

Object Detection

Learning Objectives:

- Identify the components used for object detection (landmark, anchor, bounding box, grid, ...) and their purpose
- Implement object detection
- Implement non-max suppression to increase accuracy
- Implement intersection over union
- Handle bounding boxes, a type of image annotation popular in deep learning
- Apply sparse categorical crossentropy for pixelwise prediction
- Implement semantic image segmentation on the CARLA self-driving car dataset
- Explain the difference between a regular CNN and a U-net



FPT UNIVERSITY

Object Detection



FPT UNIVERSITY

- 1 Object localization Object localization
- 2 Landmark detection
- 3 Object detection
- 4 Convolutional implementation of sliding windows
- 5 Bounding box predictions
- 6 Intersection over union
- 7 Non-max suppression
- 8 Anchor boxes
- 9 Region proposals (Optional)
- 10 Putting it together: YOLO algorithm
- 11 Semantic segmentation with U-Net



FPT UNIVERSITY

Object Detection

Object localization

Object localization

- Object detection, a subfield of computer vision, involves identifying and localizing objects in an image.
 - The process begins with object localization, where the algorithm not only identifies but also draws bounding boxes around objects.
- Unlike image classification, which identifies a single object, object detection handles multiple objects of different categories in a single image.
- Modification for classification with localization requires the neural network to output four additional parameters for bounding box localization.
- The target label includes object class and bounding box parameters. Training utilizes a squared error loss function.
- This approach of outputting real numbers for localization extends to various areas in computer vision.

What are localization and detection?

Image classification



1 object

Classification with
localization



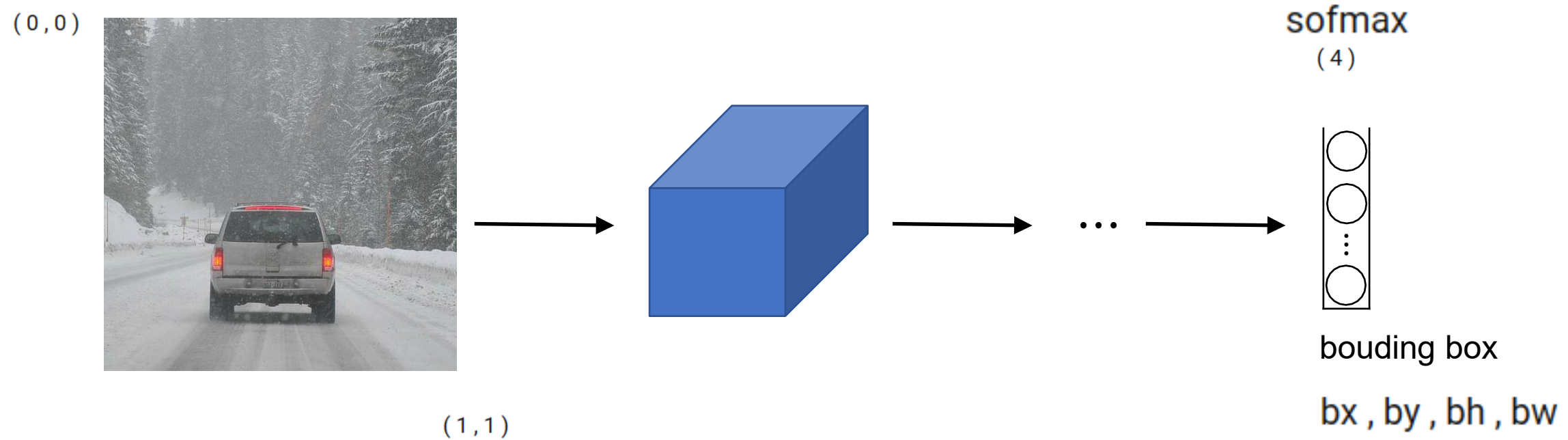
1 object

Detection



multiple object

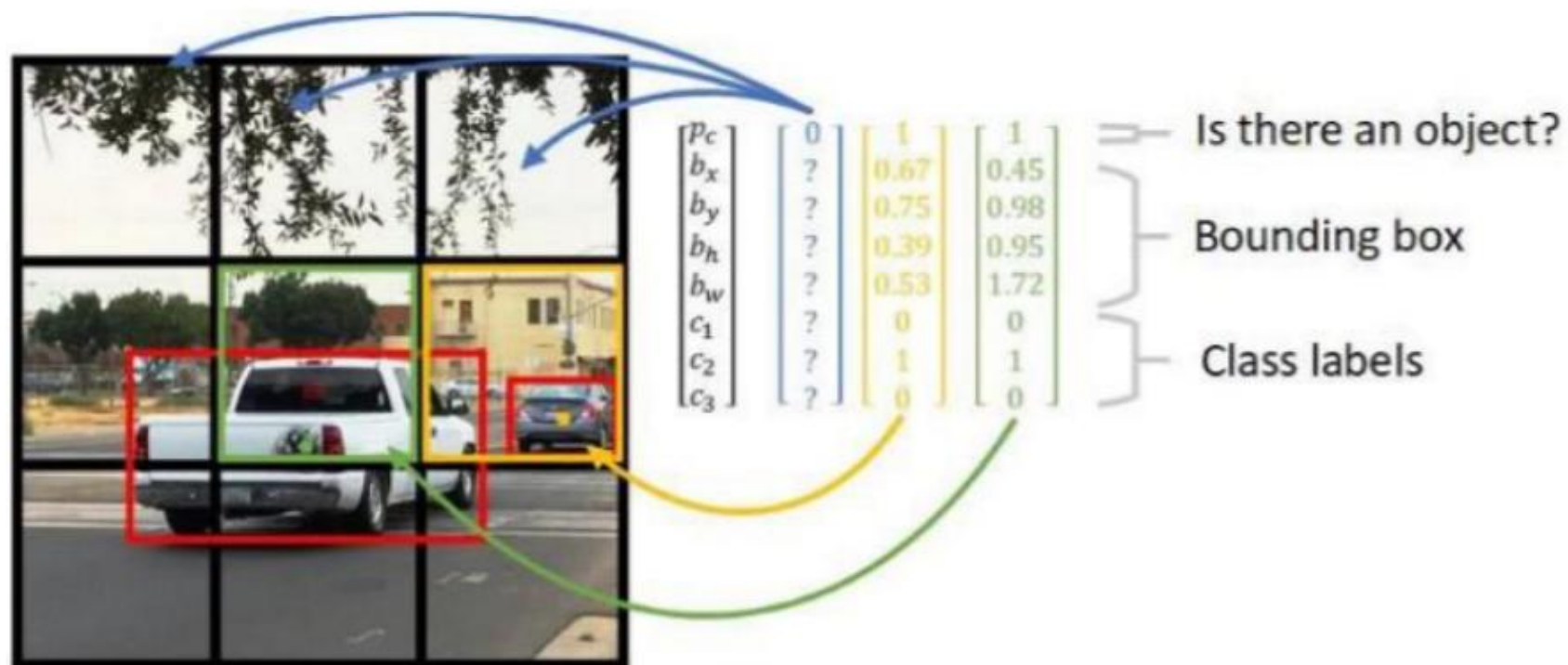
Classification with localization



- 1 - pedestrian
- 2 - car
- 3 - motorcycle
- 4 - background

Defining the target label y

- 1 - pedestrian
- 2 - car
- 3 - motorcycle
- 4 - background





FPT UNIVERSITY

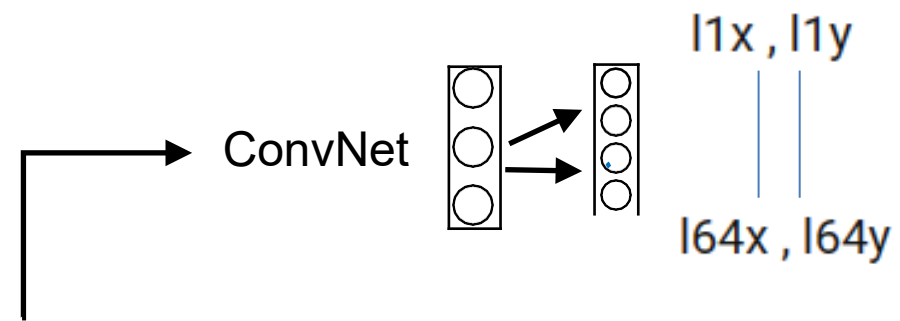
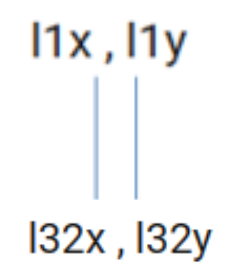
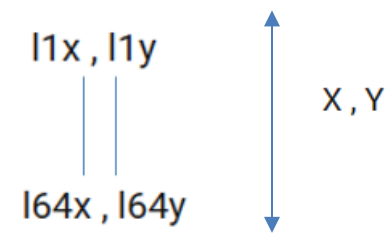
Object Detection

Landmark detection

Landmark detection

- Landmark detection in computer vision identifies key points on an object or image, such as facial features for expressions.
- A neural network is trained to output X and Y coordinates for each landmark, with varying numbers depending on the application.
- Training requires a labeled set with consistent coordinates for each point across images.
- Once trained, the network detects landmarks in new images, applied in face recognition, pose detection, and augmented reality effects.
- Additionally, landmark detection enhances object detection accuracy by providing shape and position information.

Landmark detection





FPT UNIVERSITY

Object Detection

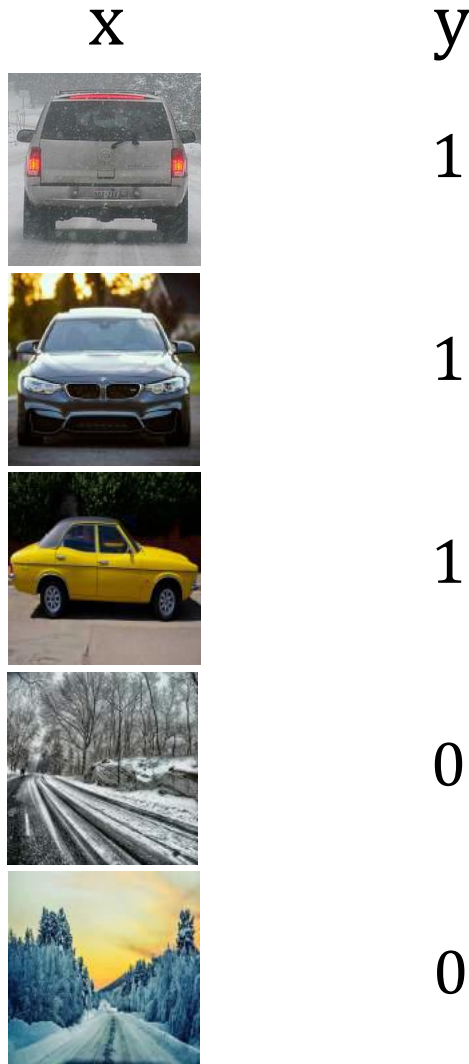
Object detection

Object detection

- The Sliding Windows Detection Algorithm utilizes a ConvNet for object detection. It involves creating a labeled training set, training the ConvNet to detect the object's presence, and using sliding windows to pass rectangular regions of an input image into the ConvNet for classification.
- The algorithm divides the test image into fixed-sized rectangular regions, shifting them with a fixed stride to cover the entire image. Windows of increasing sizes detect objects of different scales.
- However, the algorithm's computational cost is high due to processing numerous regions, particularly challenging with a ConvNet.

Car detection example

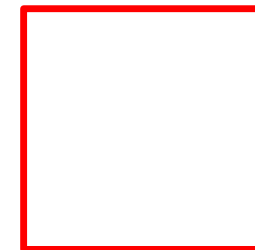
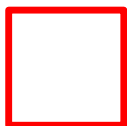
Training set:



Convnet

Y

Sliding windows detection





FPT UNIVERSITY

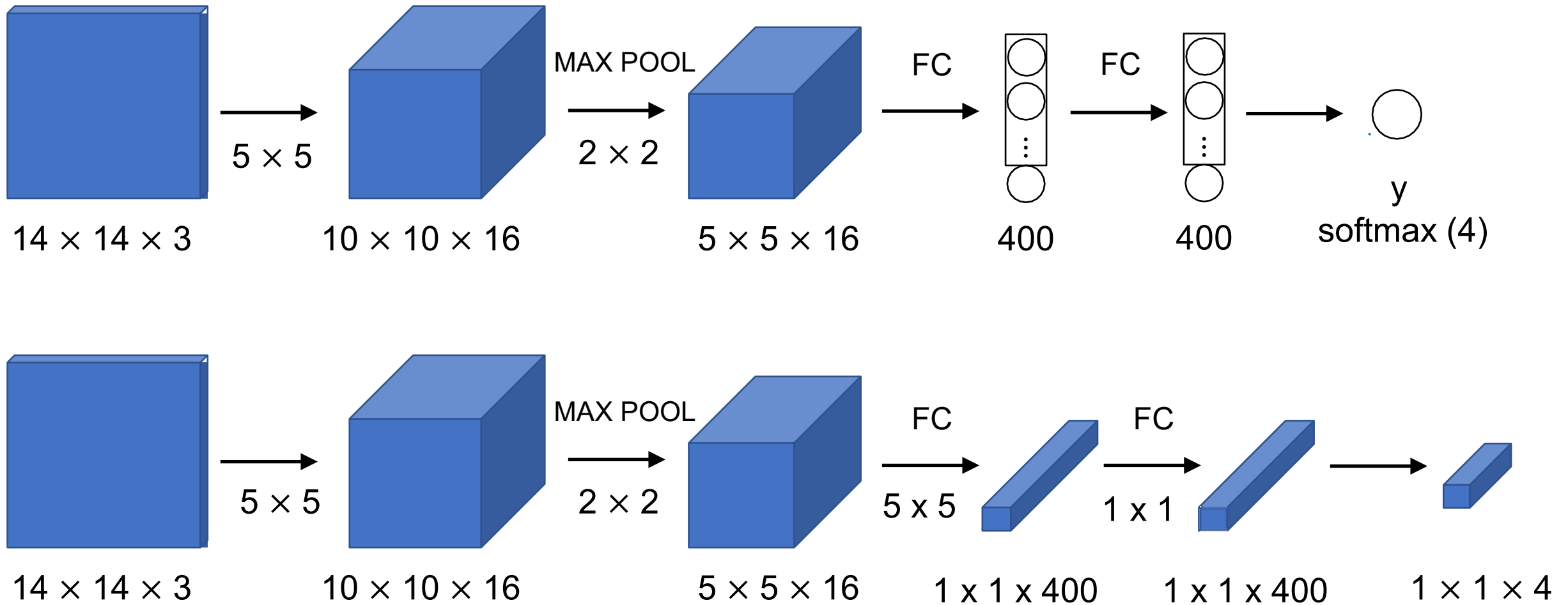
Object Detection

Convolutional implementation of
sliding windows

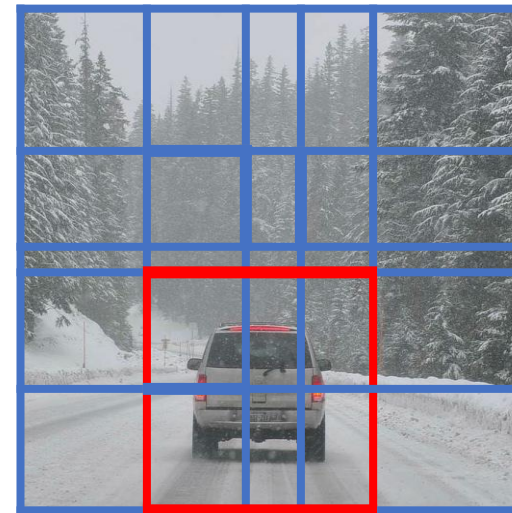
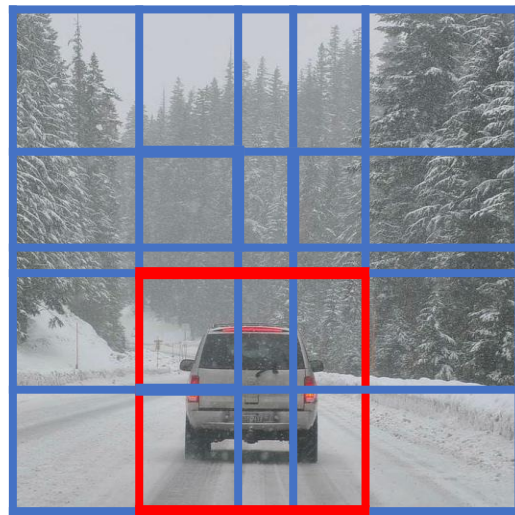
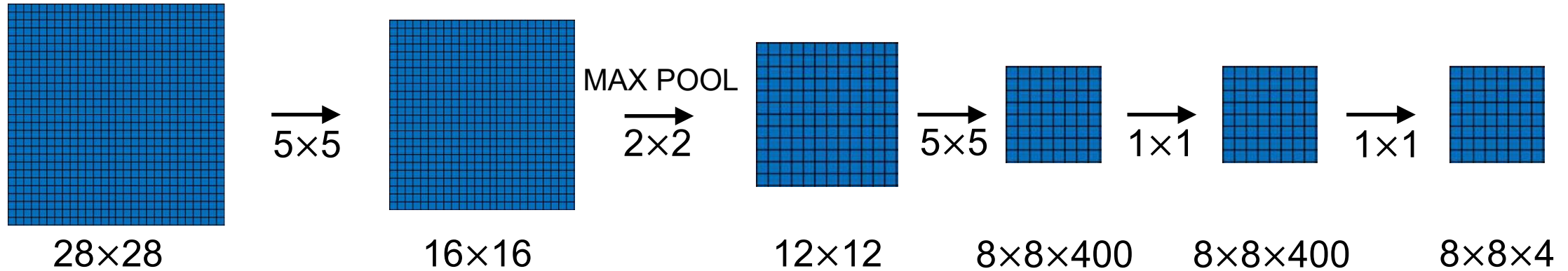
Convolutional implementation of sliding windows

- The sliding windows object detection algorithm involves cropping small image regions and feeding them to a ConvNet for predictions. However, the computational cost, especially with fine granularity, is high.
- To address this, fully connected layers can be converted to 5x5 filters, allowing the use of convolutional layers for improved efficiency.
- The convolutional implementation shares computation across the entire image, passing it through the ConvNet with 5x5 filters, followed by max pooling and fully connected layers. This results in a 1x1x4 volume, and using 5x5 filters yields a 2x2x4 volume for each region.
- While more efficient, this algorithm may lack precision in bounding box positioning, a concern addressed in the next section.

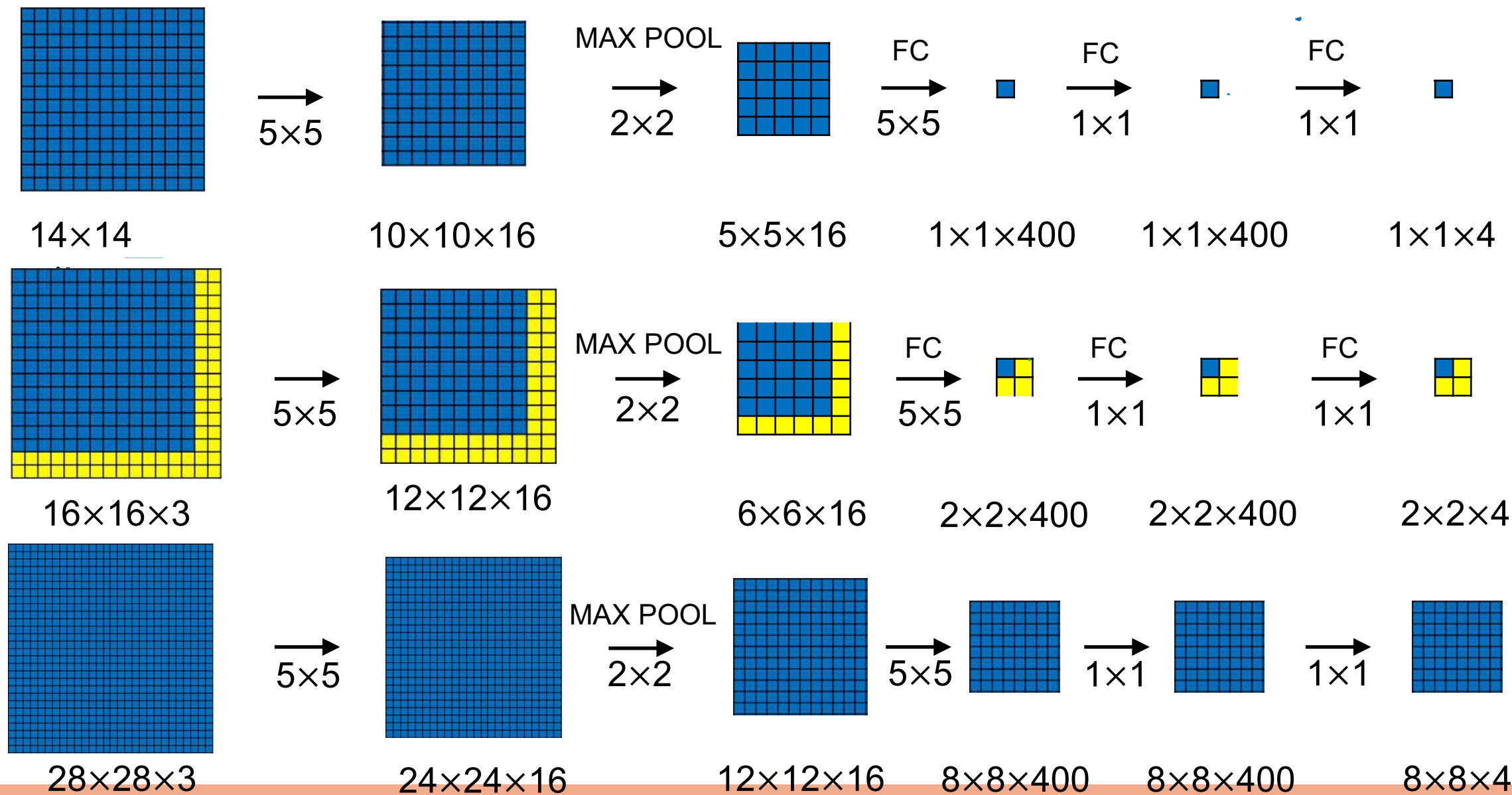
Turning FC layer into convolutional layers



Convolution implementation of sliding windows



Convolution implementation of sliding windows





FPT UNIVERSITY

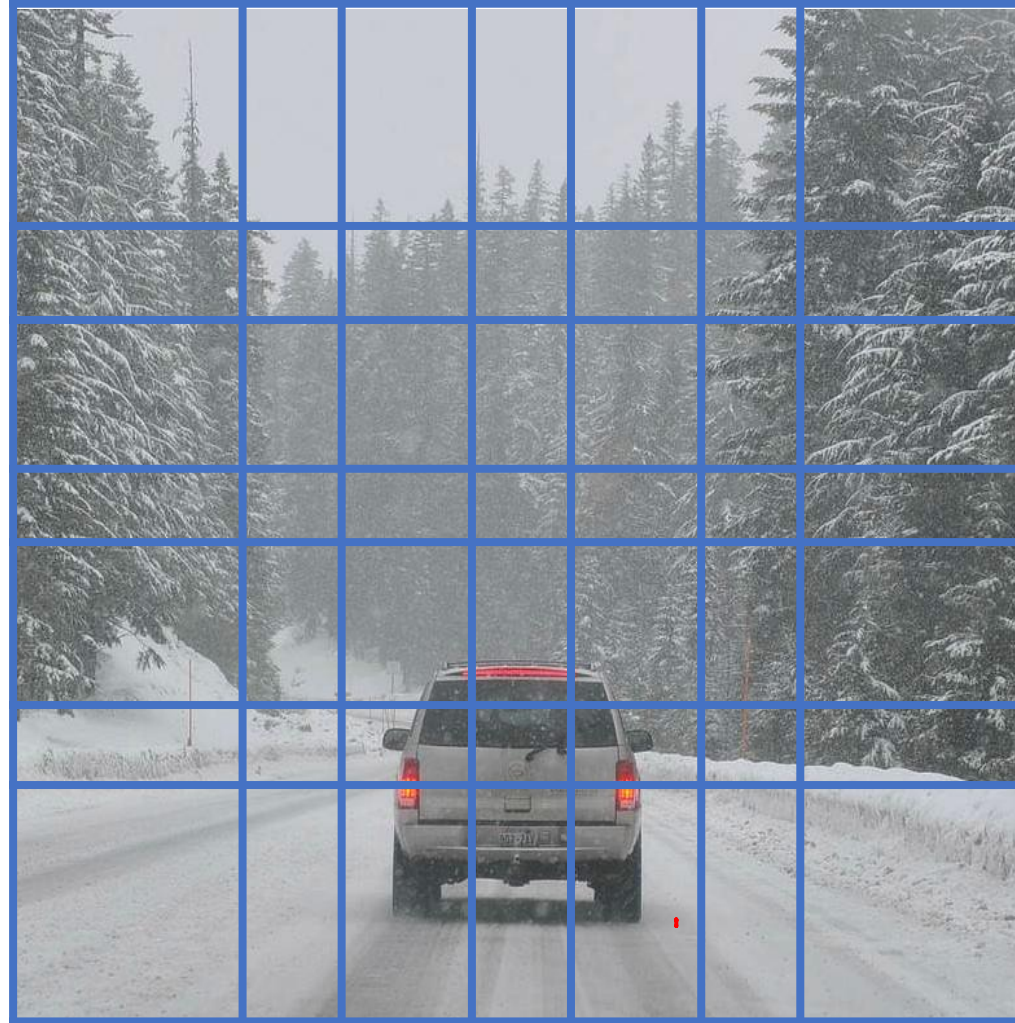
Object Detection

Bounding box predictions

Bounding box predictions

- The YOLO (You Only Look Once) algorithm enhances bounding box prediction accuracy by dividing the image into a grid and applying classification and localization to each cell.
- Each cell is assigned an 8-dimensional vector, including object presence probability, bounding box coordinates (BX, BY, BH, BW), and class probabilities (C1, C2, C3, etc.).
- Training involves passing the input image through convolutional and max pool layers to produce a 3 by 3 by 8 output volume, aligning with target labels. YOLO encodes bounding box coordinates using a convention relative to the grid cell. Its efficiency lies in convolutional implementation, allowing shared computation for real-time object detection.
- While the YOLO paper can be complex, improvements beyond its presented parameterization are explored in the following sections.

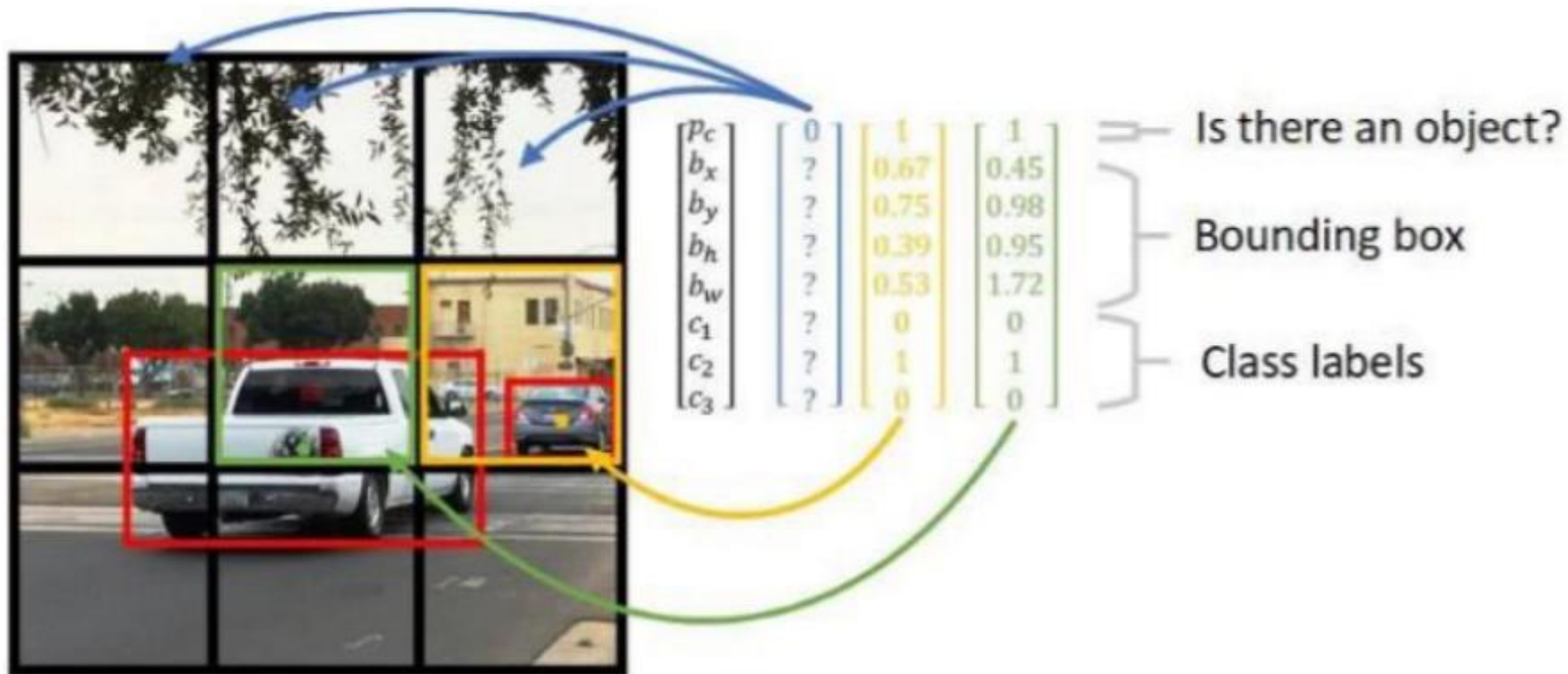
Output accurate bounding boxes



YOLO algorithm

Labels for training

For each grid cell:





FPT UNIVERSITY

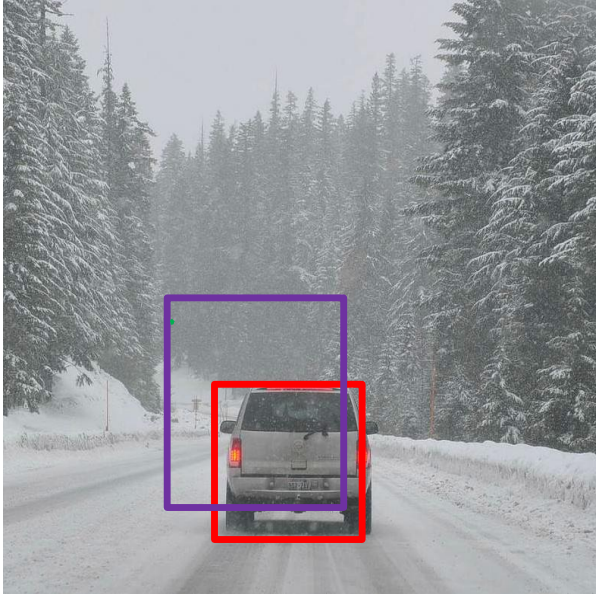
Object Detection


Intersection over union

Intersection over union

- Intersection over Union (IoU) assesses object detection accuracy by evaluating the overlap between predicted and ground-truth bounding boxes.
- The IoU is computed as the intersection area divided by the union area, producing a value from 0 to 1, where 1 indicates perfect overlap.
- A threshold, commonly 0.5, determines the correctness of the predicted bounding box.
- Adjusting the threshold allows for more stringent evaluation.
- IoU is versatile, serving not only as an evaluation metric for object detection algorithms but also as a tool to measure bounding box similarity, as discussed further in the next section on non-max suppression.

Evaluating object localization



$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


“Correct” if $\text{IoU} \geq 0.5$

More generally, IoU is a measure of the overlap between two bounding boxes.



FPT UNIVERSITY

Object Detection

Non-max suppression

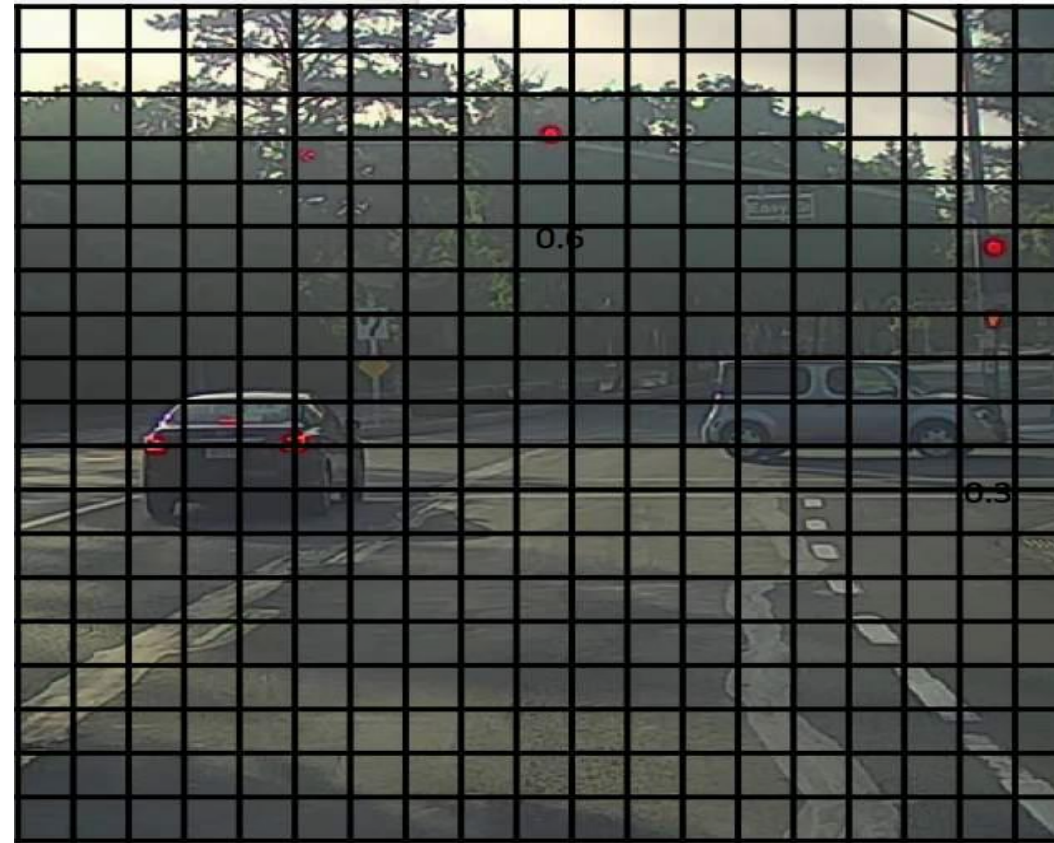
Non-max suppression

- Non-max suppression ensures object detection algorithms identify each object only once.
 - 1. Assigning a probability (P_c) to each predicted bounding box, boxes below a threshold are discarded.
 - 2. Boxes with the highest P_c values are highlighted, and overlapping boxes are suppressed based on Intersection over Union (IoU).
 - 3. This process repeats until all boxes are either predicted or suppressed.
- Applied independently to each class's output, non-max suppression enhances accuracy by eliminating duplicates and reducing false positives in object detection algorithms.

Non-max suppression example

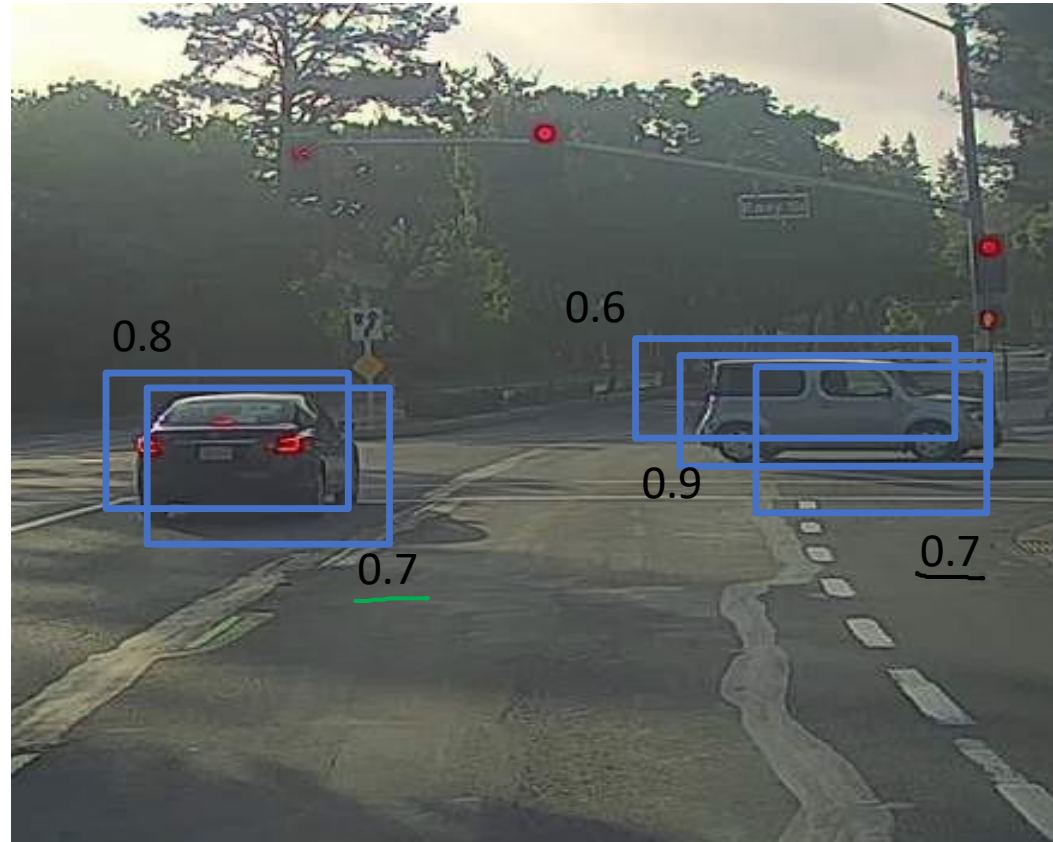


Non-max suppression example

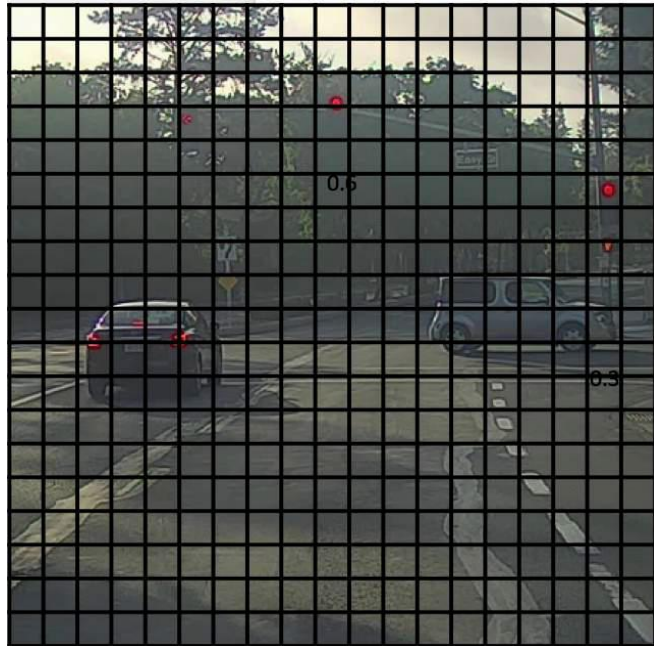


19 x 19

Non-max suppression example



Non-max suppression algorithm



19×19

Each output prediction is:

$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \end{bmatrix}$$

Discard all boxes with $p_c \leq 0.6$

While there are any remaining boxes:

- Pick the box with the largest p_c
Output that as a prediction.
- Discard any remaining box with $\text{IoU} \geq 0.5$ with the box output in the previous step



FPT UNIVERSITY

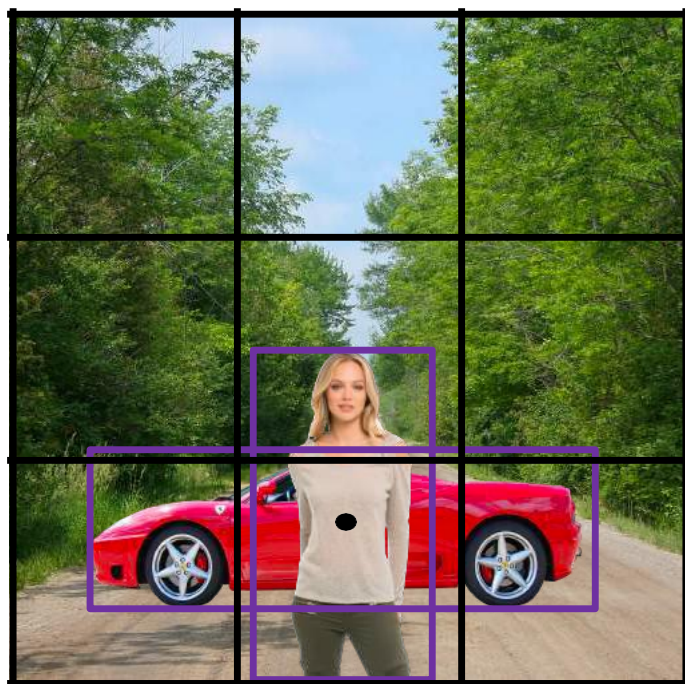
Object Detection

Anchor boxes

Anchor boxes

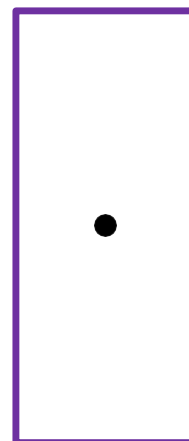
- Anchor boxes in object detection enable each grid cell to detect multiple objects, overcoming the limitation of one object per cell.
- By pre-defining anchor box shapes and associating predictions based on the highest Intersection over Union (IoU) with the object's shape, the YOLO algorithm encodes objects in a 16-dimensional vector.
- This method allows specialization for various object types, such as tall or wide objects.
- Anchor boxes can be selected manually or through a K-means algorithm, providing versatility in object shape detection.

Overlapping objects:

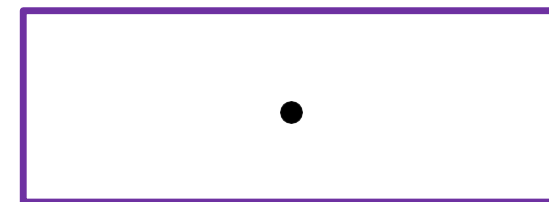


$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \end{bmatrix}$$

Anchor box 1:



Anchor box 2:



$$y =$$

$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \end{bmatrix}$$

Anchor box 1:

$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \end{bmatrix}$$

Anchor box 2:

Anchor box algorithm

- Previously:
 - Each object in training image is assigned to grid cell that contains that object's midpoint.
- With two anchor boxes:
 - Each object in training image is assigned to grid cell that contains object's midpoint and anchor box for the grid cell with highest IoU.



FPT UNIVERSITY

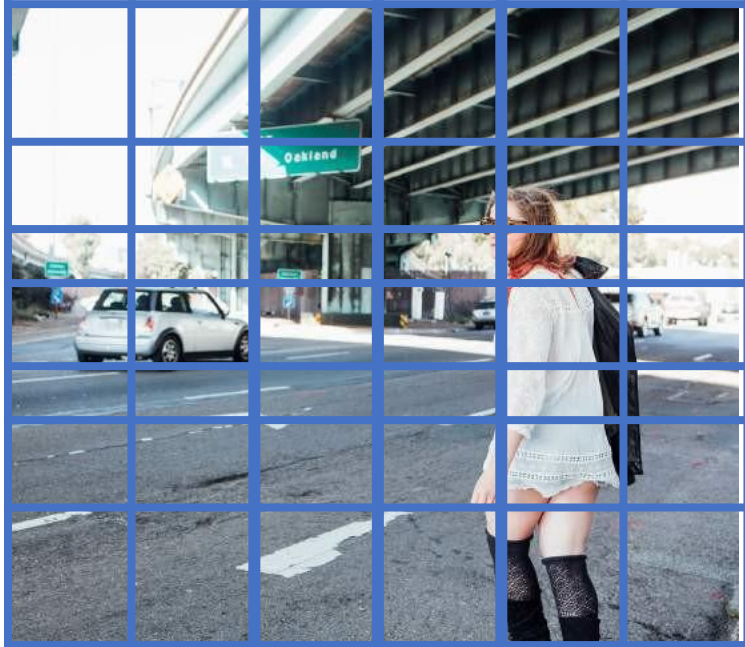
Object Detection

Region proposals (Optional)

Region proposals (Optional)

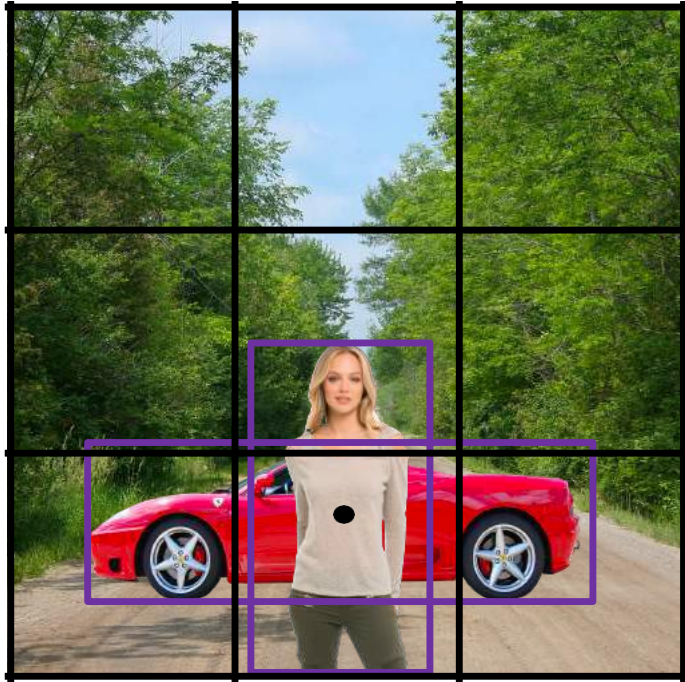
- In object detection, region proposals, like those in R-CNN (Regions with Convolutional Neural Networks), select specific windows for algorithm processing instead of evaluating each window individually.
- While effective, R-CNN is slow, leading to improvements like Fast R-CNN and Faster R-CNN, employing convolutional methods. An alternative is the YOLO (You Only Look Once) algorithm, processing the entire image at once for direct object detections.
- Understanding region proposals is key, but ongoing research aims to develop more efficient object detection algorithms without relying on region proposals.

Region proposal: R-CNN



segmentation algorithm

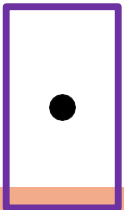
Anchor box example



$y =$

$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 1 \\ 0 \\ 0 \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Anchor box 1: Anchor box 2:





Object Detection

FPT UNIVERSITY Putting it together: YOLO algorithm

Putting it together: YOLO algorithm

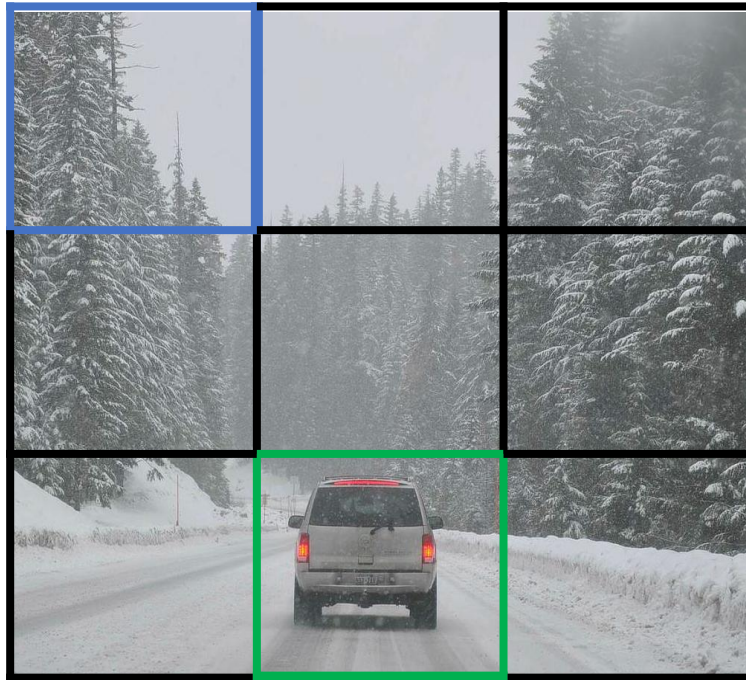
- The YOLO (You Only Look Once) object detection algorithm is a popular and effective algorithm that encompasses many of the best ideas in computer vision related to object detection.
- To train the YOLO algorithm, you need to construct a training set with explicit labels for the three classes you want to detect (pedestrians, cars, and motorcycles) and a background class. If you use two anchor boxes, the output y will be 3 by 3 by 2 by 8 or 3 by 3 by 16. For each of the nine grid cells in the training set, you form the appropriate target vector y , specifying whether there is an object there and its class, along with its bounding box. You train a convolutional neural network (ConvNet) on this training set, with an input image (e.g., 100 by 100 by 3) and an output volume (e.g., 3 by 3 by 16).

Putting it together: YOLO algorithm

- To make predictions, the neural network outputs a 3 by 3 by 2 by 8 volume, with each grid cell having a vector specifying whether there is an object there, its class, and its bounding box:
 - Run non-max suppression on the predicted bounding boxes to eliminate low-probability predictions and to eliminate overlapping predictions for the same object.
 - Run non-max suppression for each of the classes (pedestrians, cars, and motorcycles) to generate the final predictions.
- The YOLO algorithm is one of the most effective object detection algorithms, and you get to practice implementing many of its components in this week's problem exercise.

Training

- 1 - pedestrian
- 2 - car
- 3 - motorcycle

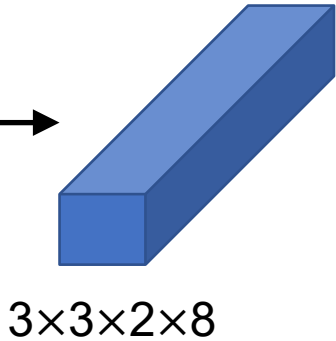
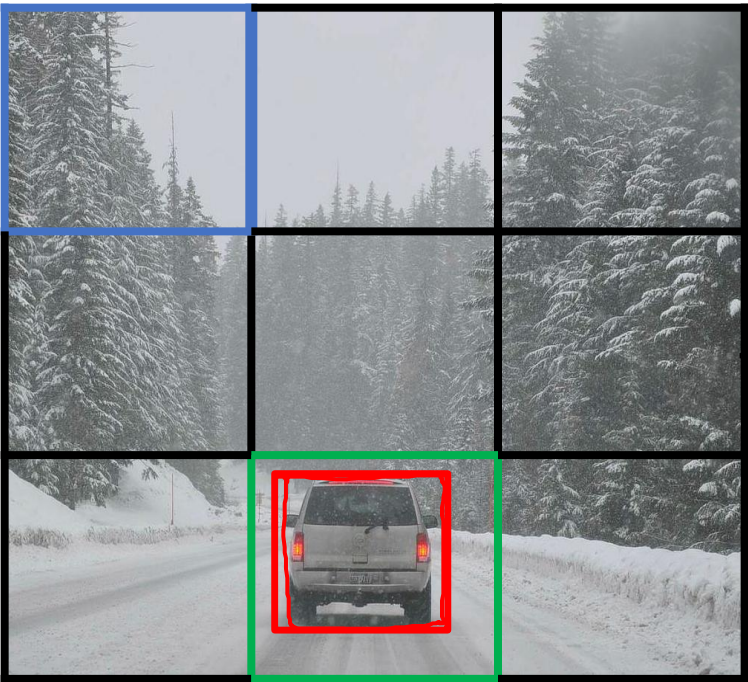


$y =$

y is $3 \times 3 \times 2 \times 8$

$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix} \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Making predictions



$y =$

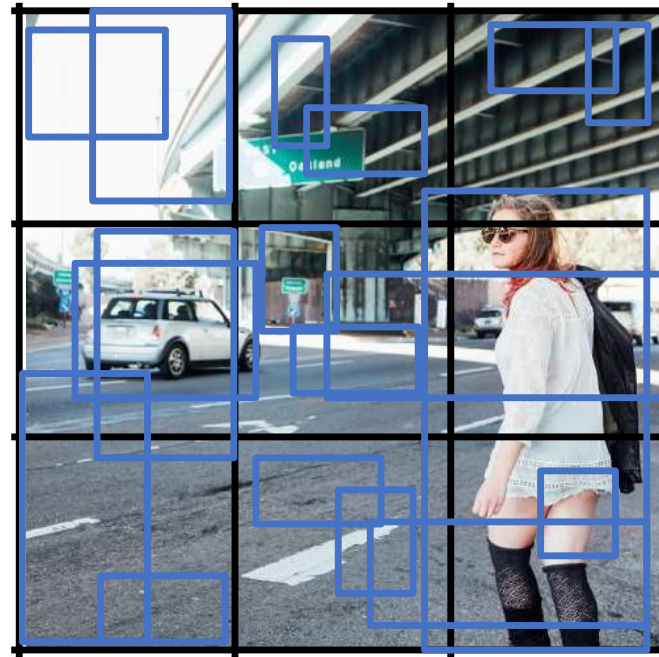
$$\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Outputting the non-max suppressed outputs

- For each grid cell, get 2 predicted bounding boxes.
- Get rid of low probability predictions.
- For each class (pedestrian, car, motorcycle) use non-max suppression to generate final predictions.



Faster algorithms

- R-CNN: Propose regions. Classify proposed regions one at a time. Output label + bounding box.
- Fast R-CNN: Propose regions. Use convolution implementation of sliding windows to classify all the proposed regions.
- Faster R-CNN: Use convolutional network to propose regions.



Convolutional Neural Networks

FPT UNIVERSITY

Semantic segmentation with U-Net

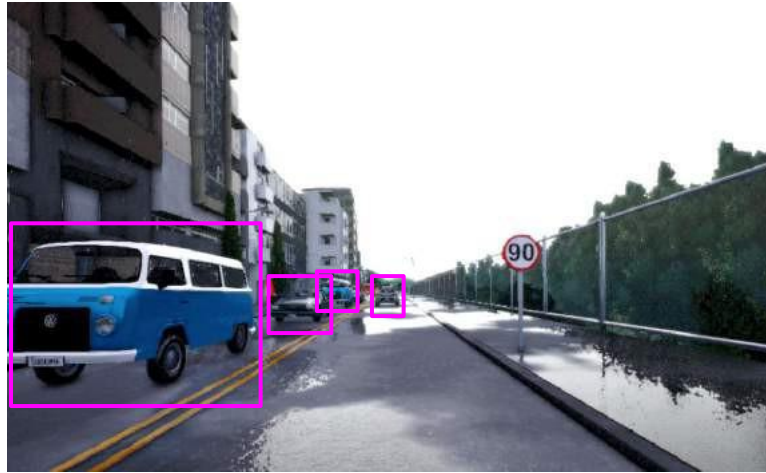
Semantic segmentation with U-Net

- Semantic segmentation, distinct from object detection, labels each pixel in an image to determine object presence. Applied in various domains like self-driving cars and medical imaging, it relies on algorithms outputting a matrix of class labels for every pixel.
- The unit architecture, a convolutional neural network, employs transpose convolutions to expand activations and label each pixel.
- Semantic segmentation is crucial in computer vision applications, including medical imaging, self-driving cars, and object recognition.

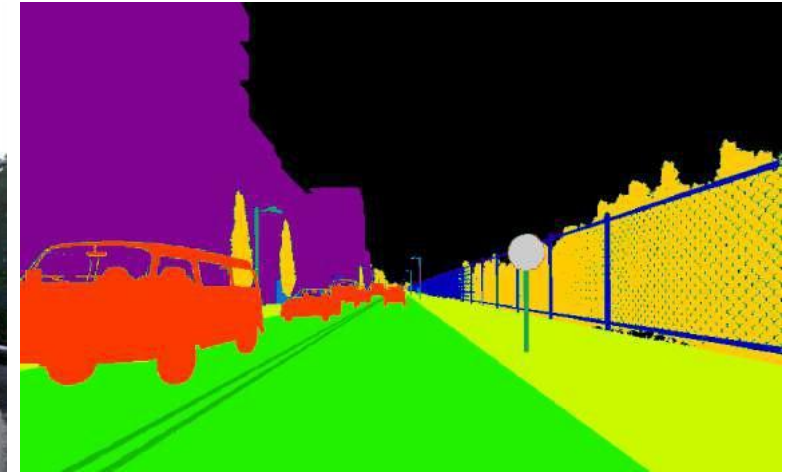
Object Detection vs. Semantic Segmentation



Input image

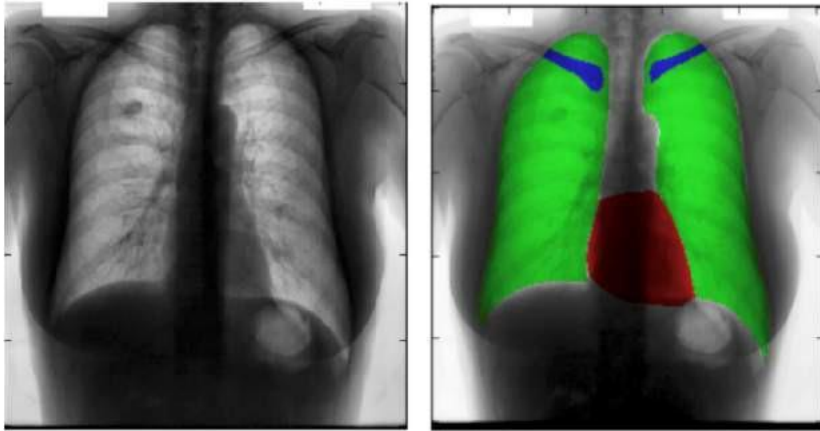


Object Detection

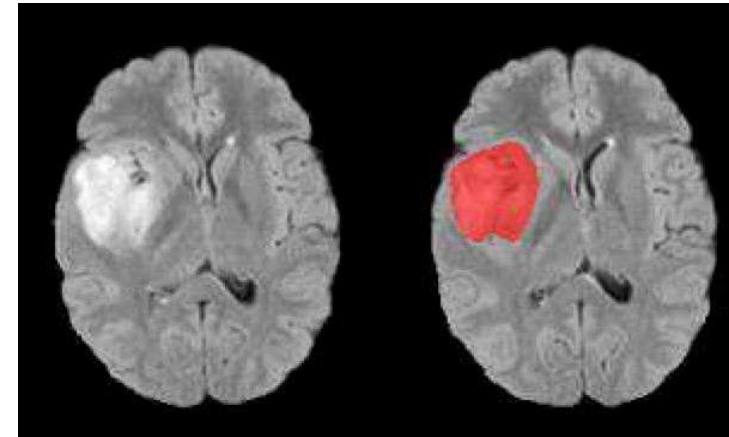


Semantic Segmentation

Motivation for U-Net

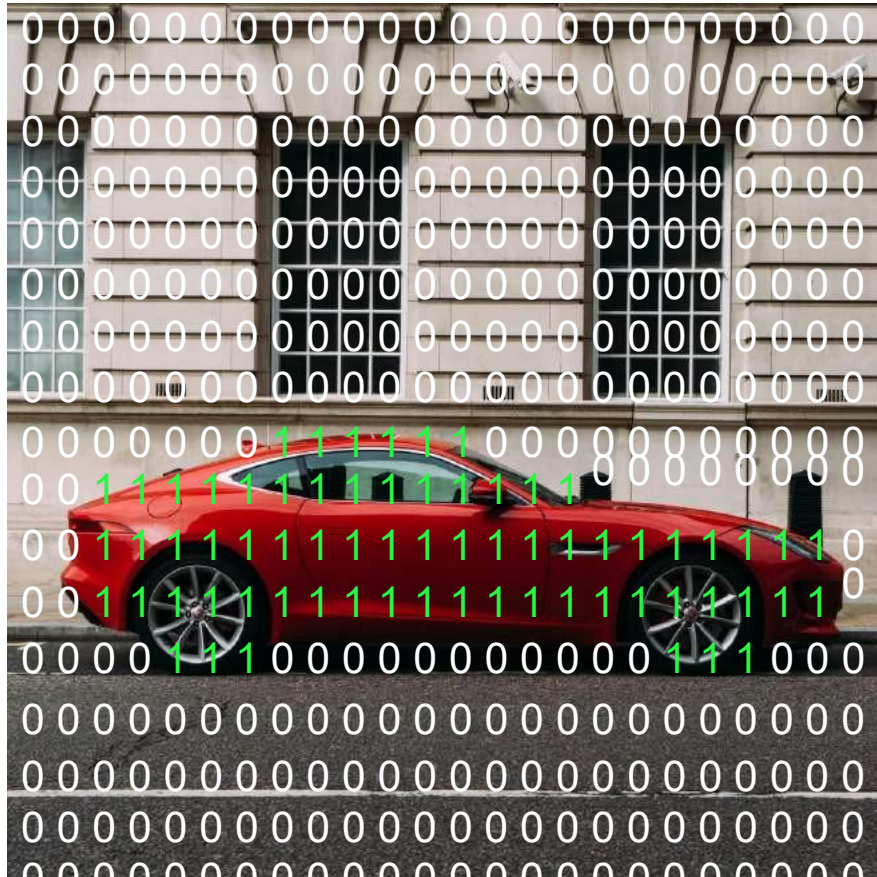


Chest X-Ray



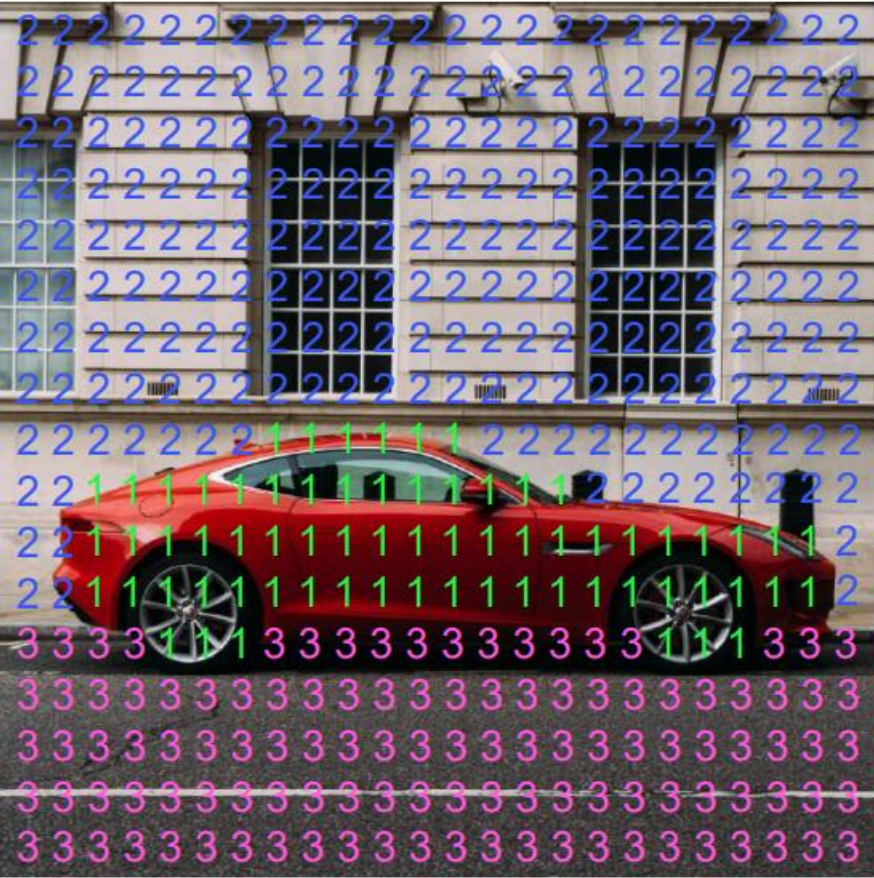
Brain MRI

Per-pixel class labels



1. Car
0. Not Car

Per-pixel class labels

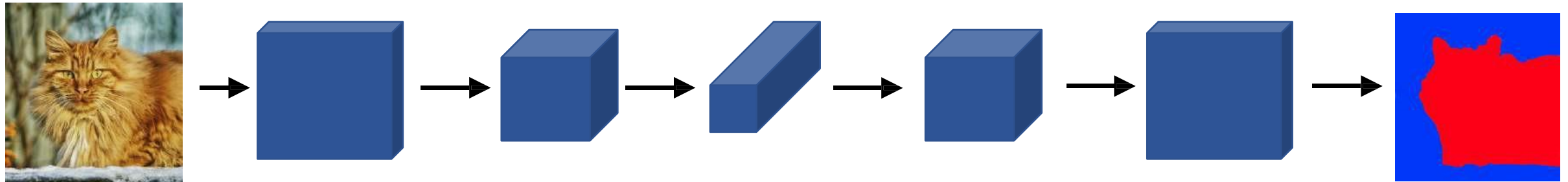


- 1. Car
- 2. Building
- 3. Road



Segmentation Map

Deep Learning for Semantic Segmentation





FPT UNIVERSITY

Convolutional Neural Networks

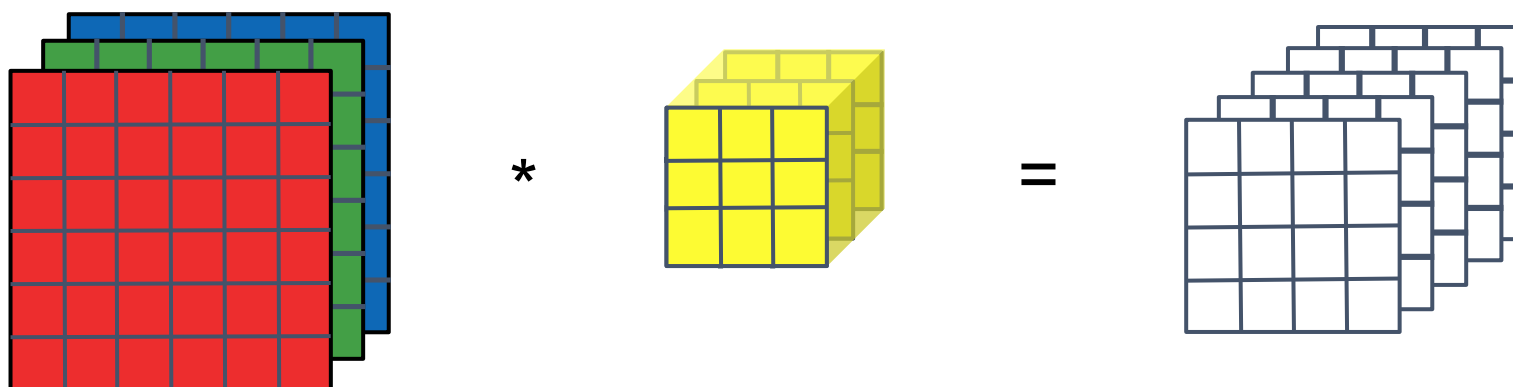
Transpose Convolution

Transpose Convolution

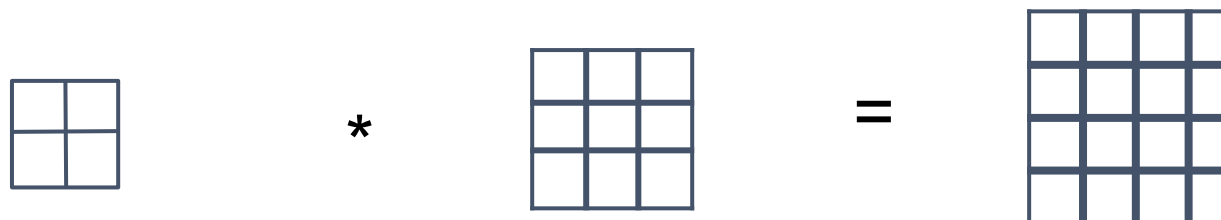
- Transpose convolution is a crucial element in the unit architecture for semantic segmentation, employed to enlarge a small input into a larger output.
- Unlike regular convolution where the filter is applied to the input, in transpose convolution, the filter is positioned on the output.
- For instance, using a 3×3 filter with padding $p=1$ and stride $s=2$ on a 2×2 input creates a 4×4 output. Each entry in the input matrix illustrates the filter application, resulting in a 4×4 output matrix.
- This operation is fundamental in the unit architecture for gradually expanding feature maps in semantic segmentation until the final segmentation map is achieved.

Transpose Convolution

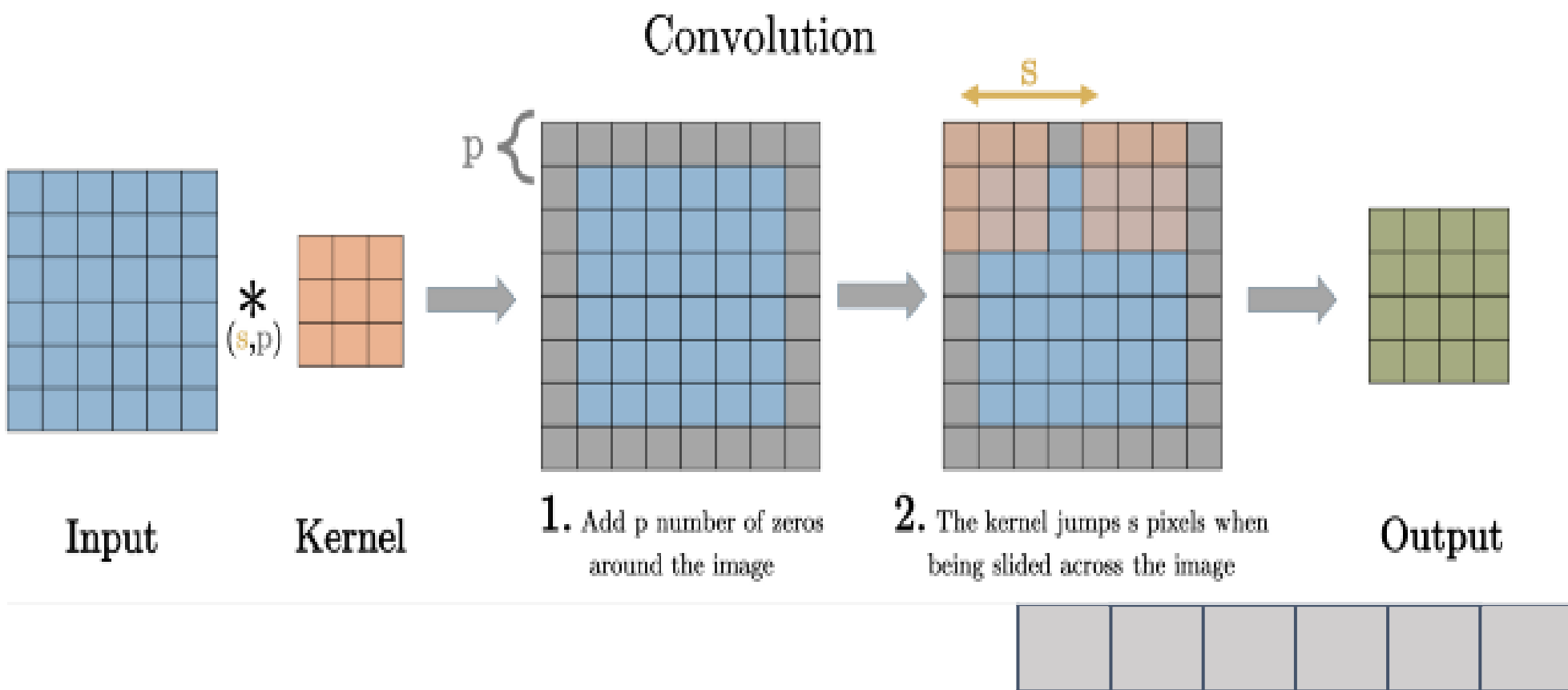
Normal Convolution



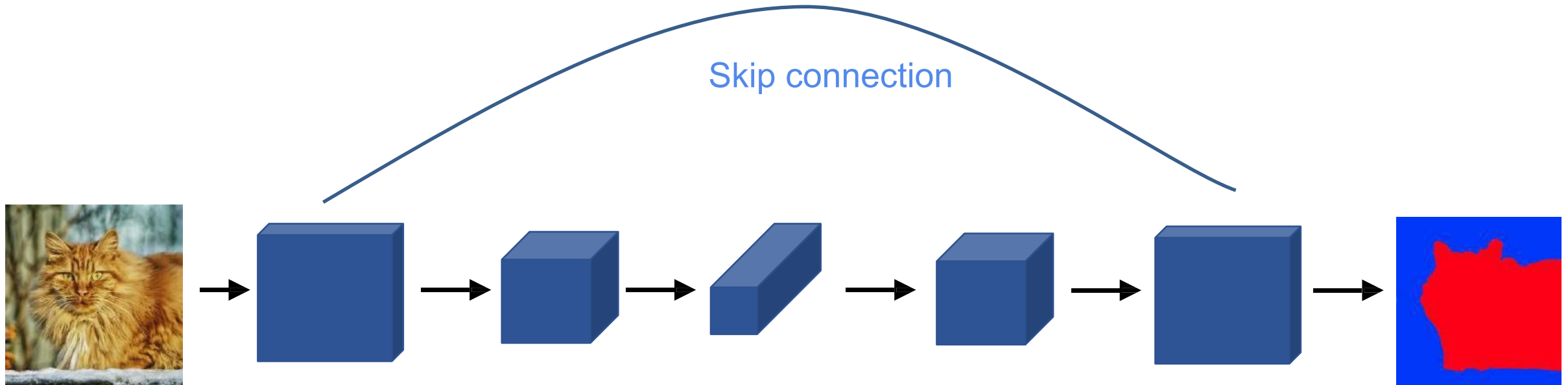
Transpose Convolution



Transpose Convolution



Deep Learning for Semantic Segmentation



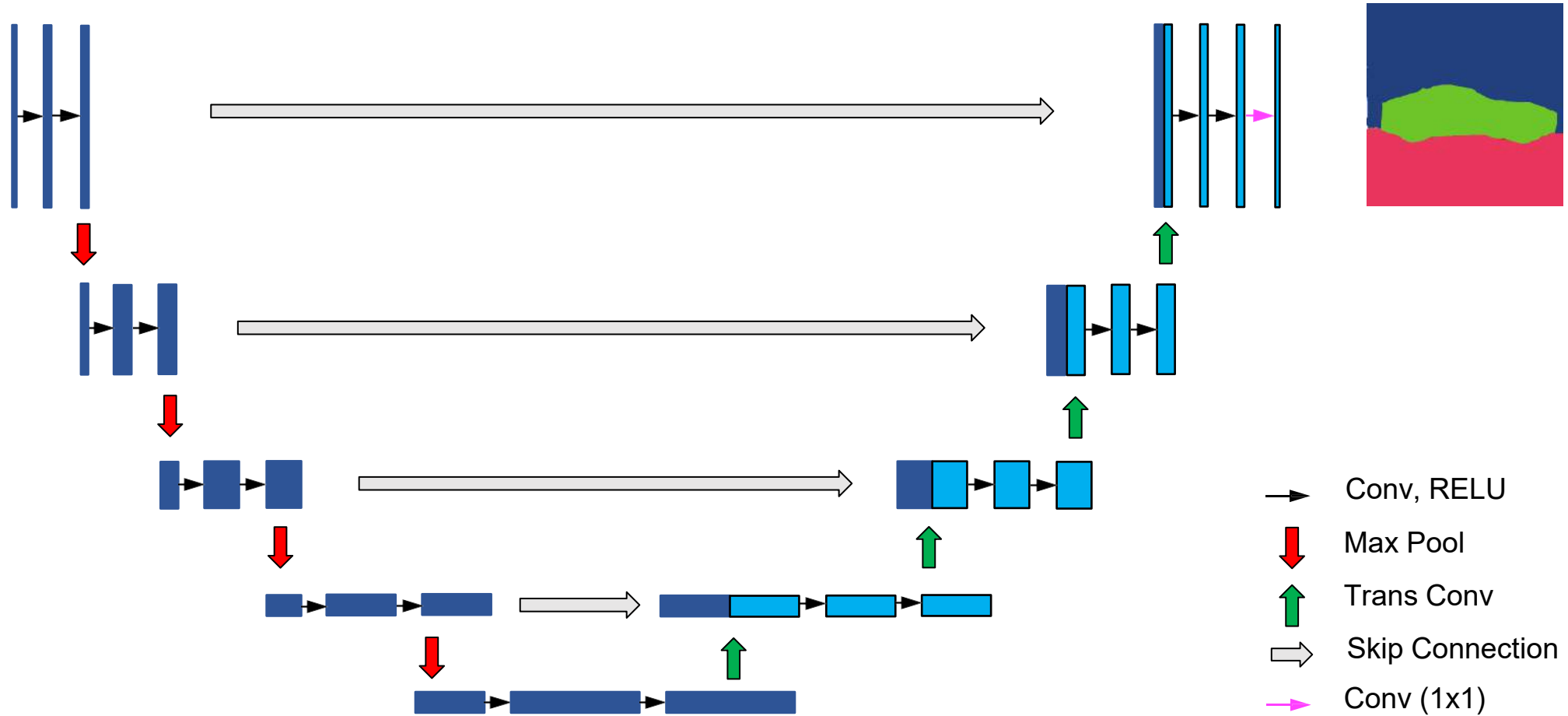
U-Net Architecture Intuition

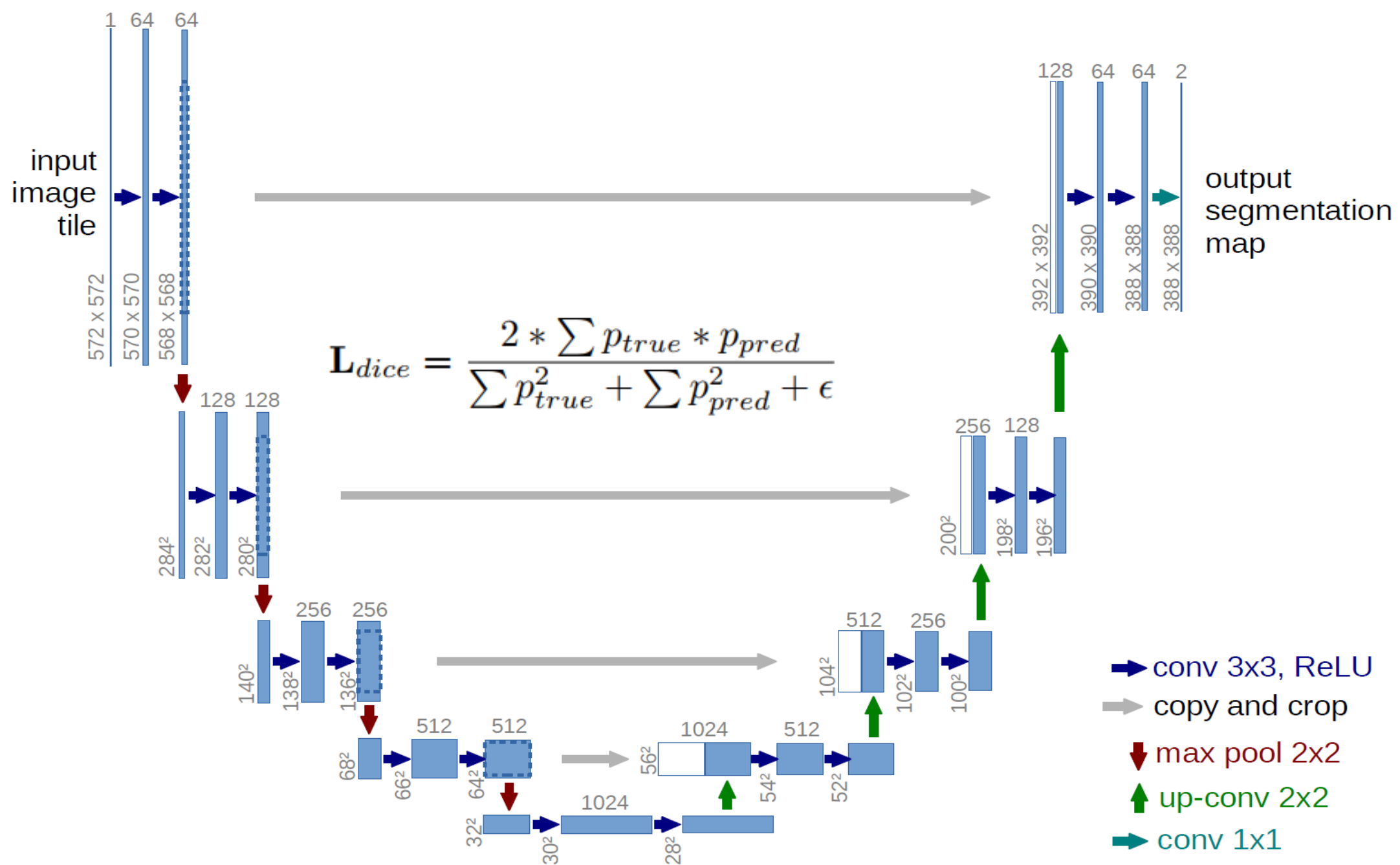
- The neural network for semantic segmentation involves initial convolutions for compression and spatial reduction, followed by transpose convolutions to restore the image size.
- Adding skip connections, as in the unit architecture, significantly enhances network performance.
- These connections provide the final layer with both high-level spatial/contextual information and low-level texture details for more accurate segmentation decisions.

U-Net Architecture

- U-Net, a convolutional neural network for semantic segmentation in computer vision, comprises an encoder and a decoder.
- The encoder has convolutional layers reducing spatial dimensionality and increasing channels, while the decoder has transposed convolutional layers for spatial expansion.
- Unique skip connections between the encoder and decoder enable the use of both high-level and low-level information in segmentation decisions.
- U-Net is widely used in biomedical image segmentation and other tasks like road segmentation and object detection, known for its distinctive U-shaped architecture.

U-Net

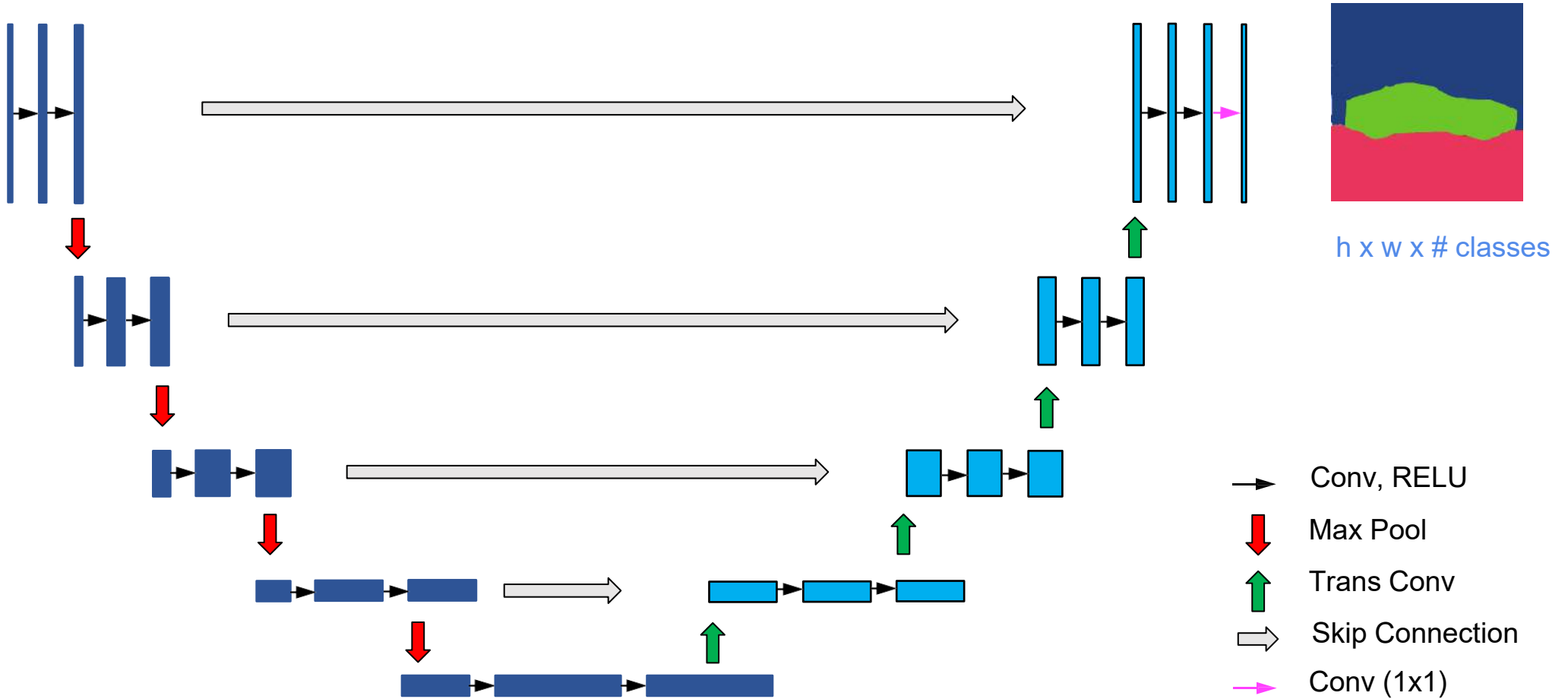




U-Net



$h \times w \times 3$



Summarization

- Object localization predicts a bounding box for an object, while landmark detection identifies specific points on it.
- Object detection combines localization and classification for multiple objects.
- Convolutional sliding windows efficiently implement this using CNNs.
- Bounding box predictions use regression to predict coordinates.
- IoU measures detection accuracy by evaluating overlap.
- Non-max suppression removes redundant and low-confidence boxes. Anchor boxes, pre-defined bounding boxes, enhance accuracy.
- YOLO (You Only Look Once) is a real-time object detection algorithm.
- Semantic segmentation classifies each pixel using the U-Net architecture.
- Region proposals generate potential object locations for detection refinement.

Questions

- What is the correct label y for a 3-class object detection output with classes: pedestrian (1), car (2), motorcycle (3)?
- What should a network output to detect a fixed-size soft-drink can in an image?
- How many output units are needed to detect N facial landmarks?
- Do object detection systems require bounding boxes in training data?
- Does increasing stride in sliding window (non-convolutional) increase accuracy but reduce computation?
- In YOLO, is only the cell containing the object center responsible for detecting it?
- What is the IoU for two boxes (2×2 and 2×3) with a 1×1 overlap?
- After non-max suppression (threshold 0.4, IoU 0.5), how many boxes remain?
- For YOLO with a 19×19 grid, 20 classes, and 5 anchors, what is the output volume dimension?
- Is $y = [pc, bx, by, bh, bw, c1]$ the correct output for soft-drink detection?
- What is the shape of $\hat{y}(i)$ for a face landmark detector with N landmarks?
- Can semantic segmentation only classify pixels in a binary way?
- In transpose convolution (stride=2, padding=1), what are X, Y, Z for the given input and filter?
- Does U-Net always output shape $h \times w \times c$?
- Does U-Net always output shape $h \times w$?
- In YOLO, is each object assigned to its grid cell and the anchor box with highest IoU?
- Is pixel-wise tumor segmentation a localization task?
- Can anchor boxes in YOLO replace the need for bounding box coordinates?