Neurophysiologist first introduced the idea of neurons[1]. Later, in 1949 Donald Hebb suggested that connection between neuron increases, when fired at the same time. Although after a failed attempt by IBM, Bernard Widrow and Marcian Hoff in 1962 were able to design a simple binary pattern recognizer that could predict the next bit given streaming bits as input. Although, it wasn't very reliable but it was good start. A book published in 1969 named Perceptron[1] by Marvin Minsky and Seymour Papert gave a complete blow to the research of Neural Network. As they argued that one perceptron cannot handle XOR problem, hence a combination of perceptron will also won't be able to handle it. After that in 1980s a couple of researchers worked out how to use back-propagation to train multi layer neural nets but they didn't have enough computation resources to train all the variables and they didn't have enough data. So, that led to waning to neural network for another decade. However, in recent years our applications of neural nets have advanced mainly because of the following reasons.

First, Prior to 2006, it was not possible to go beyond 2-3 layers in the neural network. As the neural network didn't seem to use intermediate layer well and it would get stuck in local minima. This problem was overcome by using layer-wise greedy unsupervised learning. The way they handled it are first, pre-training one layer at a time in a greedy way; second, using unsupervised learning at each layer in order to preserve information from the input; and finally, fine-tuning the whole network with respect to the ultimate criterion of interest. The reason for the success of layer wise training strategy was that unsupervised pre-training helps to mitigate the difficult optimization problem of deep networks by better initializing the weights of all layers.

Second, increase in availability of labeled data[2]. Earlier, researchers didn't have enough labeled data to run back-propagation algorithm on top of them since it used the correct labels to adjust weights. With availability of open and terabytes of data set it is now possible to adjust gradient so as to make it very precise. This also helps to compare various algorithms by running them on common dataset.

Third, apart from creating a new data set we now have tools to perturb existing data to create new ones. Data augmentation is popular way to do so such that we can asymmetrically cropped by a few percent to create new examples with the same label as the original.

Fourth, increase in computation power also enhanced usage of neural nets. With the advent of distributed and GPU systems, Neural networks can be trained faster and the same algorithms that didn't work in 1986, worked in 2006 because of availability of infrastructure. Using specialized hardware such as GPU combined with highly-optimized implementation of 2D convolution helped solve computer vision problems easily.

Fifth, advent of new libraries that help to implement neural networks easily. This is another important factor that lead to increase popularity of neural network. Various libraries such as Caffee, Tensorflow, Keras, Theano helped researchers to easily implement and experiment with back-propagation algorithm. They abstracted complex functionalities to implement pooling layer, subsampling etc.

Sixth, apart from the infrastructural advancements further experiments with neural network resulted in popularity of neural nets. For ex: back-propagation was applied in 1990s on handwritten zip code[5] to understand digits. This laid foundation for neural networks using convolutional neural nets or ConvNets. In general convnets make use of convolution, pooling, fully connected layer, inception to solve a particular problem. Varying depth and breadth can control their capacity. One of the earliest such net was AlexNet. That made use of 8 layers and used local response normalization to fasten training. Followed by AlexNet, VGG[3] focused on making use of small filters and typically 3 fully connected layers. Succeeded by GoogleNet that introduced the idea of longer but narrow nets, got away with fully connected layers completely and used inception modeling. Also, GoogleNet uses average output from three different layer rather than just output given by final layer. However, it was observed that stacking new layers wasn't linearly improved performance. In fact, it proved with the network depth increasing, accuracy gets saturated (which might be unsurprising) and then degrades rapidly. ResNet[7] introduced a completely new concept of carrying forward the result of one layer to next to next layer.

Because of the above points, neural network are successfully able to detect objects in real time using algorithms such as Faster R-CNN. To generate subtitles by using images or describe scene presented in an image.

Object classification and detection are two core problems in computer vision and pattern recognition. Object classification and detection share some common components and face many common challenges e.g., variability in illumination, rotation and scales, as well as deformation, clutter, occlusion, multi-stability and large intra-class variations[9]. In other words we can also say detection is classifying a part of an image into a category for ex: car, face etc. We can turn an object classifier into an object detector by sliding a small window across the image. At each step you run the classifier to get a prediction of what sort of object is inside the current window. Using a sliding window gives several hundred or thousand predictions for that image, but you only keep the ones the classifier is the most certain about. Mentioned below is the research that can optimize this process.

The earliest work done to make use of a classifier to perform detection using a single framework and shared features[9]. In general, first step in either is to extract features i.e. representing image patches is implemented via statistical analysis over pixels of image patches for ex SIFT or HOG[10]. To generate object bounding box predictions, Overfeat[9] simultaneously ran the classifier and regressor networks across all locations and scales. Since these share the same feature extraction layers, only the final regression layers needed to be recomputed after computing the classification network. Later, R-CNN was proposed as an improvement over Overfeat. Overfeat[9] used sliding window followed by regression to localize an object. However, R-CNN used unsupervised pre-training, followed by supervised domain specific fine-tuning[11]. R-CNN uses external region proposals and uses them to localize an object. Although, R-CNN is agnostic to region proposal method used but they used selective search. Using these proposals R-CNN runs the images in the bounding boxes through a pre-trained AlexNet and finally an SVM to see what object the image in the box is. Once, it classifies the object, it then runs the box through a linear regression model to output tighter coordinates for the box once the object has been classified[12]. However, R-CNN required forward pass for every single proposal generated. Next, Spatial Pyramid Pooling(SPPNet) was introduced. SPPNet suggested that cropped region may not contain entire object as done by R-CNN and pre-define scales shouldn't be a bottleneck. For this, they perform some information "aggregation" between convolutional layers and fully-connected layers. Hence, they run the convolutional layers only once on the entire image (regardless of the number of windows), and then extract features by SPP-net on the feature maps. For detection, they extract window-wise features from regions of the feature maps, while R-CNN extracts directly from image regions. Now, that researchers had been using region proposals, authors of Fast R-CNN suggested that images invariably have overlapping. Hence they suggested a single-stage training rather than using a multi-task loss unlike its predecessors. For detection in R-CNN, they consider each Region of Interest r, the forward pass outputs a class posterior probability distribution p and a set of predicted bounding-box offsets relative to r (each of the K classes gets its own refined bounding-box prediction). They assign a detection confidence to r for each object class k. Then they perform non-maximum suppression independently of each class using the algorithm and settings from R-CNN[13]. Next, idea of Faster R-CNN was based on the fact that region proposals depended on features of the image that were already calculated with the forward pass of the CNN (first step of classification). So, why not reuse those same CNN results for region proposals instead of running a separate selective search algorithm. Faster R-CNN adds a fully Convolutional Network on top of the features of the CNN creates what's known as the Region Proposal Network (RPN). RPN works by passing a sliding window over the CNN feature map and at each window, outputting k potential bounding boxes and scores for how good each of those boxes is expected to be. Now once we have bounding box we use Fast R-CNN methodology to classify and tighten the bounding box[12]. Extending the idea of Faster R-CNN is mask R-CNN that makes use of a mask to figure out if a pixel belongs to an object or not. Once the masks are generated, Mask R-CNN combines them with the classifications and bounding boxes from Faster R-CNN to generate[12]. Further real time object detection solutions have been suggested. YOLO, devides image into cell. Each cell is responsible for predicting bounding boxes and a certainty level. The confidence score for the bounding box and the class prediction are combined into one final score that tells us the probability that this bounding box contains a specific type of object[14].

So, all these research papers somehow optimize on sliding window idea either by feeding probable regions or generating region on run time to convert an object classifier to an object detector[14].

References:

1.) https://en.wikipedia.org/wiki/Perceptrons_(book)
2.) ImageNet Classification with Deep Convolutional Neural Networks by A. Krizhevsky, I. Sutskever.
3.) Very Deep Convolutional networks for large scale Image recognition by K. Simonyan and A. Zisserman
4.) Neural Networks and Neuroscience-Inspired Computer Vision by D Daniel Cox and T. Dean
5.) Backpropagation applied to Hanwritten zip code by Y. LeCun, B. Boser and J.S. Denker
6.) https://www.quora.com/Why-has-it-taken-so-long-for-the-use-of-neural-networks-and-machine-learning-to-grow
7.) Deep Residual Learning for Image Recognition by K. He, X. Zhang, S. Ren and J. Sun
8.) https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history1.html
9.) OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks by P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun
10.) Recent Progress on Object Classification and Detection by T. Tan, Y. Huang, and J. Zhang
11.) Rich feature hierarchies for accurate object detection and semantic segmentation by R. Girshick, J. Donahue, T. Darrell and J. Malik
12.) https://blog.athelas.com/a-brief-history-of-cnns-in-image-segmentation-from-r-cnn-to-mask-r-cnn-34ea83205de4
13.) Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition by K. He, X. Zhang, S. Ren, and J. Sun
14.) http://machinethink.net/blog/object-detection-with-yolo/