Deep Learning para detección de objetos

Vanessa Fernández Martínez
v.fernandezmarti@alumnos.urjc.es
José María Cañas Plaza
jmplaza@gsyc.es
Francisco Rivas
franciscomiguel.rivas@urjc.es

Universidad Rey Juan Carlos Máster de Visión Artificial Introducción a la investigación en VA

Abstract

La detección de objetos es una de las tareas fundamentales en visión artificial. En este documento se presenta el estado del arte sobre la detección de objetos. Se describirán bases de datos para detección de objetos como PASCAL VOC [16], COCO [30] o IMAGENET [18], así como diferentes arquitecturas de redes neuronales de Deep Learning empleadas para el mismo problema, como pueden ser SSD o R-CNN. Además, se presentará una herramienta para aplicaciones de Deep Learning conocida como DetectionSuite.

1 Introducción

Los humanos somos capaces de mirar una imagen y saber al instante qué objetos están la imagen, dónde están situados y cómo interactúan entre ellos. Esto quiere decir que aunque no nos demos cuenta el sistema visual humano es preciso y rápido. Sin embargo, este proceso es más complicado para los computadores.

Uno de los problemas más estudiados en visión artificial es el de la detección de objetos: el proceso de encontrar instancias de objetos del mundo real en imágenes o vídeos. Los algoritmos de detección de objetos usualmente tienen dos fases: (1) extracción de características y (2) detección de instancias de una categoría de objetos. La forma más común de mostrar visualmente la detección de objetos es mediante *bounding box* (cuadro delimitador) que encierran los objetos detectados.

La detección de objetos ha sido ampliamente estudiada en los casos de detección de rostros y de peatones. La detección de objetos tiene aplicaciones en muchas áreas de la visión artificial, incluida la recuperación de imágenes y la videovigilancia.

La parte más difícil para la detección de objetos es la gran varianza existente en las imágenes, ya sea por iluminación, deformación no rígida, oclusión y variabilidad intraclase.

Algunas técnicas clásicas, como el algoritmo Discriminatively Trained Part Based Models (DPM) [20] tienen en cuenta estas varianzas. DPM asume que un objeto está construido

por sus partes. Así, el detector primero encontrará una coincidencia por un filtro raíz más grueso (a media resolución), y luego usará sus modelos de piezas para afinar el resultado. Utiliza las características de HOG en niveles de pirámide antes del filtrado, y SVM lineal para encontrar las diferentes ubicaciones de partes de un objeto.

Otro ejemplo es Integral Channel Features (ICF) [23], que trata de combinar múltiples canales de imágenes registradas calculados mediante transformaciones lineales y no lineales. Esas ocho transformaciones, que incluyen color, gradiente, borde, histograma de gradiente, diferencia de gaussianas, umbralización y el valor absoluto de Gaussiana, todos son invariantes a la traslación, lo que significa que solamente necesitan ser computados una vez. Entrenando estas características mediante el clasificador AdaBoost en cascada, el método es rápido y efectivo en la detección de peatones.

Estos métodos pueden tardar mucho tiempo en la detección. Además de estos métodos "clásicos", en los últimos han surgido algortimos de Deep Learning para la detección de objetos, como SSD [33] o R-CNN [25].

En las próximas subsecciones se describirán los diferetes *dataset* y algoritmos empleados para la detección de objetos, así como la herramienta DetectionSuite, que forma parte del proyecto JdeRobot de software libre para robótica y visión.

2 Bases de Datos para la detección de objetos

La detección de objetos pretende encontrar un determinado objeto en una imagen o en un vídeo. Dado que queremos encontrar el objeto en cuestión bajo diferentes circunstancias, es decir, en diferentes entornos y diferentes iluminaciones, necesitaremos entrenar el modelo con un conjunto de imágenes representativo. Por ello, a lo largo de los últimos años han surgido diferentes *datasets* con el fin de solucionar este problema.

2.1 PASCAL VOC

El desafío PASCAL Visual Object Classes (VOC) [16] es un punto de referencia en el reconocimiento y detección de objetos, proporcionando a la visión artificial y al *Machine Learning* un conjunto de imágenes estándar y procedimientos de evaluación estándar. Este reto se organizó anualmente desde 2005 hasta el 2012.

Es necesario que los conjuntos de datos contengan variabilidad significativa en términos de tamaño del objeto, orientación, pose, iluminación, posición y oclusión. También es importante que los conjuntos de datos no muestren sesgos sistemáticos, por ejemplo, favoreciendo imágenes con objetos centrados o una buena iluminación. Del mismo modo, para garantizar un entrenamiento y una evaluación precisa, es necesario que las anotaciones de imagen sean consistentes, precisas y exhaustivas para las clases especificadas.

En 2007, se llevó a cabo una recolección de imágenes, como las mostradas en la Figura 1, formando el conjunto de datos VOC2007. Este conjunto dispone de dos grandes bases de datos, una de ellas compuesta por un conjunto de validación y otro de entrenamiento, y la otra con un único conjunto de test. Ambas bases de datos contienen alrededor de 5000 imágenes

Figura 1: Ejemplo de imágenes en VOC.

Figura 2: Ejemplo de imágenes en COCO.

en las que se representan, apróximadamente, 12.000 objetos diferentes, por lo que, en total, este conjunto dispone de unas 10000 imágenes en las que se representan unos 24000 objetos. En el año 2012 se modifica este conjunto, aumentando a 11530 el número de imágenes con representación de 27450 objetos diferentes.

2.2 COCO

Microsoft Common Objects in Context (COCO) [30] es un gran conjunto de datos de imágenes diseñado para la detección de objetos, segmentación y generación de subtítulos. En la Figura 2 podemos ver un ejemplo de estas imágenes. Algunas de las características principales de este conjunto de datos son: múltiples objetos en cada imagen, más de 300.000 imágenes, más de 2 millones de instancias, 80 categorías de objetos.

Se utilizan conjuntos de entrenamiento, prueba y validación con sus correspondientes anotaciones. COCO tiene tres tipos de anotaciones: instancias de objeto, puntos clave de objeto y leyendas de imagen, que se almacenan utilizando el formato de archivo JavaScript Object Notation (JSON) y comparten estructura de datos.

2.3 ILSVRC

El desafío, ImageNet Large Scale Visual Recognition Challenge(ILSVRC) [18] es un punto de referencia en la clasificación y detección de objetos en cientos de categorías de objetos y millones de imágenes. Este reto tiene lugar desde el 2010 hasta el presente.

Figura 3: Ejemplo de imágenes en ILSVRC.

El *dataset* ImageNet es el punto central de ILSVRC. ImageNet es un conjunto de imágenes organizado de acuerdo con la jerarquía de WordNet. Cada concepto en WordNet, posiblemente descrito por palabras múltiples o frases de palabras, se llama un *conjunto de sinónimos* o *synset*. ImageNet tiene 21,841 *synsets* de WordNet con un promedio de 650 verificados manualmente e imágenes de resolución completa. Como resultado, ImageNet contiene 14,197,122 imágenes anotadas organizadas por la jerarquía semántica de WordNet. ILSVRC usa un subconjunto de ImageNet imágenes para entrenar los algoritmos y algunos de los protocolos de colección de imágenes de ImageNet para anotar imágenes adicionales para probar los algoritmos. Algunas de estas imágenes las podemos ver en la Figura 3.

2.4 PlantCLEF

El conjunto de datos PlantCLEF [21] 2015 se compone de 113,205 fotos pertenecientes a 41,794 observaciones de 1000 especies de árboles, hierbas y helechos que viven en las regiones de Europa occidental. Esta información fue recolectada por 8.960 contribuidores distintos. Cada imagen pertenece a uno y solo uno de los 7 tipos de vistas reportadas en los metadatos (planta entera, fruta, hoja, flor, tallo, rama, escaneo de hojas) y está asociado con un único identificador de observación de planta que permite vincularlo con las otras imágenes de la misma planta individual.

PlatCLEF tiene las siguientes propiedades: (1) las imágenes de la misma especie provienen de distintas plantas que viven en áreas distintas, (2) las imágenes son tomadas por diferentes usuarios que podrían no utilizar el mismo protocolo de adquisición de imágenes, (3) las imágenes se toman en diferentes períodos del año. Cada imagen del conjunto de datos está asociada a metadatos contextuales (autor, fecha, nombre de la localidad, identificación de la planta) y datos sociales (calificaciones de los usuarios en la calidad de la imagen, nombre de taxón validado colaborativamente, nombre vernáculo). La geolocalización GPS y la configuración del dispositivo están disponibles solamente para algunas de las imágenes.

3 Redes neuronales para la detección de objetos

La detección de objetos no es posible sin un algoritmo. En esta sección se describirán diferentes arquitecturas de redes neuronales empleadas en la detección de objetos.

3.1 R-CNN, Fast R-CNN y Faster-RCNN

El algoritmo Region-based Convolutional Network (R-CNN) [25] logró una gran mejora en el desafío de la detección de objetos. Consta de 3 pasos simples: (1) escaneo de la imagen de entrada para detectar posibles objetos usando un algoritmo llamado búsqueda selectiva, generando aproximadamente 2000 propuestas de región; (2) ejecución de una red neuronal convolucional (CNN) en la parte superior de cada una de estas propuestas de región; (3) tomar la salida de cada CNN y alimentarla en a) un SVM para clasificar la región y b) un regresor lineal para apretar el *bounding box* del objeto, si tal objeto existe.

El método Fast Region-based Convolutional Network (Fast R-CNN) [6] se basa en Region-based Convolutional Network, el trabajo previo para clasificar de manera eficiente objetos utilizando redes convolucionales profundas.

Fast R-CNN emplea varias innovaciones para mejorar el entrenamiento y la velocidad de prueba, y además aumenta la precisión de detección. Estas mejoras se consiguen principalmente por dos modificaciones respecto a R-CNN: (1) realización de extracción de características sobre la imagen antes de proponer regiones, por lo tanto, solo se ejecuta una CNN sobre toda la imagen en lugar de 2000 CNN en más de 2000 regiones superpuestas, (2) se reemplaza el SVM con una capa de softmax, extendiendo así la red neuronal para las predicciones en lugar de crear un nuevo modelo.

Posteriormente se desarrolló Faster R-CNN [28]. La idea principal de Faster R-CNN fue reemplazar el lento algoritmo de búsqueda selectiva con una red neuronal rápida. Específicamente, se introdujo la *region proposal nerwork* (RPN). Una RPN es una red completamente convolucional que predice simultáneamente límites de objeto y puntuaciones de objetividad en cada posición. Las RPN están entrenadas de extremo a extremo para generar propuestas de la región de calidad, que son utilizadas por Fast R-CNN para la detección. Con una optimización alterna simple, RPN y Fast R-CNN pueden ser entrenados para compartir características convolucionales.

3.2 YOLO

You Only Look Once (YOLO) [9] es otro enfoque para la detección de objetos. El trabajo previo a YOLO designa clasificadores para realizar la detección. En este trabajo, se enmarca la detección de objetos como un problema de regresión en *bounding boxes* espacialmente separados y probabilidades de clase asociadas. Una red neuronal única predice *bounding box* y probabilidades de clase directamente desde imágenes completas en una evaluación. Como todo el *pipeline* de la detección es una red única, se puede optimizar de extremo a extremo directamente en el rendimiento de detección.

La arquitectura unificada de YOLO (Figura 4) es extremadamente rápida, procesando imágenes en tiempo real a 45 *frames* por segundo. Una versión más pequeña de la red, Fast

Figura 4: Sistema de detección YOLO.

YOLO, procesa 155 *frames* por segundo, logrando duplicar el mAP (mean Average Precision) de otros detectores en tiempo real.

En comparación con los sistemas de detección de última generación, YOLO comete más errores de localización pero es mucho menos probable que prediga detecciones falsas donde no existe nada. Finalmente, YOLO aprende representaciones muy generales de objetos. YOLO supera a todos los demás métodos de detección, incluidos DPM y R-CNN, por un amplio margen al generalizar de imágenes naturales a obras de arte.

3.3 R-FCN

Region-based Fully Convolutional Networks (R-FCN) [7] es creada para obtener una detección de objetos precisa y eficiente. A diferencia de los detectores que se basan en región, como Fast o Faster R-CNN, que aplican una costosa subred por región cientos de veces, en R-FCN el detector basado en la región es completamente convolucional con casi todos los cálculos compartidos en toda la imagen.

R-FCN comparte el 100% de los cálculos en cada salida individual. Al ser completamente convolucional, se encontró con un problema en el diseño del modelo. Por un lado, al realizar la clasificación de un objeto, queremos aprender la invarianza de ubicación en un modelo: independientemente de dónde aparezca el gato en la imagen, queremos clasificarlo como un gato. Por otro lado, cuando realizamos la detección del objeto, queremos aprender la varianza de ubicación: si el gato está en la esquina superior izquierda, queremos dibujar un cuadro en la esquina superior izquierda. Entonces, si estamos tratando de compartir cálculos convolucionales en el 100% de la red, ¿ cómo comprometemos la invarianza de ubicación y la varianza de ubicación?.

La solución que aportó R-FCN es usar mapas de puntuación sensibles a la posición. Cada mapa de puntuación sensible a la posición representa una posición relativa de una clase de objeto. Por ejemplo, un mapa de puntuación puede activarse donde detecta la parte superior derecha de un gato. Otro mapa de puntuación puede activarse donde se ve la esquina inferior izquierda de un automóvil. Esencialmente, estos mapas de puntuación son mapas de características convolucionales que han sido entrenados para reconocer ciertas partes de cada objeto.

R-FCN funciona de la siguiente manera (Figura 5): (1) ejecuta una CNN (ResNet) sobre la imagen de entrada; (2) agrega una capa completamente convolucional para generar un banco de puntuación de los “mapas de puntuación sensibles a la posición” antes mencionados

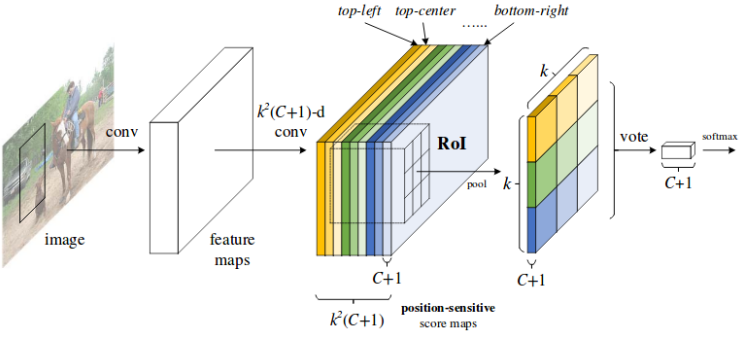

Figura 5: Estructura de R-FCN para detección de objetos

(debería haber mapas de puntuación k^2(C+1), con k^2 representando el número de posiciones relativas para dividir un objeto y C+1 representa el número de clases más el fondo); (3) ejecuta una red de propuesta de región completamente convolucional (RPN) para generar regiones de interés (ROI); (4) para cada ROI, la divide en los mismos k^2 "bins" o subregiones que los mapas de puntuación; (5) para cada *bin*, verifica el banco de puntuaciones para ver si ese *bin* coincide con la posición correspondiente de algún objeto y repite el proceso para cada clase; (6) una vez que cada uno de los k^2 *bins* tiene un valor de "coincidencia de objetos" para cada clase, promedia los *bins* para obtener una puntuación individual por clase; (7) clasifica la ROI con una softmax sobre el vector dimensional C+1 restante. Como resultado, R-FCN es varias veces más rápido que Faster R-CNN, y logra una precisión comparable.

3.4 SSD

Single-Shot Detector (SSD) [33] es un método para detectar objetos en imágenes usando una sola red neuronal profunda. SSD proporciona una gran ganancia de velocidad frente a Faster R-CNN.

Faster R-CNN realiza propuestas de región y clasificaciones de región en dos pasos separados. Primero, usa una red de propuesta de región para generar regiones de interés; luego, usa capas conectadas completamente o capas convolucionales sensibles a la posición para clasificar esas regiones. SSD (Figura 6) lleva a cabo estos dos pasos en uno solo, prediciendo simultáneamente el cuadro delimitador y la clase a medida que procesa la imagen.

Concretamente, dada una imagen de entrada y un conjunto de etiquetas de *ground truth*, SSD hace lo siguiente: (1) pasa la imagen a través de una serie de capas convolucionales, produciendo varios conjuntos de mapas de características a diferentes escalas; (2) para cada ubicación en cada uno de estos mapas de características, usa un filtro convolucional de 3x3 para evaluar un pequeño conjunto de cuadros delimitadores por defecto; (3) para cada recuadro, predice simultáneamente a) el desplazamiento del cuadro delimitador y b) las probabilidades de clase; (4) durante el entrenamiento, hace coincidir la (*bounding box*) de *ground truth* con estas cajas predichas según IoU intersection over union). La mejor (*bounding box*) pronosticada se etiquetará como "positiva", junto con todas las demás (*bounding boxes*) que

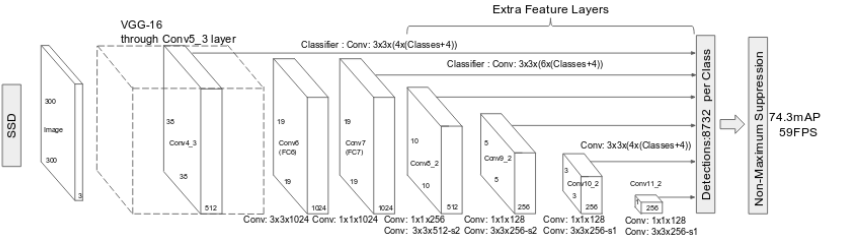

Figura 6: Arquitectura del modelo SSD.

tengan una Id con *truth*> 0.5.

SSD suena sencillo, pero el entrenamiento tiene un desafío único. Clasificamos y estimamos (*bounding boxes*) desde cada posición en la imagen, usando múltiples formas diferentes, en diferentes escalas. Como resultado, generamos un número mucho mayor de cuadros delimitadores que en otros modelos, y casi todos ellos son ejemplos negativos.

Para solucionar este desequilibrio, SSD hace dos cosas. En primer lugar, utiliza la *non-maximum suppression* (NMS) para agrupar cuadros altamente superpuestos en un solo cuadro. En otras palabras, si cuatro cajas de formas, tamaños, etc. similares contienen el mismo perro, el NMS conservará el que tenga la mayor confianza y descartará el resto. En segundo lugar, el modelo usa una técnica llamada minería negativa para equilibrar las clases durante el entrenamiento. En la minería negativa dura, solo se utiliza un subconjunto de los ejemplos negativos con la mayor pérdida de entrenamiento (es decir, falsos positivos) en cada iteración de entrenamiento. SSD mantiene una relación de 3: 1 de negativos a positivos.

3.5 Mask R-CNN

Mask R-CNN [10] extiende el concepto de Faster R-CNN al agregar una rama para predecir las máscaras de segmentación en cada región de interés (ROI), en paralelo a la rama existente para la regresión de la clasificación y el *bounding box*. La rama de la máscara es una pequeña FCN aplicada a cada ROI, prediciendo una máscara de segmentación píxel a píxel. La generación de máscaras para cada clase se realiza de forma que las clases no compitan entre ellas. Esto permite separar la máscara y la predicción de la clase del objeto.

Mask R-CNN es simple de implementar y entrenar dado el marco Faster R-CNN, que facilita una amplia gama de diseños de arquitectura flexible. Adicionalmente, la rama de la máscara solamente agrega una pequeña carga computacional, permitiendo un sistema rápido y una experimentación rápida. En principio, Mask R-CNN es una extensión intuitiva de Faster R-CNN, sin embargo, construir la rama de máscara correctamente es fundamental para obtener buenos resultados.

Faster R-CNN no fue diseñado para la alineación de píxel a píxel entre las entradas y salidas de la red. Para corregir la desalineación, Mask R-CNN propone una capa simple, libre de cuantización, llamada RoIAlign, que conserva fielmente las ubicaciones espaciales exactas.

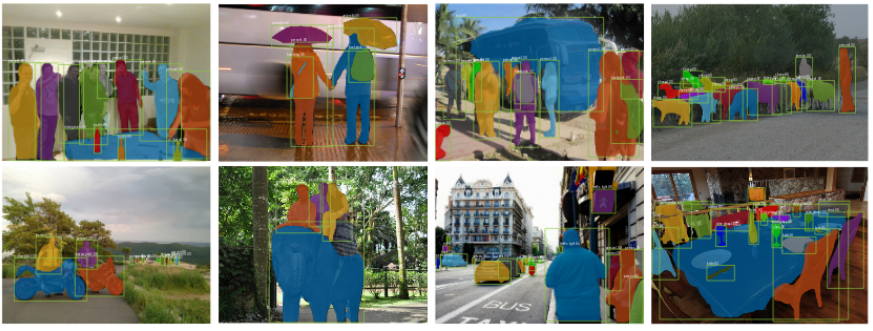

Figura 7: Resultados de Mask R-CNN.

4 Herramienta DetectionSuite

JdeRobot [1] es un middleware de software libre para el desarrollo de aplicaciones con robots y visión artificial. DeepLearning Suite [24] es un conjunto de utilidades de JdeRobot que simplifica la evaluación de los conjuntos de datos de detección de objetos más comunes con varias redes neuronales de detección de objetos. La idea de esta herramienta es ofrecer una infraestructura genérica para evaluar los algoritmos de detección de objetos contra un conjunto de datos y calcular las estadísticas más comunes: *Intersecion Over Union*, *precision*, *recall*.

DetectionSuite tiene 5 funcionalidades principales: (1) generación y etiquetado de muestras, (2) importar *datasets* variados, (3) ejecutar una red, (4) entrenar una red, y (5) evaluar una red mediante el cálculo de estadísticos de su calidad y rendimiento.

DeepLearning Suite soporta los formatos de *dataset*: YOLO, *JdeRobot recorder logs*, *Princeton RGB dataset*, *Spinello dataset*. Además, soporta los *frameworks*/algoritmos de detección de objetos: YOLO (darknet), y *Background Substraction*.

Esta herramienta cuenta con *Sample Generation Tool*, que ha sido desarrollado para simplificar el proceso de generación de muestras para conjuntos de datos enfocados en la detección de objetos. Las utilidades proporcionan algunas características para reducir el tiempo de etiquetado de objetos como rectángulos.

5 Conclusiones

En este estado del arte hemos visto diferentes *datasets*, así como diferentes algoritmos empleados para la detección de objetos. Unos obtendrán mejores resultados que otros en determinadas imágenes o con determinados objetos, así como unos tendrán mayor velocidad de cómputo que otros. Además, hemos visto la herramienta de DetectionSuite JdeRobot, que soporta algún *dataset* y el *framework* de YOLO.

[1] http://jderobot.org/Main_Page

El TFM que se propone es la mejora de esta última herramienta [24]. En concreto, se pretende que sea válida para numerosos conjuntos de datos como pueden ser COCO [30], PASCAL VOC [16], etc; así como se pretende que la herramienta se pueda usar con otros *frameworks* como pueden ser Keras, Caffe, TensorFlow, etc. Además, el objetivo es que la herramienta cuente con numerosas métricas de evaluación.

Esta herramienta pretende facilitar al usuario el empleo de diferentes *frameworks* de Deep Learning para la detección de objetos. Por ahora, ha sido probada para la detección de personas, por lo que se podría emplear en un sistema de seguridad, así como extenderla para la detección de más tipos de objetos.

Referencias

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. *CVPR*, 2017.

[2] Wei Liu Cheng-Yang Fu, Ambrish Tyagi Ananth Ranga, and Alexander C. Berg. Dssd : Deconvolutional single shot detector. *CoRR*, 2017.

[3] Alexander Toshev Christian Szegedy and Dumitru Erhan. Deep neural networks for object detection. *NIPS*, 2013.

[4] Li Deng and Dong Yu. Deep learning: Methods and applications. https://pdfs.semanticscholar.org/9260/a65f84384b0b12b32810e171a3c1cc0c0ffa.pdf, 2014.

[5] Christian Szegedy Dumitru Erhan and Dragomir Anguelov Alexander Toshev. Scalable object detection using deep neural networks. *IEEE*, 2014.

[6] Ross Girshick. Fast r-cnn. *ICCV*, 2016.

[7] Kaiming He Jifeng Dai, Yi Li and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. *NIPS*, 2016.

[8] Hyojin Park Jisoo Jeong and Nojun Kwak. Enhancement of ssd by concatenating feature maps for object detection. *BMVC*, 2017.

[9] Ross Girshick Joseph Redmon, Santosh Divvala and Ali Farhadi. You only look once: Unified, real-time object detection. *CVPR*, 2016.

[10] Piotr Dollár Kaiming He and Georgia Gkioxari Ross Girshick. Mask r-cnn. *CoRR*, 2017.

[11] Xiangyu Zhang Kaiming He and Jian Sun Shaoqing Ren. Deep residual learning for image recognition. *IEEE*, 2015.

[12] Rob Fergus Li Fei-Fei and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *CVPRW*, 2004.

[13] David Eigen Li Wan and Rob Fergus. End-to-end integration of a convolutional network, deformable parts model and non-maximum suppression. *CVPR*, 2014.

[14] Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. *IEEE*, 2015.

[15] Mohammad Rastegari Mahyar Najibi and Larry S. Davis. G-cnn: an iterative grid based object detector. *CVPR*, 2016.

[16] Luc Van Gool Mark Everingham, John Winn Christopher K. I. Williams, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2009.

[17] HIEU T. NGUYEN and ARNOLD W. M. SMEULDERS. Robust tracking using foreground-background texture discrimination. *IJCV*, 2006.

[18] Jia Deng* Olga Russakovsky*, Jonathan Krause Hao Su, Sean Ma Sanjeev Satheesh, Andrej Karpathy Zhiheng Huang, Michael Bernstein Aditya Khosla, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[19] Constantine Papageorgiou and Tomaso Poggio. A trainable system for object detection. *IJCV*, 2000.

[20] Ross B. Girshick Pedro F. Felzenszwalb and Deva Ramanan David McAllester. Object detection with discriminatively trained part-based models. *IEEE*, 2010.

[21] Alexis Joly Pierre Bonne and Hervé Goëau. Plant identification in an open-world (lifeclef 2016). *CLEF*, 2016.

[22] David Eigen Pierre Sermanet, Michael Mathieu Xiang Zhang, and Yann LeCun Rob Fergus. Overfeat: Integrated recognition, localization and detection using convolutional networks. *ICLR*, 2014.

[23] Zhuowen Tu Piotr Dollár and Serge Belongie Pietro Perona. Integral channel features. *BMVA*, 2009.

[24] Francisco Rivas. Detectionsuite. http://jderobot.org/index.php/Tools#DetectionSuite, 2017.

[25] Trevor Darrell Ross Girshick and Jitendra Malik. Region-based convolutional networks for accurate object detection and segmentation. *ICCV*, 2016.

[26] Henrik I. Christensen Samarth Brahmbhatt and James Hays. Stuffnet: Using 'stuff' to improve object detection. *WACV*, 2017.

[27] Byungseok Roh Sanghoon Hong, Yeongjae Cheon Kye-Hyeon Kim, and Minje Park. Pvanet: Lightweight deep neural networks for real-time object detection. *CVPR*, 2016.

[28] Kaiming He Shaoqing Ren and Jian Sun Ross Girshick. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEE*, 2017.

[29] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.

[30] James Hays Tsung-Yi Lin, Pietro Perona Michael Maire, Lubomir Bourdev Serge Belongie, Deva Ramanan, Ross Girshick Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. *ECCV*, 2015.

[31] Piotr Dollár Tsung-Yi Lin, Kaiming He Ross Girshick, and Serge Belongie Bharath Hariharan. Feature pyramid networks for object detection. *CVPR*, 2017.

[32] Xiaogang Wang Wanli Ouyang, Ping Luo Xingyu Zeng, Shi Qiu, Hongsheng Li Yonglong Tian, Zhe Wang Shuo Yang, and Xiaoou T ang Chen-Change Loy. Deepid-net: Deformable deep convolutional neural networks for object detection. *CVPR*, 2015.

[33] Dragomir Anguelov Wei Liu, Christian Szegedy Dumitru Erhan, Cheng-Yang Fu Scott Reed, and Alexander C. Berg. Ssd: Single shot multibox detector. *ECCV*, 2016.

[34] Quanfu Fan Zhaowei Cai and Nuno Vasconcelos Rogerio S. Feris. A unified multi-scale deep convolutional neural network for fast object detection. *CVPR*, 2016.