CSE 803 : Computer Vision
Single Shot Meal Detector (SSMD)

Piyush Gupta Chinmay Dharmatti Avni Sharma

December 1, 2019

Abstract

Meal detection system has plethora of applications ranging from calorie counting and food journal to applications such as food recommender system and health management. Therefore, we develop a single shot meal detection (SSMD) system which is capable of detecting 14 meal classes, while localizing them in the image or video sequence using a bounding box. Specifically, we train a convolution neural network (CNN) based meal detection system which is based on darknet architecture. Our trained classifier is capable of detecting multiple meals along with their bounding box in a single shot with high accuracy. It is capable of inferring on images and video sequences of any frame-size with a typical detection time of around 0.1-0.2s. We use a combination of hand-labelled images along with labelled images from Imagenet and UEC256 food dataset for our training dataset. We train SSMD for 21300 epochs using 7532 images from 14 different meal classes. We show an average true detection rate of 68% and average true detection rate of 98% across all 14 classes on the competition dataset. The highest true detection rate and true rejection rate of 100% each was obtained for class "apple" and "tomato", respectively.

1 Introduction

Modern day computers are getting faster and efficient; allowing automation to perform complex tasks which were once unimaginable. Advancements in artificial intelligence and other learning techniques has led to increased automation, not just for repetitive tasks, but complex tasks that have traditionally been the domain of humans. A core component of the automation is composed of computer vision systems which are capable of object recognition and detection. These systems, which are often powered by artificial intelligence, have led to development of plethora of smart applications which are improving human lives beyond imagination e.g. pedestrian detection for autonomous vehicles, and facial recognition for smart-phone unlocking. One such computer vision system, with has an unlimited potential for improving human lives, is a real-time meal detection system. A fast and accurate meal-detection system can unlock wide range of mobile and computer based applications such as calorie counter, food journal, meal recommender system etc.

It has been proven time and again that convolutional neural networks (CNN) [1], are by far the most superior when it comes to image classification. The network consists of alternating convolutional and pooling layers to extract and combine local features from a two-dimensional input. However, traditional CNNs are not helpful when we want to perform the task of detection. Current detection systems take classifier for the object to be detected, and evaluate it at various scales and locations in the image. In CNN, a kernel is run through the entire image in a sliding window fashion to extract feature maps. This reduces the number of filters to train, and makes classification possible independent of the spacial location of the object. Modern approaches like R-CNN [2] use region proposal methods to generate potential bounding boxes in an image and then run the classifier on these boxes. The R-CNN based approach is slow and hard to optimize because each individual component must be trained separately.

To unlock the true potential of meal detection, the system must be capable of performing fast, and accurate meal detection on multiple meals in an image or a video sequence. Therefore, we develop a single shot meal detector (SSMD) system which is capable of detecting 14 meal classes, while localizing them in the image or video sequence using a bounding box. SSMD is based on darknet architecture, which has been proven to be fast and accurate to perform multi-object detections [3]. A single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation. Since detection pipeline is a single network, end-to-end optimization on detection performance is easier. Contrary to prior work, where classifiers are re-purposed to perform detection, in darknet, object detection is framed as a regression problem to separate bounding boxes and associated class probabilities.

To train SSMD, we use a combination of hand-labelled images along with labelled images from Imagenet [4] and UEC256 [5] food dataset for our training dataset. We train SSMD for 21300 epochs using 7532 images from 14 different meal classes. We show an average true detection rate of 68% and average true detection rate of 98% across all 14 classes on the competition dataset. The highest true detection rate and true rejection rate of 100% each was obtained for class "apple" and "tomato", respectively. We further discuss our training dataset, architecture and results in detail in the subsequent sections.

2 Training Dataset

SSMD is capable of accurately detecting 14 meal classes. These meal classes are: salad, pasta, hotdog, frenchfries, burger, apple, banana, broccoli, pizza, egg, tomato, rice, strawberry, and cookie. In order to train SSMD, we required an image dataset containing these 14 classes along with the bounding box annotations for each class in the image. Specifically, we required a txt label file for each image in the following format: $class_{id}$ x_{center} y_{center} $width_{bb}$ $height_{bb}$. Here, $class_{id}$ is the class number, and x_{center} and y_{center} are the normalized locations of the center of the bounding box normalized with $width$ and $height$ of the image. Similarly, $width_{bb}$ and $height_{bb}$ represents the normalized $width$ and $height$ of the bounding box normalized with $width$ and $height$ of the image. We use a combination of hand-labelled images along with labelled images from Imagenet [4] and UEC256 [5] food dataset for our training dataset.

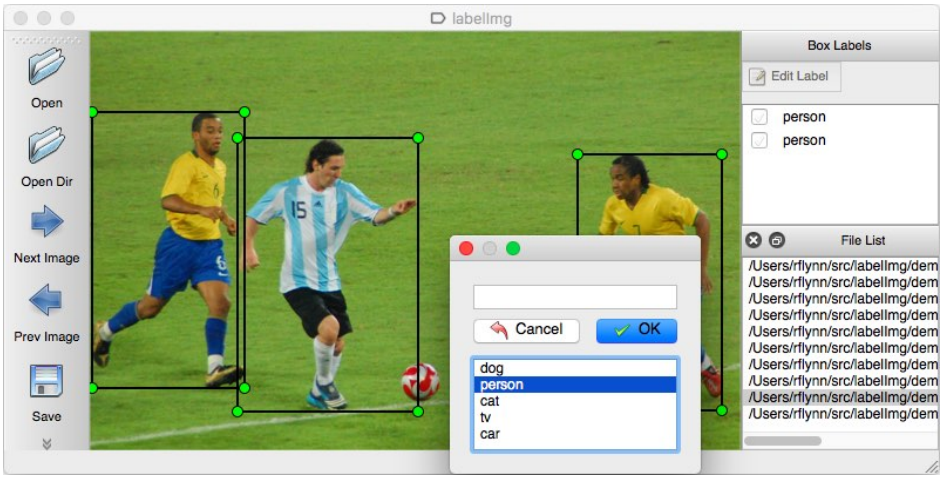


Figure 1: Hand-labelling of images using LabelImg image labeller application

2.1 Imagenet

Imagenet has a large database of images, with a corresponding XLM annotation file based on the Pascal VOC format for the bounding boxes. We utilized the imagenet dataset for the following 8 meal classes: pizza, pasta, hotdog, brocolli, strawberry, banana, burger and apple. The imagenet datset only consisted of green apples and therefore, we generated additional apple dataset by converting all green apples into red apples. Specifically, we swapped the RGB values of green apples to GRB and convereted green apples to red apples. We generated a dataset of around 4200 images for 8 classes using Imagenet database. Finally, we generated python scripts to convert the image annotations from Pascal VOC format to the required format for training.

2.2 UEC 256

UEC-256 is a food database with bounding boxes information of all images provided as a text file. Its a Japanese database and consists of food items which are common in japanese cuisine. We were able to extract data from UEC 256 dataset for the following classes: egg, rice, frenchfries, salad, pizza, burger and hot dog. Since the bounding box files were annotated in a different format, we developed python scripts to convert the labels into the format required to train the network.

2.3 Hand labelled images

Images for "cookie" and "tomato" were not available in either of Imagenet or UEC-256 food dataset. Furthermore, our "egg" dataset only consisted of scrambled eggs and cooked eggs. Therefore, we downloaded images from three classes from google and hand-labelled them using LabelImg, an open-source image labeller app available of github, to label bounding boxes in darknet format. Figure 1 shows the hand-labelling of bounding box using LabelImg application.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 2: SSMD network architecture

3 Single Shot Meal Detector (SSMD)

In this section, we provide the details of the CNN architecture utilized by our single shot meal detector.

3.1 Network Architecture

We utilize the fully convolutional neural network based on darknet architecture to train SSMD. Figure 2 shows the SSMD network architecture based on darknet. It makes use of only convolutional layers, making it a fully convolutional network (FCN). It contains 53 convolutional layers, each followed by batch normalization layer and Leaky ReLU activation. No form of pooling is used, and a convolutional layer with stride 2 is used to downsample the feature maps. This helps in preventing loss of low-level features often attributed to pooling. SSMD is invariant to the size of the input image and can work flawlessly on images as well as video sequences.

3.2 Output

SSMD was trained on the darknet architecture with YOLOv3 algorithm. SSMD, like YOLOv3, results in the following output vector.

$$Y = [p_c,\ b_x,\ b_y,\ b_w,\ b_h,\ c_1, \cdots,\ c_n]$$

where,
p_c = probability of an object in the image
b_x = bounding box center x-coordinate
b_y = bounding box center y-coordinate
b_w = bounding box width
b_h = bounding box height
c_1 = Highest confidence class prediction for object in bounding box
c_2 = Second highest confidence class prediction for object in bounding box
c_n = ... nth highest confidence class prediction for object in bounding box.

The input image is divided into an S x S grid of cells. For each object that is present in the image, one grid cell — the cell where the center of the object falls — is "responsible" for predicting it. Each grid cell predicts B bounding boxes as well as C class probabilities. Each bounding box prediction has 5 components: (x, y, w, h, confidence). The (x, y) coordinates represent the center of the box, relative to the grid cell location. These coordinates are normalized to fall between 0 and 1. The (w, h) box dimensions are also normalized to [0, 1], relative to the image size.

3.3 Loss function

We utilize the loss function utilized by the YOLOv3 algorithm, which is given by:

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbf{1}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] + \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbf{1}_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2]$$
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbf{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbf{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2 + \sum_{i=0}^{S^2} \mathbf{1}_{i}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \tag{1}$$

The five major terms in the loss function are that of bounding box location for each object, bounding box width and height, object confidence score for each bounding box predictor, no-object confidence score for each bounding box predictor, and classification loss, respectively.

4 Performance

SSMD was tested on all the 261 images provided as the competition dataset to evaluate its performance. Following are the results of the SSMD on the competition dataset. Table 1 summarizes the performance of the SSMD detector on the competition dataset. It is capable of detecting all 14 classes, but has the best detection accuracy for apple, burger and brocolli. It has the worst detection rate for pasta and salad.

Table 1: Performance of SSMD on competition dataset

Class X	Total images with class X	Total images without class X	Correct Detections	False Alarm	Correct Rejections	True Detection Rate	True Rejection Rate	Score
salad	22	239	10	1	238	0.455	0.996	0.73
pasta	14	247	3	1	246	0.214	0.996	0.61
hotdog	16	245	13	1	244	0.813	0.996	0.9
frenchfry	18	243	9	2	241	0.5	0.992	0.75
burger	22	239	20	1	238	0.909	0.996	0.95
apple	22	239	22	14	225	1	0.941	0.97
banana	26	235	23	3	232	0.885	0.987	0.94
broccoli	26	235	24	5	230	0.923	0.979	0.95
pizza	23	238	13	2	236	0.565	0.992	0.78
egg	12	249	9	7	242	0.75	0.972	0.86
tomato	12	249	6	0	249	0.5	1	0.75
rice	22	239	15	2	237	0.682	0.992	0.84
strawberry	20	241	17	6	235	0.85	0.975	0.91
cookie	18	243	9	1	242	0.5	0.996	0.75

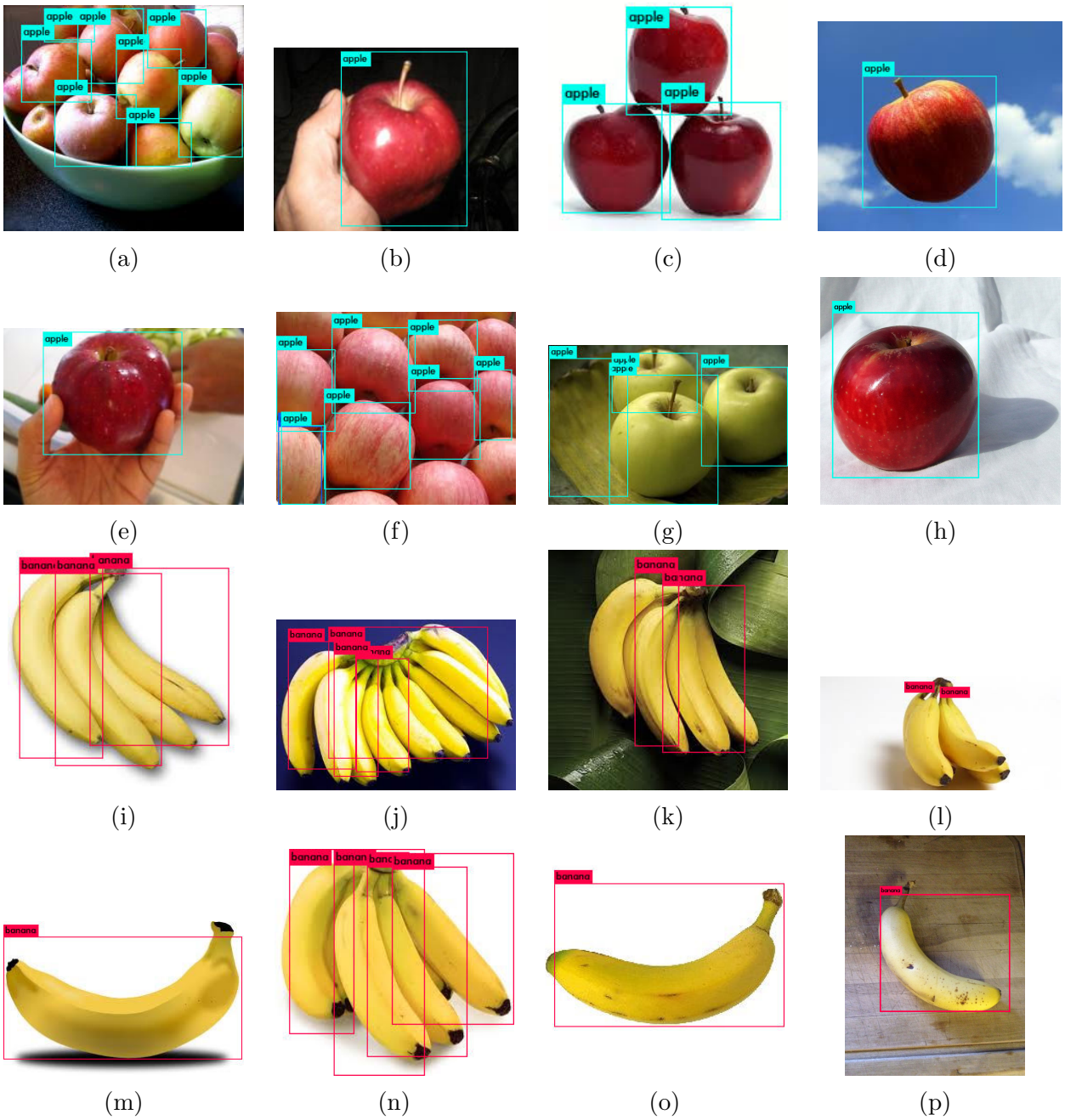


Figure 3: Examples of good meal detection by Single Shot Meal Dectector

(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

Figure 4: Examples of good meal detection by Single Shot Meal Dectector

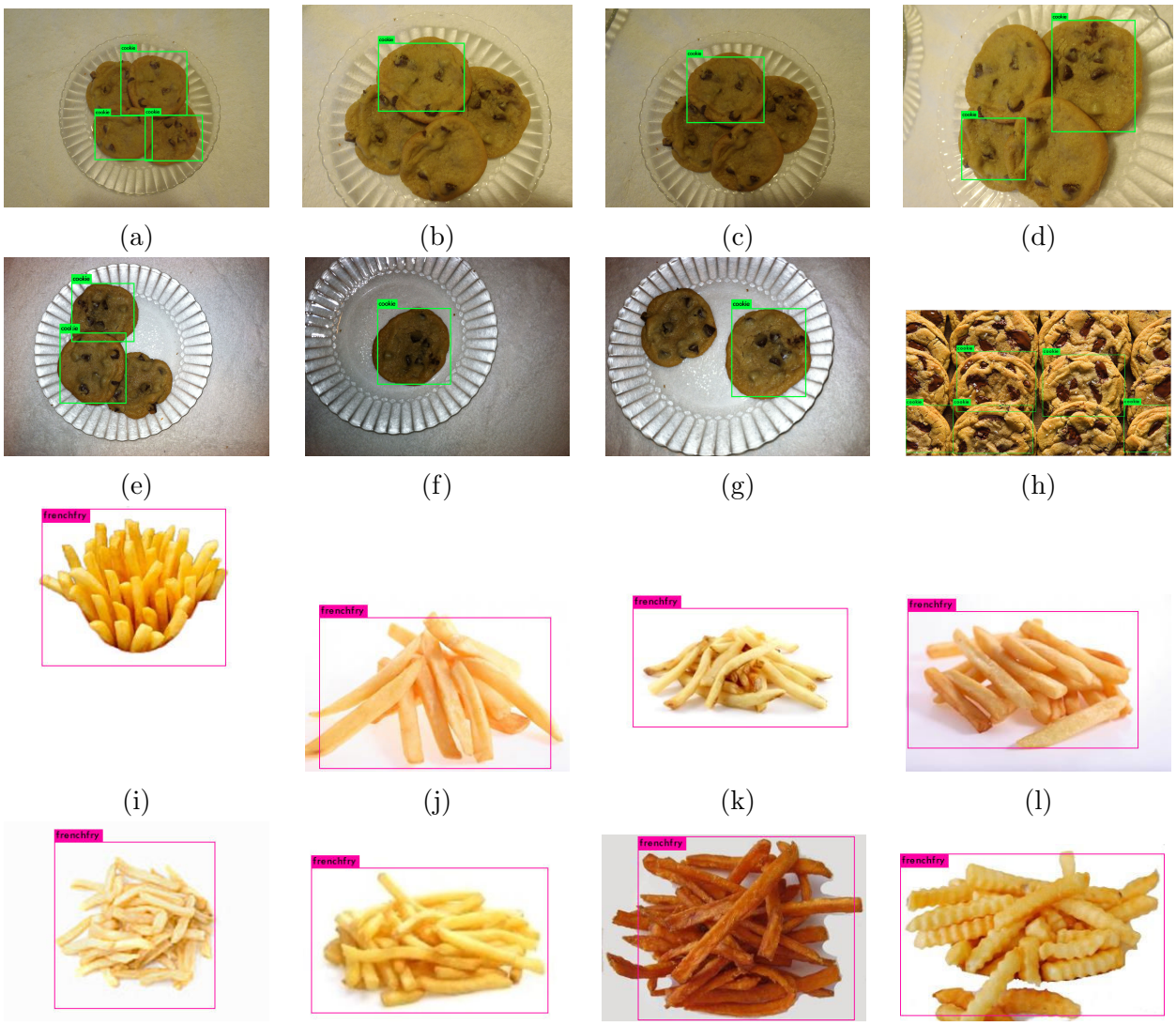
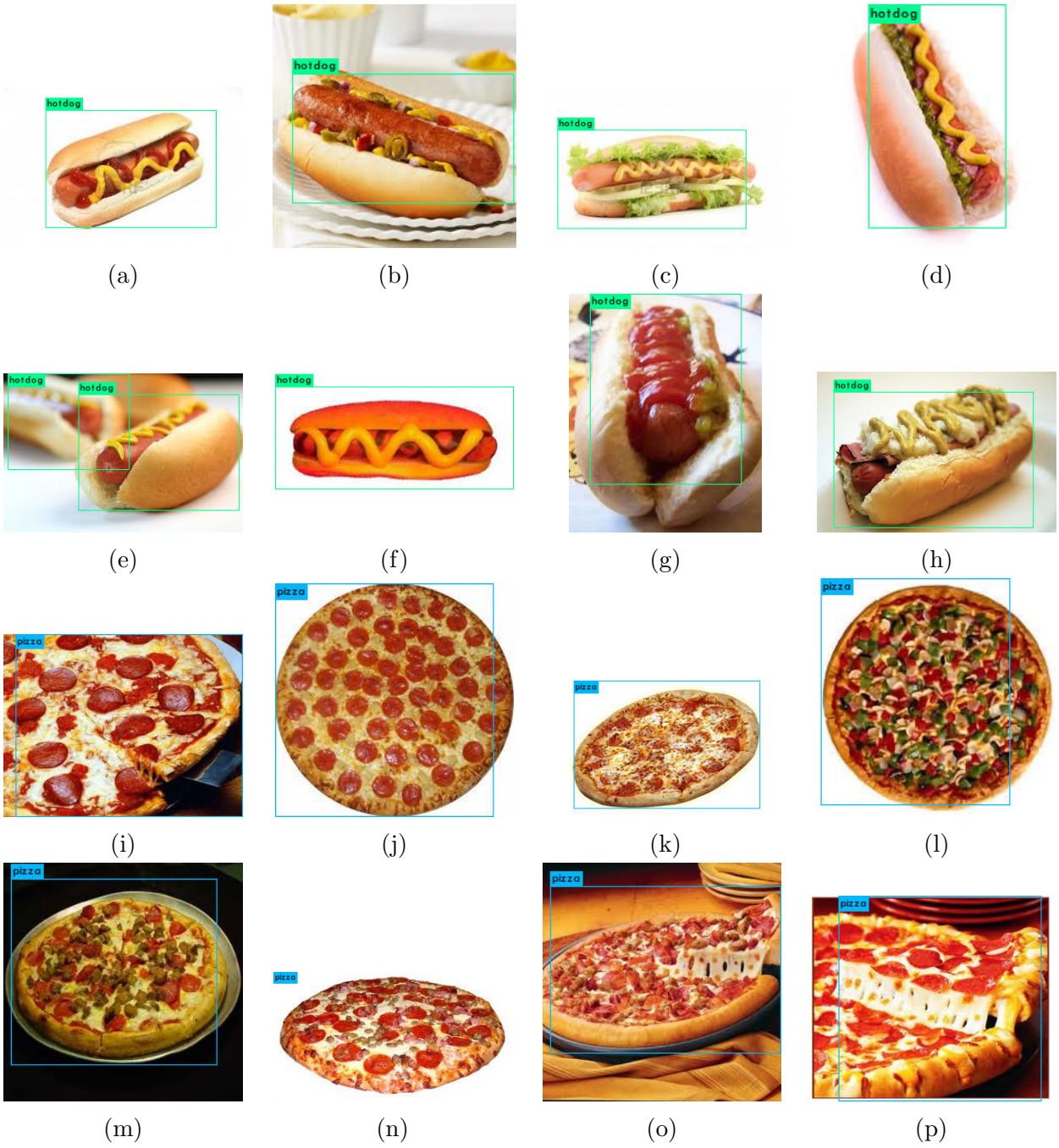


Figure 5: Examples of good meal detection by Single Shot Meal Dectector



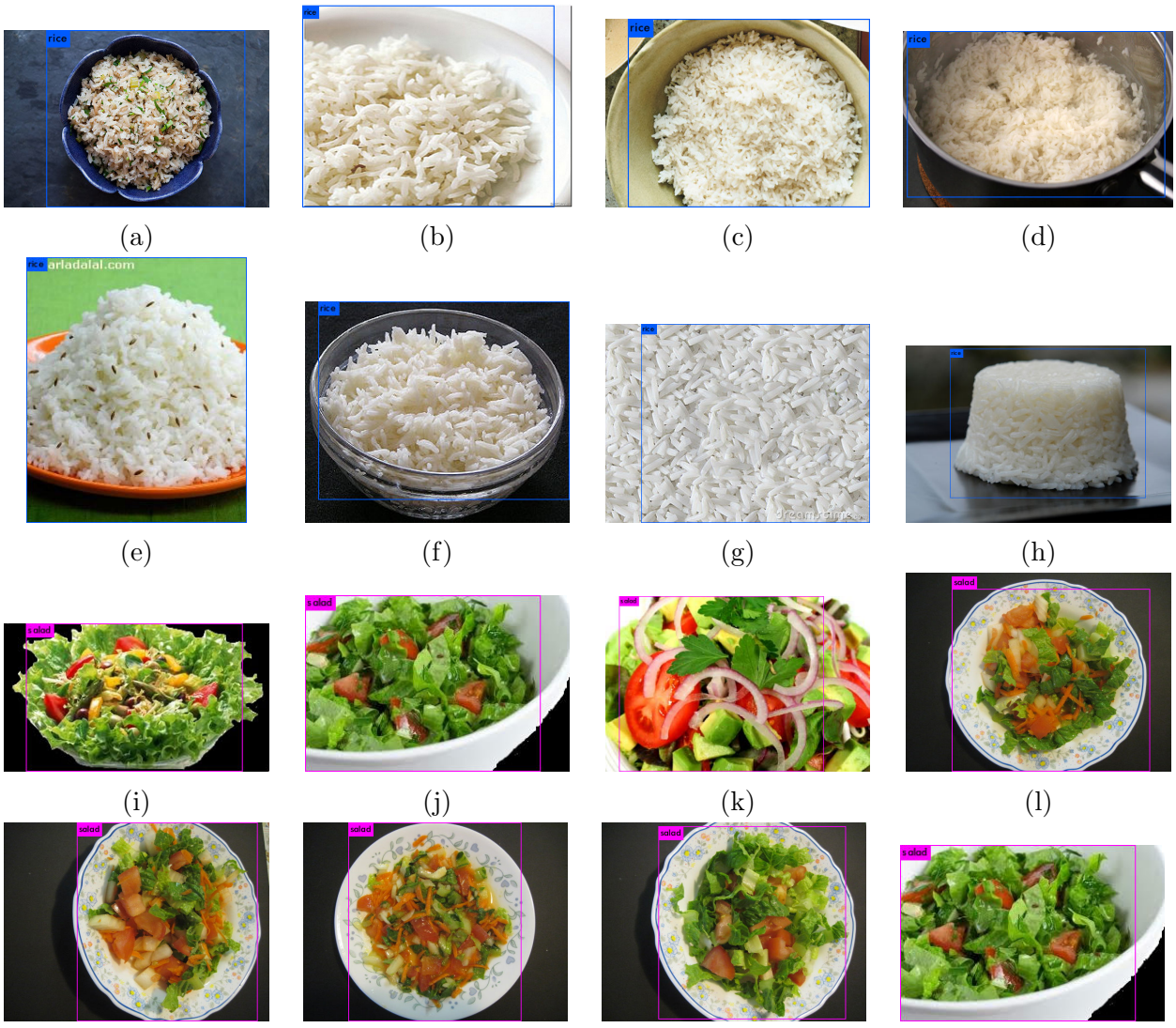
(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

Figure 6: Examples of good meal detection by Single Shot Meal Dectector



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

Figure 7: Examples of good meal detection by Single Shot Meal Dectector

(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

(q) (r) (s) (t)

Figure 8: Examples of good meal detection by Single Shot Meal Dectector

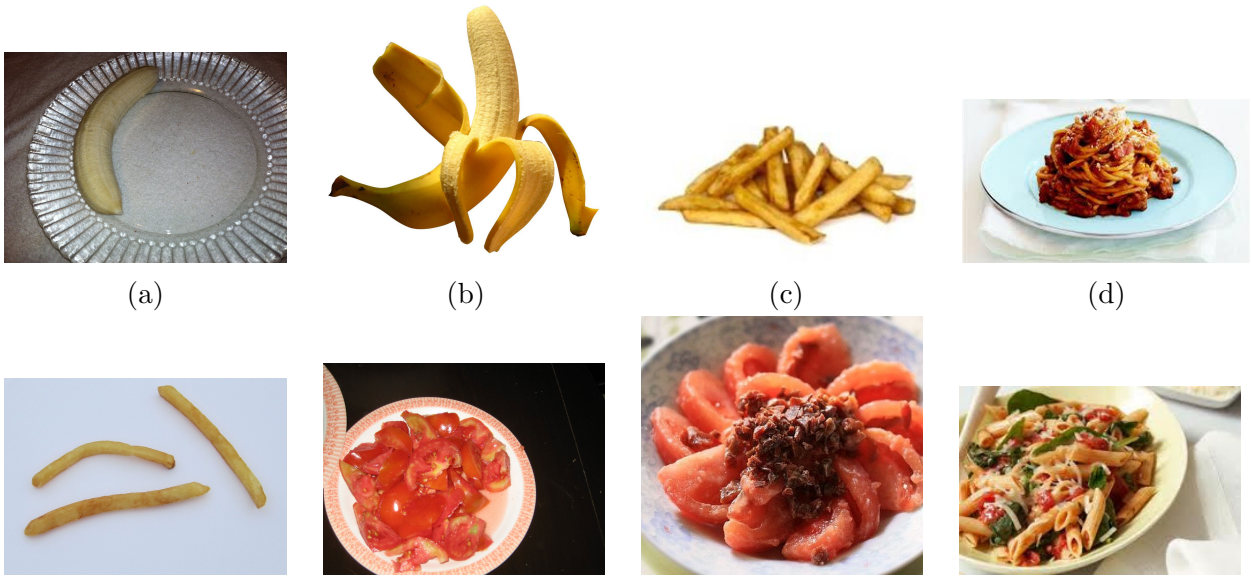
(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

Figure 9: False alarm by Single Shot Meal Dectector



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 10: Missed Detections by Single Shot Meal Dectector

Figure 3, 4, 5, 6, 7,and 8 shows some of the results of the test images from competition dataset on which the trained SSMD performs well and correctly detects the meals in the image. Figure 9 shows some of the examples of the test images from competition dataset on which the trained SSMD does not do well and leads to wrong detection, or false positives. Figure 10 shows examples of missed detections by SSMD.

These results show that SSMD does really well in detecting 14 meal classes. The worst performance was obtained on pasta and tomato. The reason for this is that most of the pasta dataset was obtained from UEC256 food dataset which containes images of Japanese pasta, which mainly includes ramen and noodles. Therefore, it doesn't do very well on many variants of pasta. Furthermore, we hand-labelled tomato dataset and therefore, had least amount of images for it (around 160 images). Furthermore, due to large number of samples for apple (around 1000 images), many tomato images were detected as apple, leading to poor detection of tomato. Some interesting cases include brocolli detected within salad, and single object classified as multiple objects.

5 Conclusions

A single shot meal detector was trained based on darknet architecture. The meal detector was trained using 7532 images for 21300 epochs. It is capable of detecting 14 classes in an image or a video sequence within 0.1-0.2s. SSMD provides the class label, prediction confidence and bounding box for each detected meal object. We show an average true detection rate of 68% and average true detection rate of 98% across all 14 classes on the competition dataset. The highest true detection rate and true rejection rate of 100% each was obtained for class "apple" and "tomato", respectively.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] R. Girshick, "Fast r-cnn," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ser. ICCV '15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 1440–1448. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2015.169

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[5] Y. Kawano and K. Yanai, "Automatic expansion of a food image dataset leveraging existing categories with domain adaptation," in *Proc. of ECCV Workshop on Transferring and Adapting Source Knowledge in Computer Vision (TASK-CV)*, 2014.