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

Resumen

La detección de objetos es una de las tareas fundamentales en visión artificial. En este documento se presenta el estado del arte para la detección de objetos. Se abordarán bases de datos para detección de objetos como PASCAL VOC, así como diferentes algortimos de Deep Learning empleados para el mismo problema, como pueden ser CNN o RCNN. 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 usan características extraídas y algoritmos de aprendizaje para reconocer instancias de una categoría de objetos. La forma más común de mostrar visualmente la detección de objetos es mediante *bunding box* que encierran los objetos detectados.

La detección de objetos ha sido ampliamente estudiada en los casos de detección de rostros y detección 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 más clásicas, como el algoritmo Discriminatively Trained Part Based Models (DPM) 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 el entrenamiento para encontrar los diferentes ubicaciones de partes de un objeto.

Otro ejemplo es Integral Channel Features (ICF), 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. Por ello, surgieron algortimos de Deep Learning para la detección de objetos, como CNN o RCNN.

2. Estado del arte

En las próximas subsecciones se describirán los diferetes dataset y algoritmos empleados para la detección de objetos, así como la herramienta DetectionSuite.

2.1. 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.

2.1.1. PASCAL VOC

El desafío PASCAL Visual Object Classes (VOC) [**?**] es un punto de referencia en el reconocimiento y detección de objetos, proporcionando a la visión artificial y al *Matching Learning* un conjunto de imágenes estándar y procedimientos de evaluación estándar. Este reto es organizado 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.

Figura 1: Ejemplo de imágenes en VOC.

Figura 2: Ejemplo de imágenes en COCO.

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 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 reperesentan unos 24000 objetos. En el año 2012 se modifica este conjunto, aumentando a 11530 el número de imágenes con representación de 27450 objetos diferentes.

2.1.2. COCO

Microsoft Common Objects in Context (COCO) 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 caracterısticas 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.

Figura 3: Ejemplo de imágenes en ILSVRC.

2.1.3. ILSVRC

El desafío ImageNet Large Scale Visual Recognition Challenge(ILSVRC) 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.

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 3.

2.2. Algoritmos para la detección de objetos

La detección de objetos no es posible sin un algoritmo. En las próximas secciones se describirán diferentes algoritmos empleados en la detección de objetos.

2.2.1. RCNN

En los últimos años las redes neuronales convolucionales (CNN) han tenido mucho éxito. Las CNN comparten muchas propiedades con el sistema visual del cerebro. Una diferencia importante es que CNN es típicamente una arquitectura de feed-forward mientras que en el sistema visual las conexiones recurrentes son abundantes.

Inspirándose en este hecho, se crearon las CNN recurrentes (RCNN) para reconocimiento de objetos incorporando conexiones recurrentes en cada capa convolucional. Aunque la entrada es estática, las actividades de las unidades RCNN evolucionan con el tiempo de mo-

do que la actividad de cada unidad se modula por las actividades de sus unidades vecinas. Esta propiedad mejora la capacidad del modelo para integrar la información de contexto, que es importante para el reconocimiento de objetos.

Al igual que otras redes neuronales recurrentes, desplegar las RCNN a través del tiempo puede dar lugar a una red arbitrariamente profunda con un número fijo de parámetros. Además, la red desplegada tiene múltiples rutas, lo que puede facilitar el proceso de aprendizaje.

Por tanto, las RCNN son capaces de localizar y detectar objetos en imágenes. El resultado generalmente es un conjunto de cuadros delimitadores que coinciden estrechamente con cada uno de los objetos detectados, así como una salida de clase para cada objeto detectado.

2.2.2. R-CNN, Fast R-CNN y Faster-RCNN

El algoritmo Region-based Convolutional Network (R-CNN) 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 cuadro delimitador del objeto, si tal objeto existe.

El método Fast Region-based Convolutional Network (Fast R-CNN) 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. La idea principal de Faster R-CNN fue reemplazar el lento algoritmo de búsqueda selectiva con una red neural rápida. Específicamente, 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.

Figura 4: Sistema de detección YOLO.

2.2.3. YOLO

You Only Look Once (YOLO) es otro enfoque para la detección de objetos. El trabajo previo a YOLO designa clasificadores para realizar la detección. En cambio, se enmarca la detección de objetos como un problema de regresión en cuadros delimitadores 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. Desde 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 YOLO, procesa 155 *frames* por segundo, logrando duplicar el mAP 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.

2.2.4. R-FCN

Region-based Fully Convolutional Networks (R-FCN) es creada para obtener una detección de objetos precisa y eficiente. A diferencia de los detectores anteriores basados en región, como Fast/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 único 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

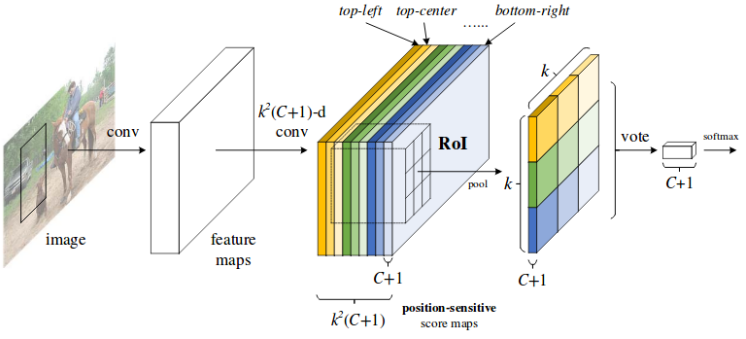

Figura 5: Estructura de R-FCN para detección de objetos

.

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 (debería haber mapas de puntuación k (C + 1), con k 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 "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 bins tiene un valor de çoincidencia 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.

2.2.5. SSD

Single-Shot Detector(SSD) 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,

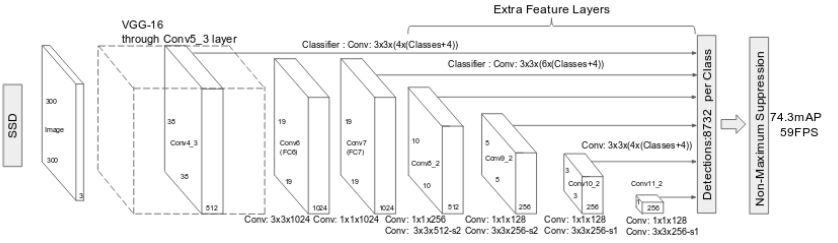

Figura 6: Arquitectura del modelo SSD.

usar 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) curante el entrenamiento, hace coincidir la caja (*box*) de ground truth con estas cajas predichas según IoU. La mejor caja pronosticada se etiquetará como "positiva", junto con todas las demás cajas que tengan una Id con *truth*>0.5.

SSD suena sencillo, pero el entrenamiento tiene un desafío único. Clasificamos y dibujamos cuadros delimitadores 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 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.

2.2.6. FPN

Las pirámides de características son un componente básico en los sistemas de reconocimiento para detectar objetos a diferentes escalas. Pero reciente los detectores de objetos de *Deep Learning* han evitado la representación de pirámides, en parte porque son de cálculo y memoria intensivos.

Feature Pyramid Network (FPN) explota la multi-escala inherente, jerarquía piramidal de redes convolucionales profundas para construir pirámides de características con un costo adicional marginal. Es una arquitectura top-down con conexiones laterales, que se desarrolla para construir mapas semánticos de características de alto nivel en todas las escalas.

FPN toma una imagen de una escala de un tamaño arbitrario como entrada, y genera mapas de características de tamaño proporcional en múltiples niveles, de forma totalmente convolucional. Este proceso es independiente de las arquitecturas convolucionales de *backbone*, y se usan ResNets. La construcción de la pirámide implica un camino *bottom-up*, un camino *top-down* y conexiones laterales.

El camino de *bottom-up* es el cálculo *feed-forward* de la red troncal ConvNet, que calcula una jerarquía de características que consiste en mapas de características a varias escalas con un paso de escalado de 2. Para la pirámide característica, se define un nivel de pirámide para cada etapa. Se elige la salida de la última capa de cada etapa como el conjunto de referencia de mapas de características, que se enriquece para crear nuestra pirámide.

El camino *top-down* obtiene características de mayor resolución por *upsampling* espacial, pero semánticamente más fuertes, cuentan con mapas de niveles de pirámides más altos. Estas características son después mejoradas con características de la ruta *bottom-up* a través de conexiones laterales. Cada conexión lateral combina mapas de características del mismo tamaño espacial del camino *bottom-up* y el camino *top-down*. El mapa de características *bottom-up* es de semántica de nivel inferior, pero sus activaciones se localizan con mayor precisión ya que se submuestreó menos veces.

2.3. DetectionSuite

DeepLearning Suite es un conjunto de herramientas 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*.

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 herramientas proporcionan algunas características para reducir el tiempo de etiquetado de objetos como rectángulos.

3. Conclusiones

En el 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 una herramienta de JdeRobot, que soporta algún dataset y el *framework* de YOLO.

El TFM que se propone es la mejora de la herramienta DetectionSuite. En concreto, se pretende que esta herramienta sea válidad para numerosos conjuntos de datos como pueden ser COCO, PASCAL VOC, 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 méticas de evaluación.

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

4. Referencias

Referencias