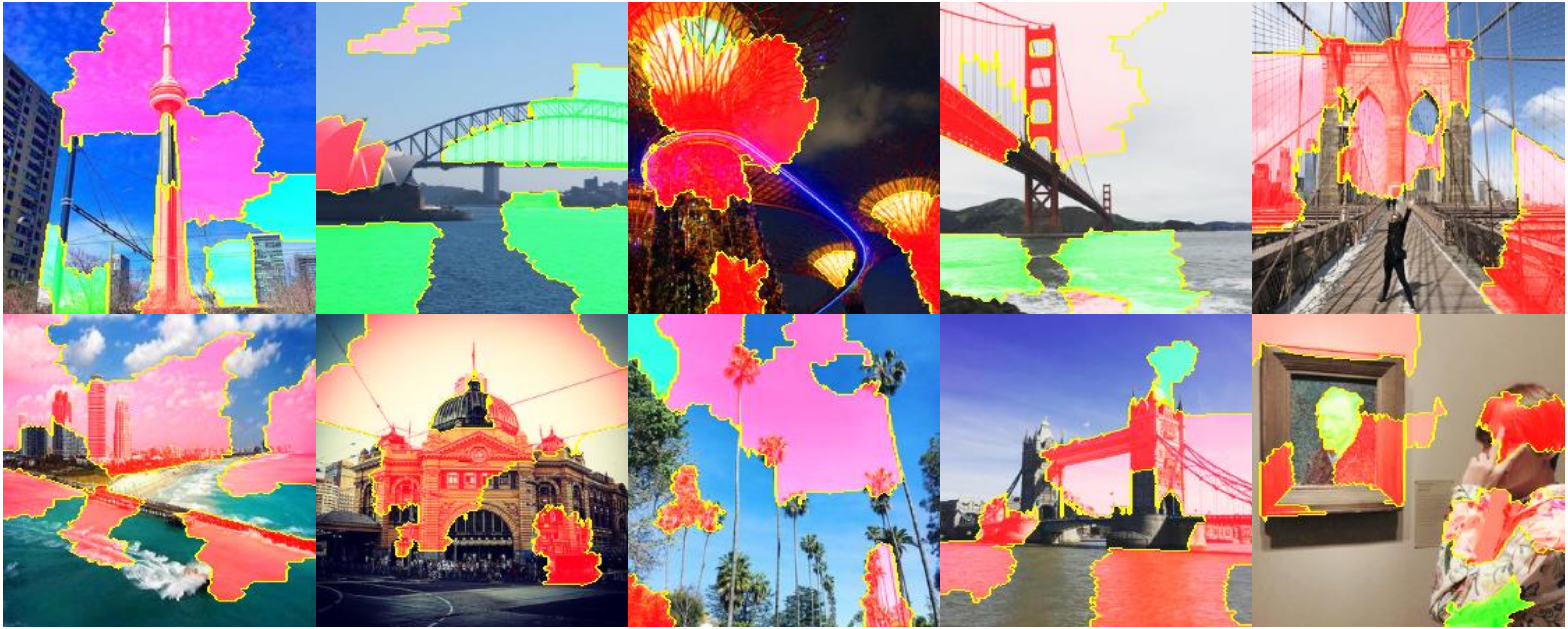
There were 54 errors in total. Only 7 of those were meaningful the other ones depicted either Miami festival or some other music event.



Top 5 – LIME explanation

Top 5 – LIME explanation – experiment 2

Just out of curiosity and to see how LIME works for correctly classified images I have plotted top 5 images for each category.



Ideas for further research

Ideas for further research

- Validation of this setup on a different data set
- Fine tuning of the model
- Creation of a universal recipe for object categorization where no labeled data is available with the use of noisy web data