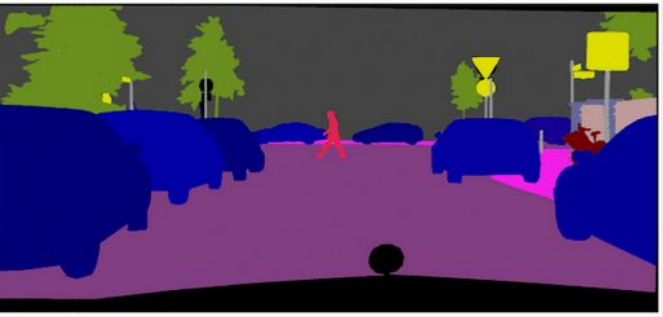


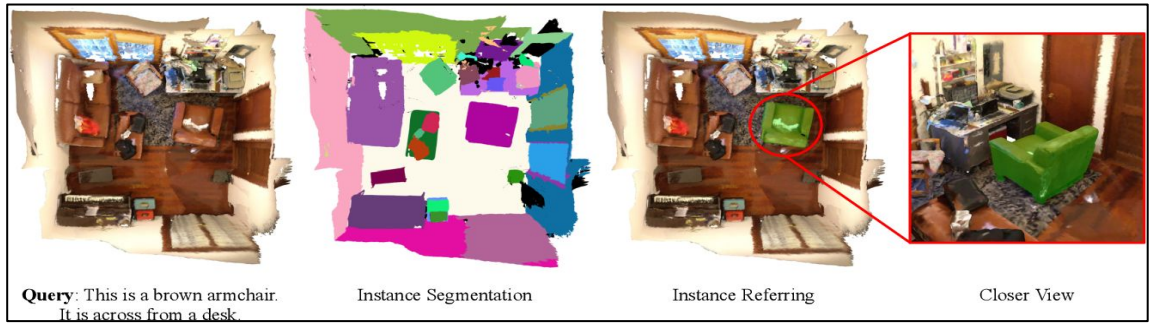


Image Segmentation

Dr. Aditya Nigam

School of Computing and Electrical Engineering (SCEE)

IIT Mandi

Table of Contents

Image Segmentation

What is Image Segmentation?

Image Segmentation is the computer vision task of identifying and locating an object (or objects) in an image by dividing it into irregularly shaped regions, called segments, that correspond to the shapes of different objects or parts of the scene.

Image Segmentation

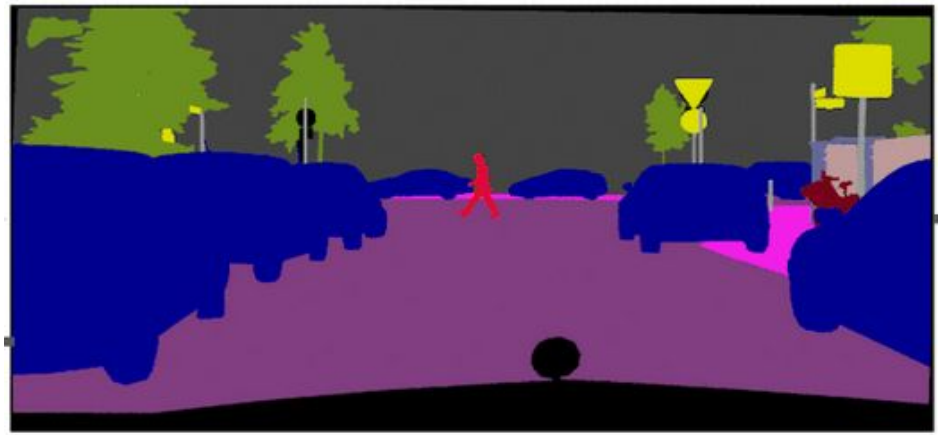
Object Detection vs Segmentation

It is similar to object detection, but while detection is satisfied with rectangular bounding boxes around objects, segmentation traces out the contours of objects at a pixel level.

Object Detection (left) **Image Segmentation (right)**

Image Segmentation : Types

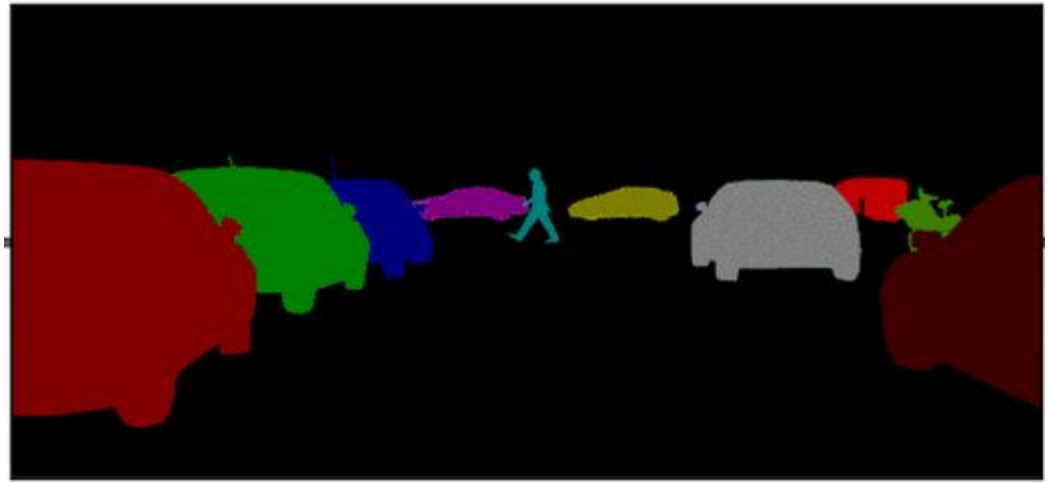
- Based on their core functionality, the **three types of segmentation are semantic, instance, and panoptic.**
- **Semantic Segmentation:** Semantic segmentation is only interested in the classes of objects, not individual objects.
- Each segment corresponds to one class covering the objects of that class, even if they're far apart in the scene.



Semantic Segmentation

Image Segmentation : Types

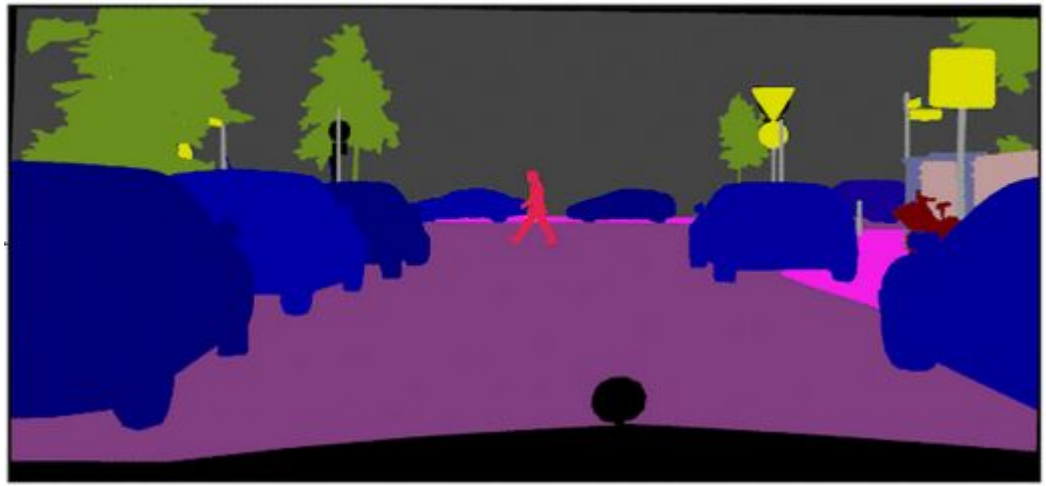
- **Instance Segmentation:** Instance or object segmentation not only identifies the classes but also differentiates each object within a class.
- Each segment corresponds to an individual object instance. It assigns both a class label and a unique object identifier to each pixel. Uncountable background elements, like the sky or the ground, are usually ignored.



Instance Segmentation

Image Segmentation : Types

- **Panoptic Segmentation:** Panoptic segmentation just combines semantic and instance segmentation.
- Like instance segmentation, it differentiates each object within a class. Like semantic segmentation, it labels the background elements, too.



Panoptic Segmentation

Image Segmentation (Advance) : Segmentation Tasks

Based on the nature of their inputs, other types of segmentation tasks are:

- **Referring or text-guided segmentation**: It takes a text description as additional input and segments the objects or background that match that description.

Image Segmentation (Advance) : Segmentation Tasks

Based on the nature of their inputs, other types of segmentation tasks are:

- **Zero-shot segmentation:** Normally, only the classes seen during training are segmented. But zero-shot segmentation goal is to segment even unseen new classes without retraining a model. It does this by relating the unseen classes to known classes.

- **One-shot segmentation:** One-shot segmentation is a type of zero-shot segmentation where each unseen class is provided via an example image. It relates the example image to known classes based on visual similarities.

Image Segmentation (Advance) : Segmentation Tasks

Based on the nature of their inputs, other types of segmentation tasks are:

- **Video segmentation:** Segmentation normally works on images. But when the input is a video, it's called video segmentation. Video segmentation must not only segment objects but also track them with the same identifiers between frames.

Image Segmentation

Modern Deep Learning Approaches to Image Segmentation and Instance Segmentation

- **Transformers** and convolutional neural networks (CNNs) are the two broad approaches to modern segmentation.
- CNNs, like **U-Net** and the **Fully Convolutional Network** for semantic segmentation and **mask R-CNN** for instance segmentation, remain popular.

Image Segmentation : Segmentation Algorithms (A timeline)

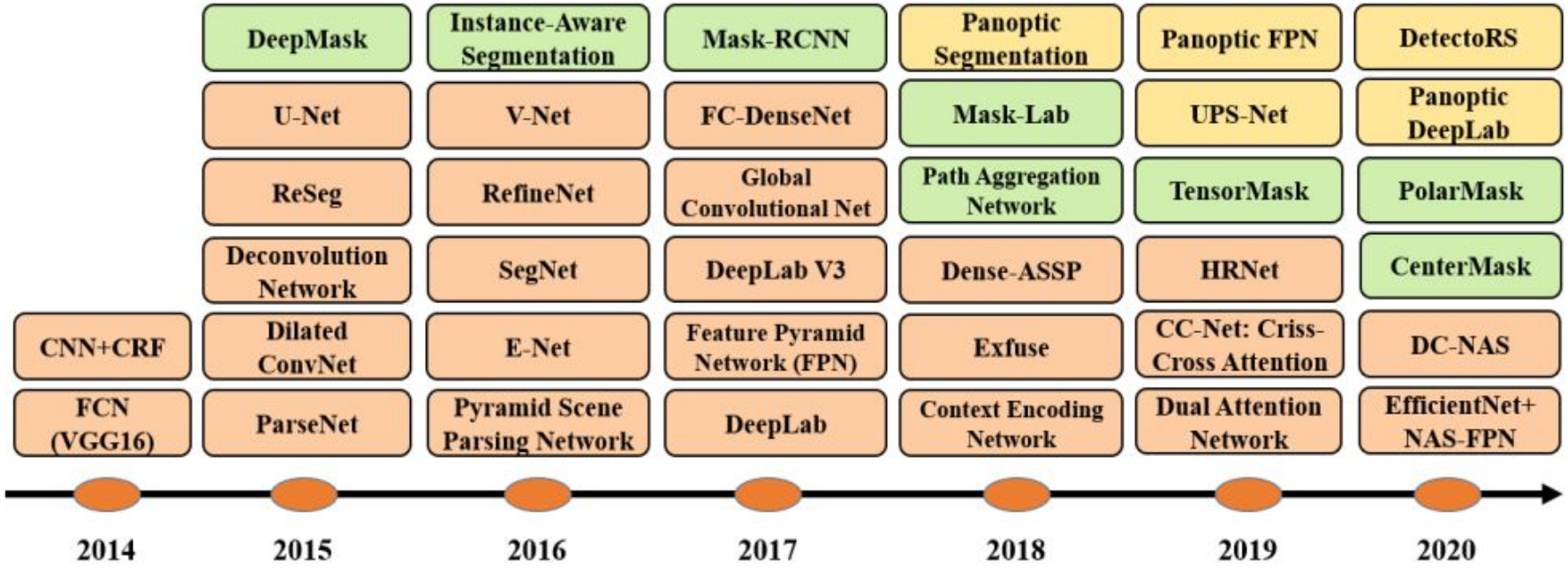
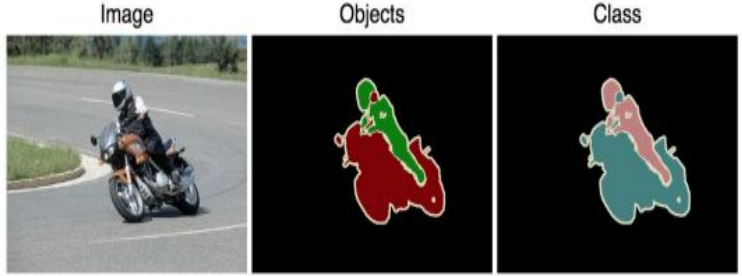


Image Segmentation : Datasets

- **2D Datasets (RGB Images)**
  - PASCAL Visual Object Classes (VOC)
  - PASCAL Context
  - Microsoft Common Objects in Context (MS COCO)
  - Cityscapes
  - ADE20K /MIT Scene Parsing (SceneParse150)
  - Sift Flow
  - Stanford background
  - Berkeley Segmentation Dataset (BSD)
  - Youtube-Objects
  - KITTI …



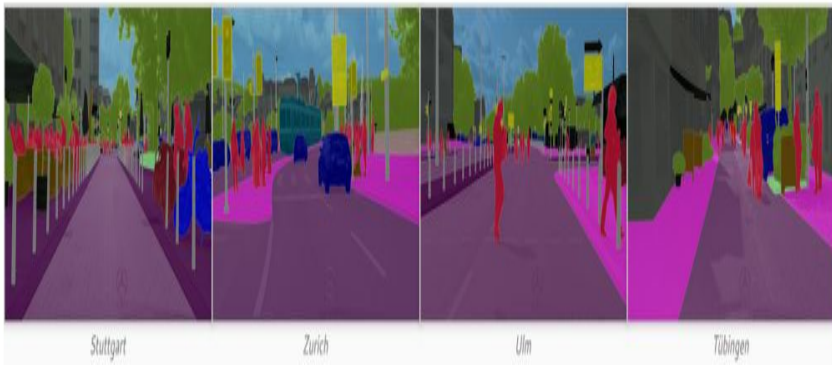


Image Segmentation : Datasets (Contd)

Dataset	Classes	Size	Train	Validation	Test	Resolution (pixels)
PASCAL-Context	540	19740	4998	5105	9637	387×470
ADE20K	150	25210	20210	2000	3000	-
KITTI	5	252	140	-	112	1392×512
Cityscapes	30	5K fine, 20K coarse	2975	500	1525	1024×2048
IDD	34	10004	7003	1000	2001	1678×968
Virtual KITTI	14	21260	-	-	-	1242×375
IDDA	24	1M	-	-	-	1920×1080

Image Segmentation : Datasets (Contd)

- **2.5D Datasets (RGB + Depth images)**
  - NYU-D V2
  - SUN-3D
  - SUN RGB-D
  - UW RGB-D Object Dataset
  - ScanNet

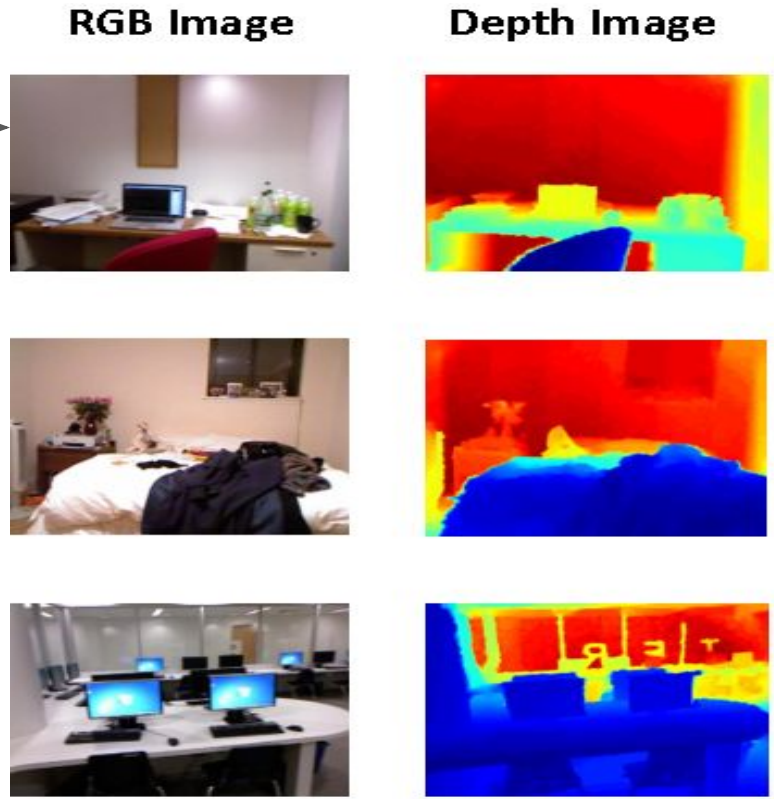
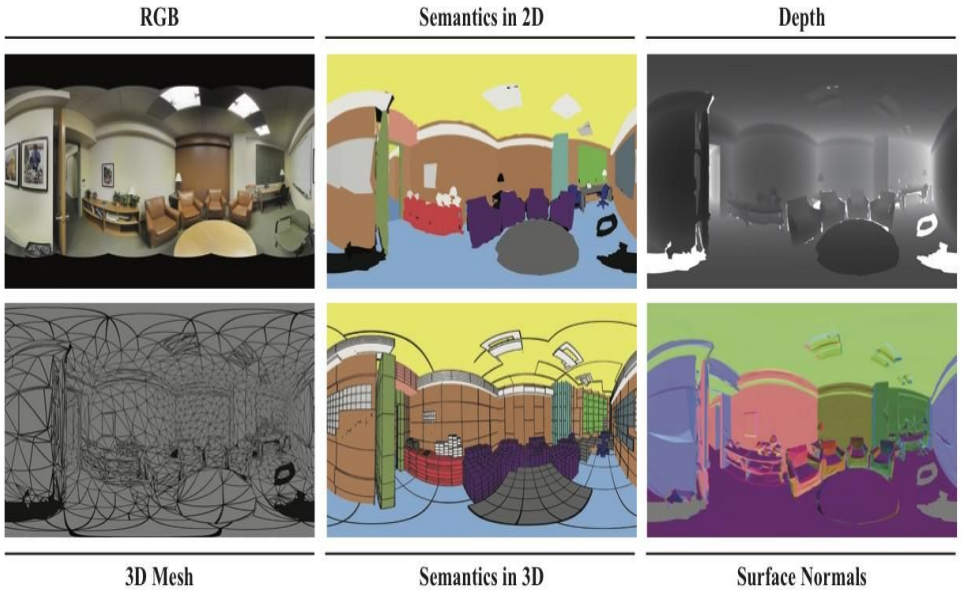
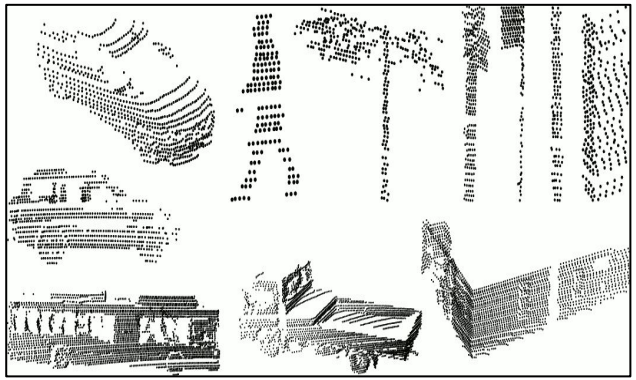


Image Segmentation : Dataset (Contd.)

- **3D Datasets (images are provided via meshes or other volumetric representations, such as point clouds.)**
  - Stanford 2D-3D
  - ShapeNet Core
  - Sydney Urban Objects Dataset





Point Clouds

Image Segmentation: Supervised Learning based Segmentation (Evolution)

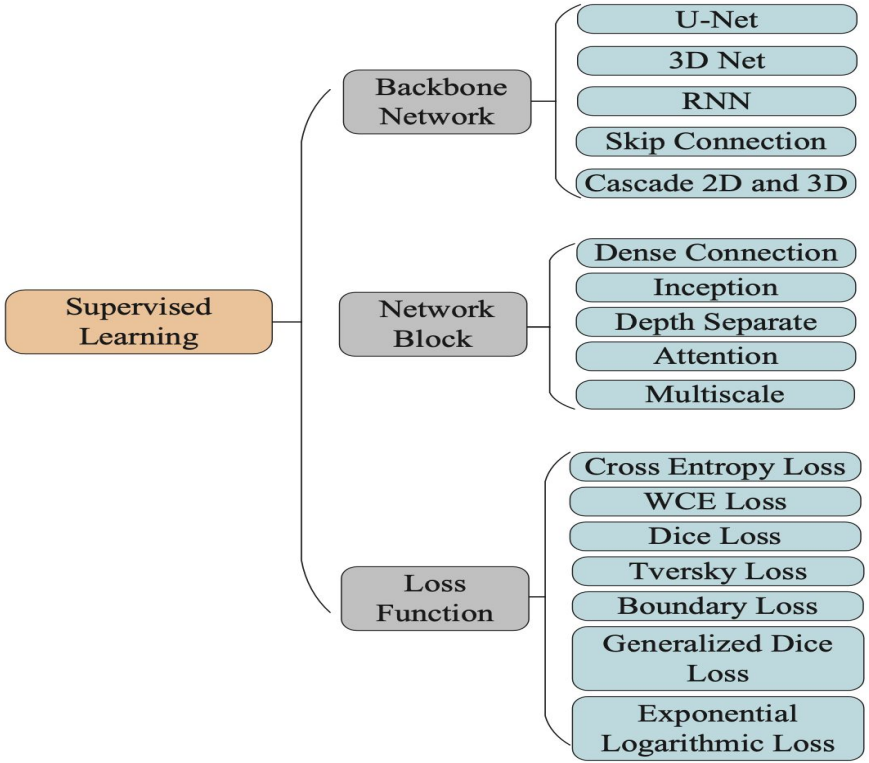


Image Segmentation : Loss Function

- Cross Entropy Loss
  - $CE(p,\hat{p}) = -(p\log(\hat{p}) + (1-p)\log(1-\hat{p}))$
- Weighted Cross Entropy Loss
  - $WCE(p,\hat{p}) = -(\beta p\log(\hat{p}) + (1-p)\log(1-\hat{p}))$
- Dice Loss
  - $\mathrm{Dice}(A,B) = \frac{2\times|A\cap B|}{A+B}\times 100\,\%$

where p is the probability of the sample belongs to the True class and $\hat{p}$ is the predicted value. (In case of Binary segmentation.)

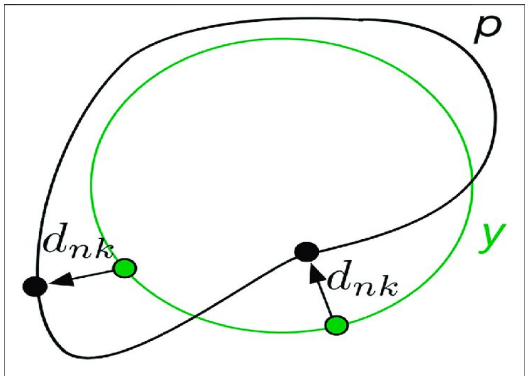
where A is a predicted segmentation result and B is a real segmentation result. A and B are segments.

Image Segmentation : Loss Function (Contd)

- Tversky Loss
  - $$TL(p, \hat{p}) = \frac{p, \hat{p}}{p, \hat{p} + \beta(1 - p, \hat{p}) + (1 - \beta)(p, 1 - \hat{p})}$$

where p is the probability of the segment belongs to the True class and p̂ is the predicted value. β is the regularization constant.

- Boundary Loss



- Boundary loss used in semantic segmentation tasks to encourage precise boundary localization.
- It addresses the problem of blurry or imprecise boundaries in segmented regions by penalizing misalignments between predicted boundaries and ground truth boundaries.
- The loss encourages the network to produce sharper and more accurate boundaries.

Image Segmentation : Evaluation Metrics

- **Precision :** $P = \frac{\text{TP}}{\text{TP} + \text{FP}}$ **Recall:** $R = \frac{\text{TP}}{\text{TP} + \text{FN}}$ **F1 - score:** $\text{F1 Score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
- **Accuracy:** $\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$

- **Pixel Accuracy:** ratio of pixels properly classified, divided by the total number of pixels. For K + 1 classes (K foreground classes and the background) pixel accuracy is defined as:

$$\text{PA} = \frac{\sum_{i=0}^{K} p_{ii}}{\sum_{i=0}^{K} \sum_{j=0}^{K} p_{ij}}$$

where pij is the number of pixels of class i predicted as belonging to class j.

- **Mean Pixel Accuracy (MPA):** ratio of correct pixels computed in a per-class manner and averaged over the total number of classes.

$$\text{MPA} = \frac{1}{K+1} \sum_{i=0}^{K} \frac{p_{ii}}{\sum_{j=0}^{K} p_{ij}}.$$

Image Segmentation : Evaluation Metrics

- **Jaccard Index / Intersection over Union (IoU):** area of intersection between the predicted segmentation map and the ground truth, divided by the area of union between the predicted segmentation map and the ground truth.

$$\text{IoU} = J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

where A and B denote the ground truth and the predicted segmentation maps, respectively. It ranges between 0 and 1.

- **Dice coefficient:** similar to IoU. Defined as: Twice the overlap area of predicted and ground-truth maps, divided by the total number of pixels in both images.

$$\text{Dice} = \frac{2|A \cap B|}{|A| + |B|}$$

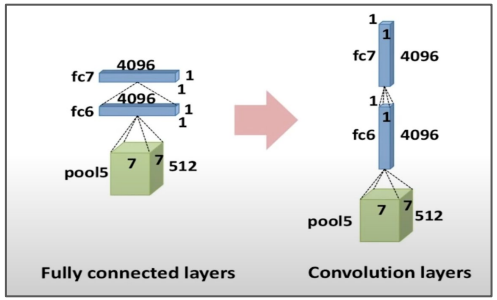
where A and B denote the ground truth and the predicted segmentation maps, respectively. It ranges between 0 and 1.

CNN Based

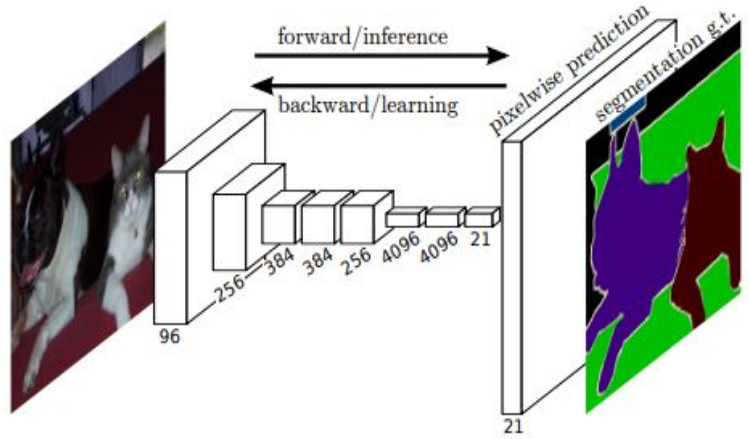
Fully Convolutional Networks: Architecture

J. Long, E. Shelhamer and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 3431-3440, doi: 10.1109/CVPR.2015.7298965.

- One of the first deep learning works for semantic image segmentation.
- Modifies existing CNN architectures, such as VGG16 by replacing all fully-connected layers with the fully-convolutional layers.



- As a result, the model outputs a spatial segmentation map instead of classification scores.

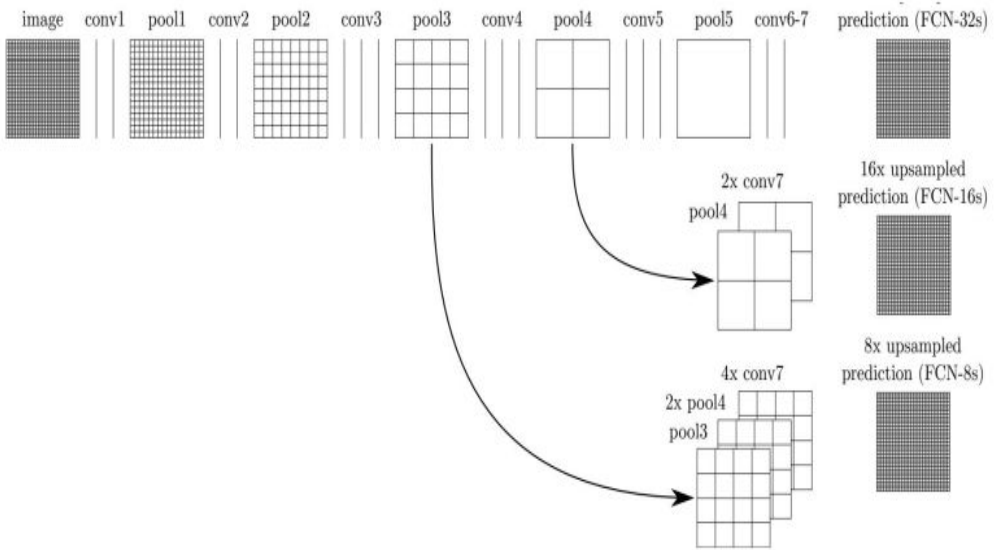


A fully convolutional image segmentation network

CNN Based

Fully Convolutional Networks: Skip Connections

- Semantic segmentation has an inherent tension between semantics and location: global information resolves *what* while local information resolves *where*.
- Skip connections: feature maps from the final layers of the model are up-sampled and fused with feature maps of earlier layers.
- Combines semantic information (from deep, coarse layers) and appearance information (from shallow, fine layers)

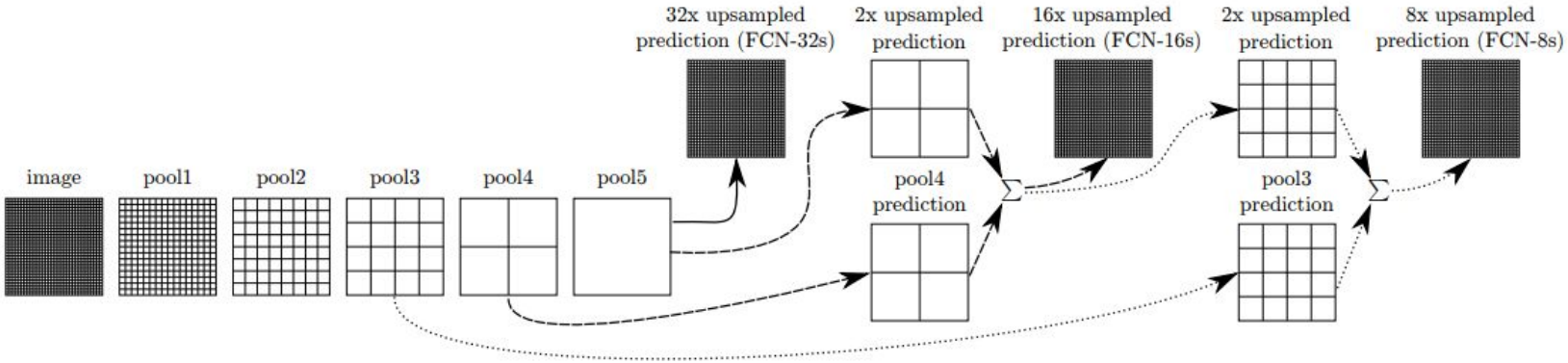


Skip connections combine coarse, high-level information and fine, low-level information to produce accurate segmentations.

CNN Based

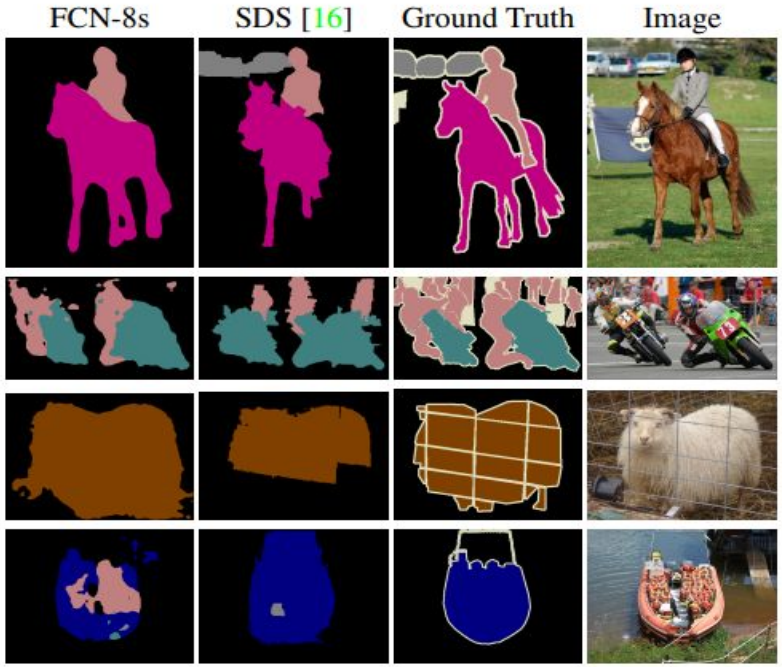
Fully Convolutional Networks: Skip Connections

- Divide the output stride in half by predicting from a 16 pixel stride layer. Add a 1 × 1 convolution layer on top of pool4 to produce additional class predictions.
- Fuse this output with the predictions computed on top of conv7 at stride 32 by adding a 2× upsampling layer and summing both predictions. This is called FCN-16s.
- Continue in this fashion by fusing predictions from pool3 with a 2× upsampling of predictions fused from pool4 and conv7, building the net FCN-8s.



CNN Based

Fully Convolutional Networks: Performance



Fully convolutional segmentation performance on PASCAL Dataset

Dataset	Pixel acc.	Mean acc.	Mean IU	f.w. IU
Pascal VOC	62.7	-	-	-
NYUDv2	65.4	46.1	34.0	49.5
SIFT Flow	85.2	51.7	39.5	76.1

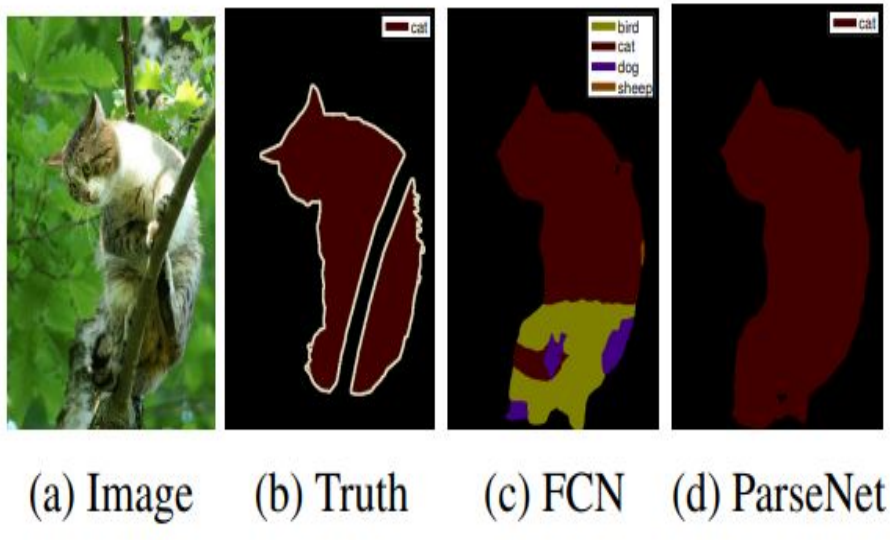
Performance on various Datasets

CNN Based

ParseNET

Liu, W., Rabinovich, A., & Berg, A. C. (2015). *ParseNet: Looking Wider to See Better*. 1–11. http://arxiv.org/abs/1506.04579

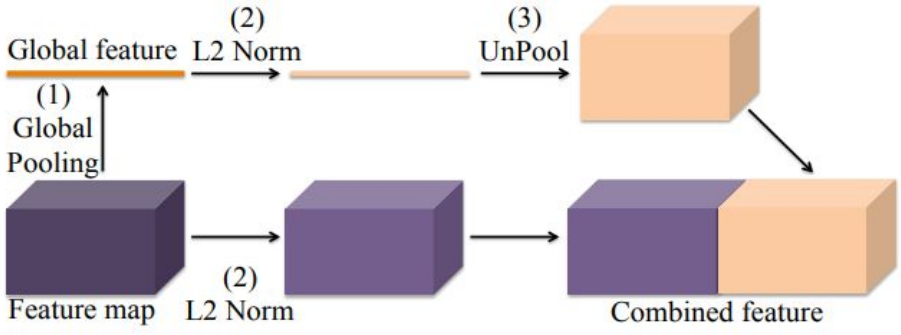
- **FCN's Drawback**: Does not deal with the global context information efficiently.



CNN Based

ParseNET

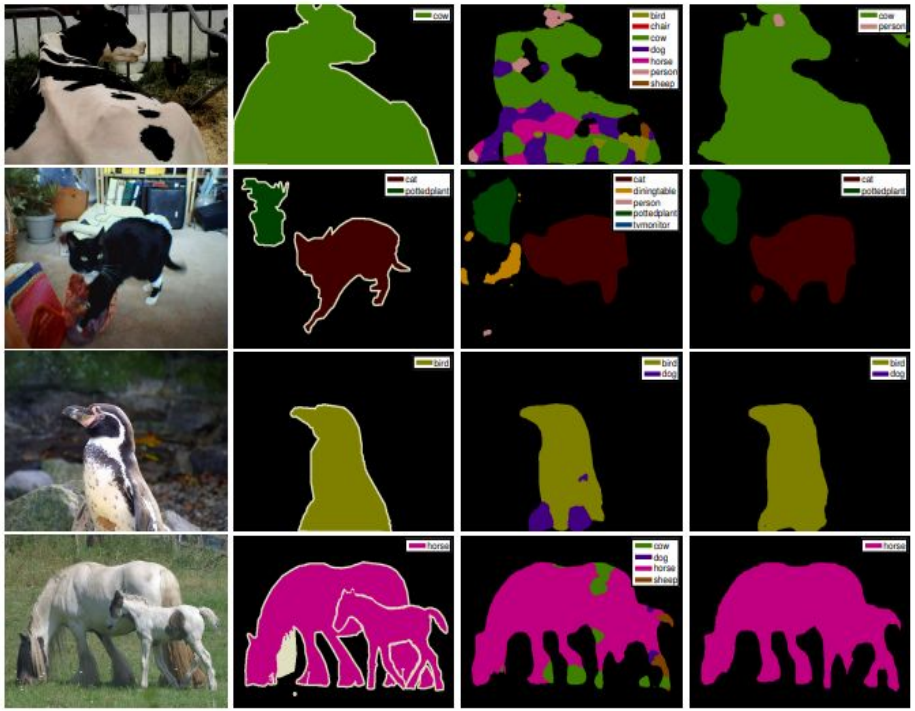
- **Solution**: ParseNet adds global context to FCNs. The feature map for a layer is pooled over the whole image to result in a context vector. The context vector is unpooled and the resulting feature map is appended with the standard feature map.
- L2 normalization addresses the different scale of features from different layers.



(e) ParseNet contexture module overview.

CNN Based

ParseNET: Performance



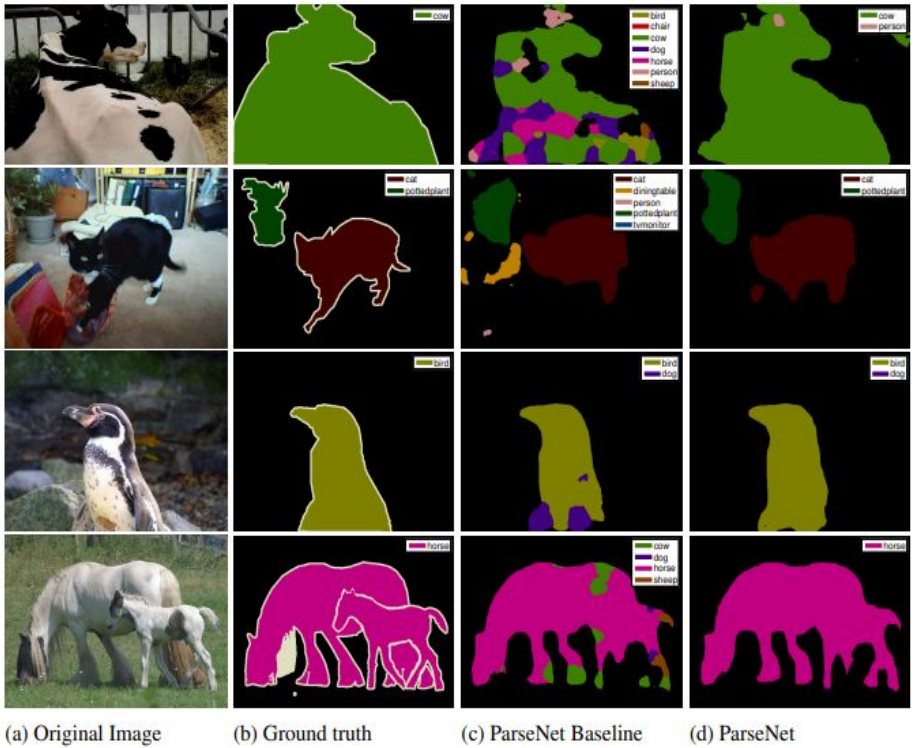
(a) Original Image (b) Ground truth (c) ParseNet Baseline (d) ParseNet

Dataset	Pixel acc.	Mean acc.	Mean IU	f.w. IU
Pascal VOC	65.82	-	-	-
Pascal Context	-	-	36.64	-
SIFT Flow	86.8	52.0	40.4	78.1

Global context helps for classifying local patches

CNN Based

ParseNET: Performance



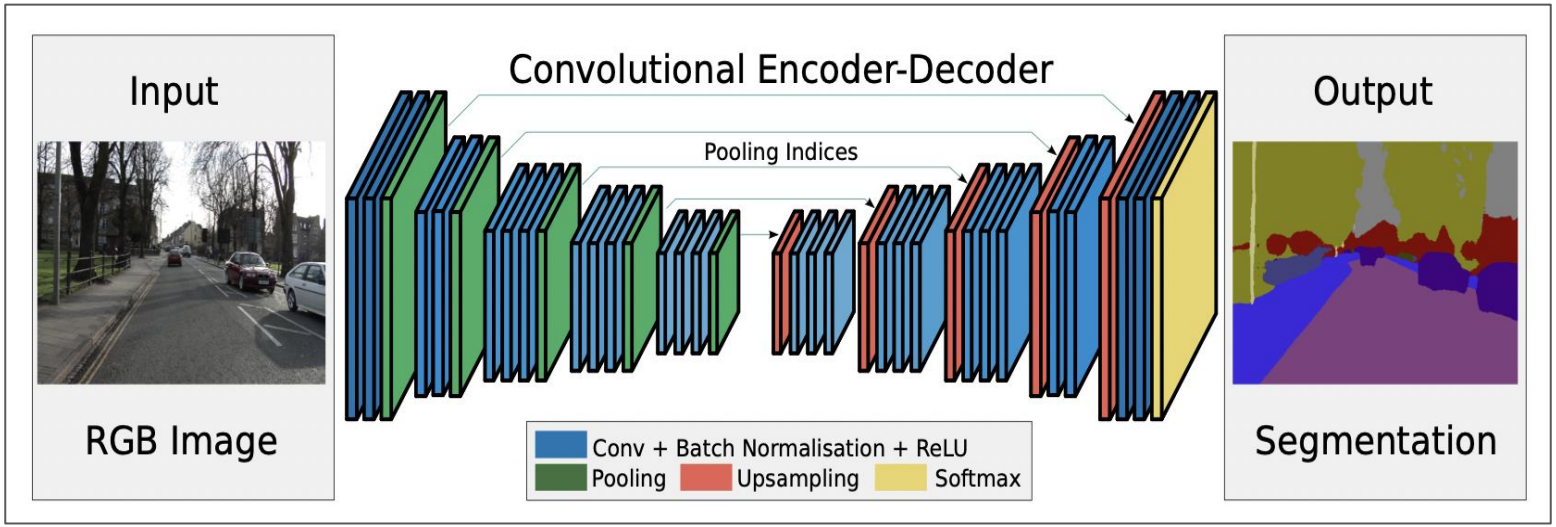
- Demonstrates that relying on the largest receptive field of FCN network does not provide sufficient global context and is not sufficient to capture global context.
- Therefore, modeling global context directly in required.

Global context helps for classifying local patches

Encoder-Decoder Based

SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(12), 2481–2495. https://doi.org/10.1109/TPAMI.2016.2644615

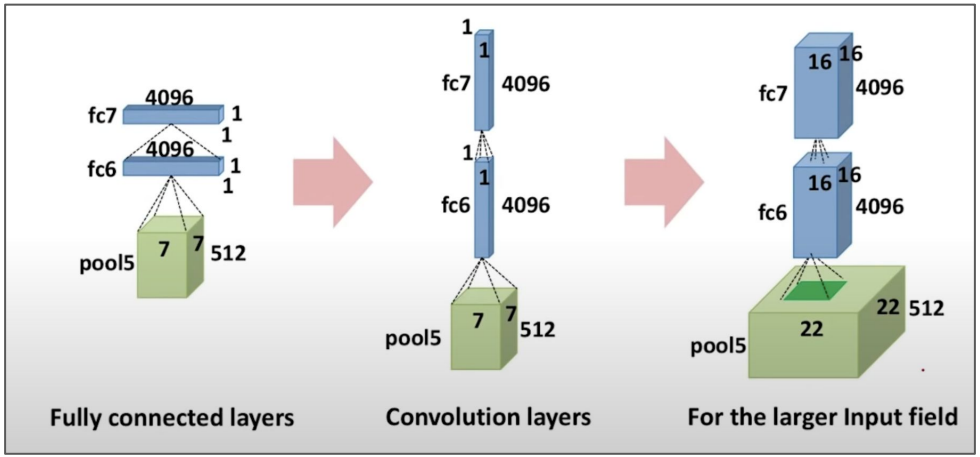


Semantic pixel-wise segmentation using SegNet

SegNet: (FCNet) Background

Fully Convolutional Segmentation models

- Convolutionalization of FC layers refers to replacing fully connected layers with convolutional layers in a neural network architecture, particularly in segmentation models.
- Adapting a classification model like VGG16, Google net to fully convolutional networks helps transfer their learned representations by fine-tuning to the segmentation task.
- Also handles variable input image sizes, which is otherwise fixed by the dimension of dense layer.



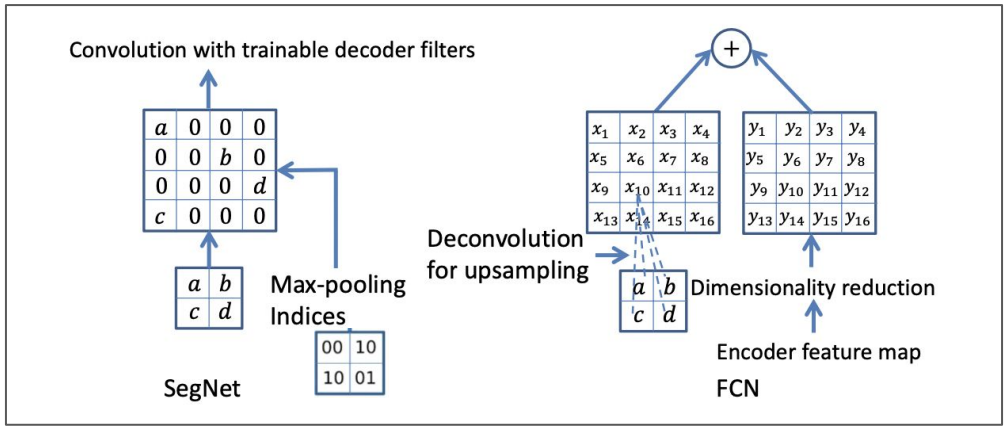
SegNet: Network Features

Fully Convolutional Encoder-Decoder Segmentation models

- SegNet Architecture has:
  - **Encoder Network**: understands the what of the image
  - **Decoder Network**: understands where of the problem [Localization]
  - **Final pixel-wise classification layer** [Softmax]
- Loss function : Sum of pixel-wise Cross Entropy Losses.
- The network is trained such that encoder extracts most relevant features from input.
- Decoder uses these features to result in pixel-wise labelling of the input.
- Upsampling is performed so that output is the same size as input image.

SegNet: Key Contribution: Upsampling

- Unlike FCN, which had only one upsampling layer at the end, SegNet has a sequence of decoder layers.
- Each Decoder stage uses pooling Indices of corresponding max-pool layers to perform upsampling.
- Max-pool indices: Location of maximum feature value for each pooling window for each encoder feature map.

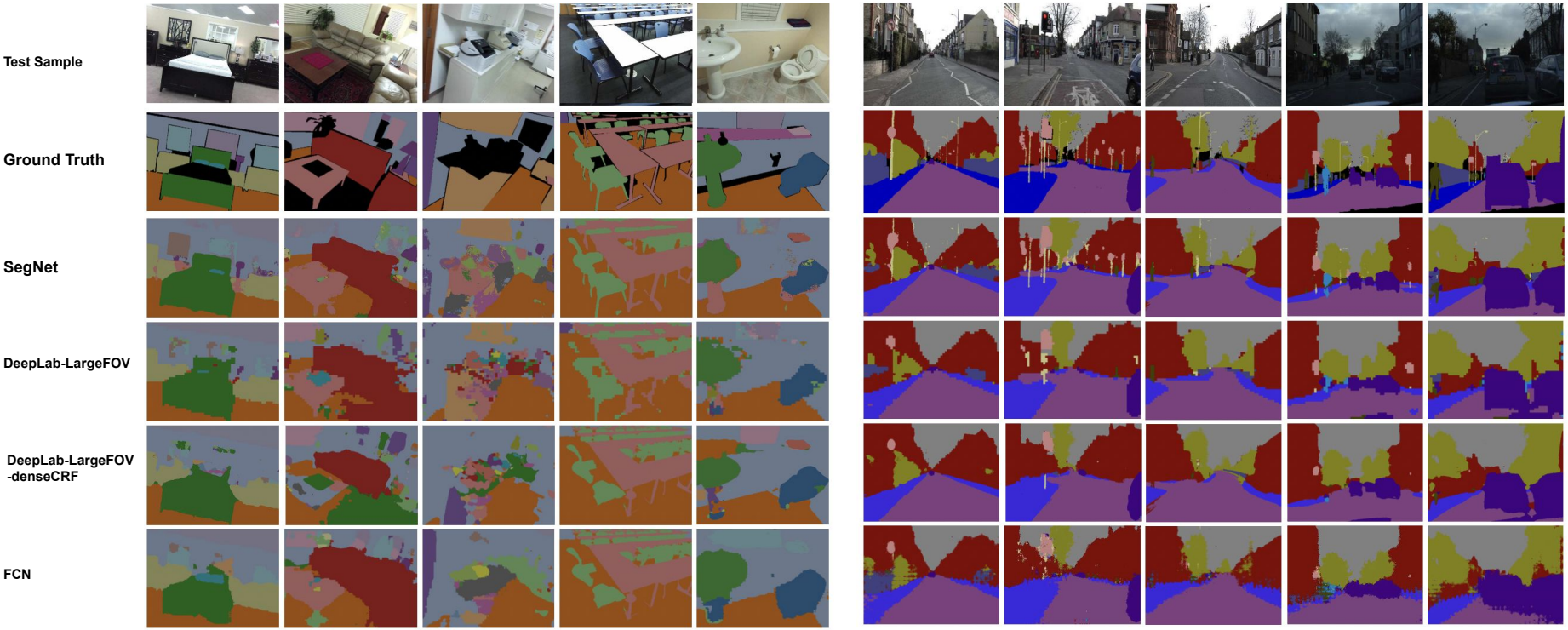


- The resulting sparse feature maps are followed by convolution layers to obtain dense feature maps.
- Memory efficient, compared to storing full encoder feature maps [11 times less than FCN].
- Output: Volume having the size of input image resolution and C+1 no. of channels, where C=no. of classes.

SegNet: Performance

Indoor Scene Segmentation: SUN RGB-D dataset

Road Scene Segmentation: CamVid dataset



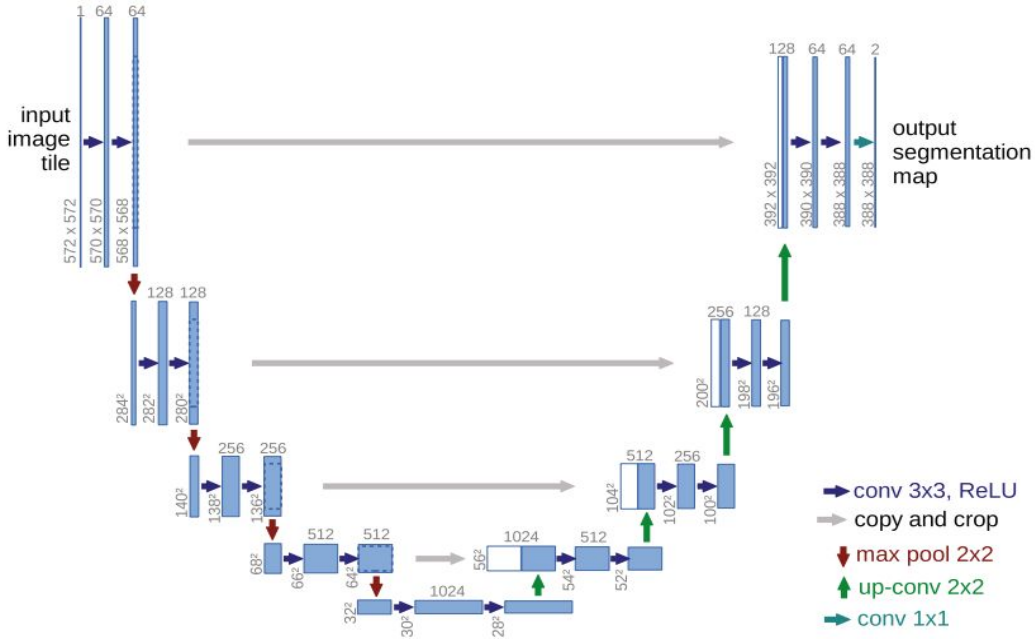
SegNet shows superior performance, particularly with its ability to delineate boundaries, qualitatively. It also outperforms FCN, DeepLab and DeconvNet by obtaining highest Global average accuracy(G), class average accuracy(C) and mIOU.

Encoder-Decoder Based

U-Net

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18 (pp. 234-241). Springer International Publishing.

- Unet modifies and extends FCN. Perfect symmetry.
- Initially proposed for segmenting biological microscopy images
- In FCNs, high resolution features from the contracting path are combined with the upsampled output. (Skip Connections)
- Major Modification: the upsampling part has a large number of feature channels, which allows the network to propagate context information to higher resolution layers.

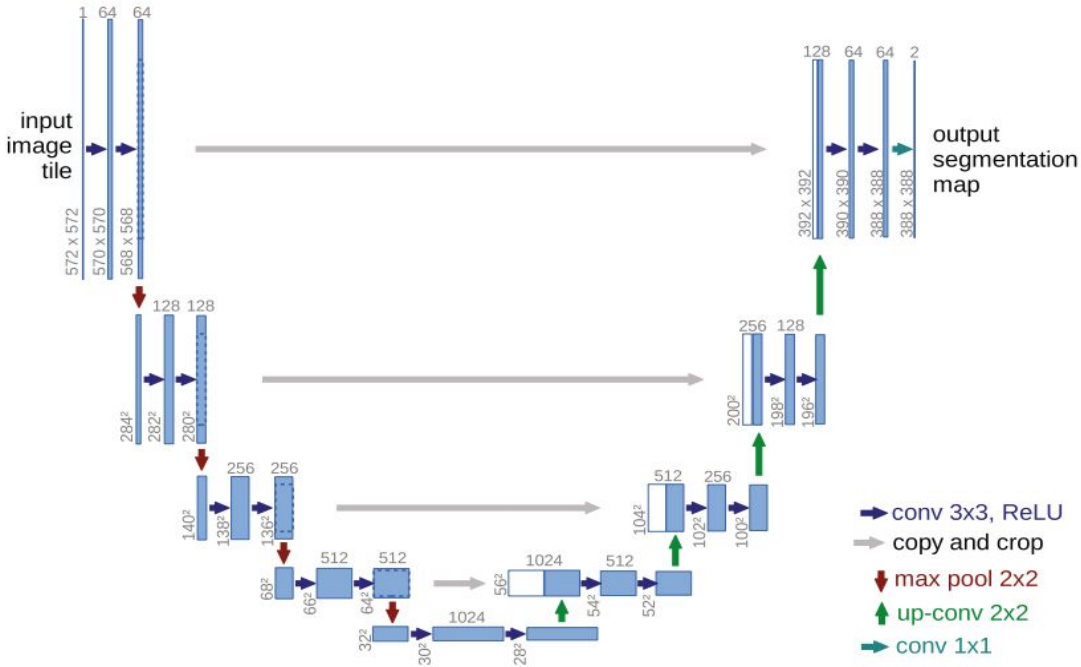


U-net architecture (each blue box: a multi-channel feature map - size and shape on top and lower left for 32 x 32 pixels)

Encoder-Decoder Based

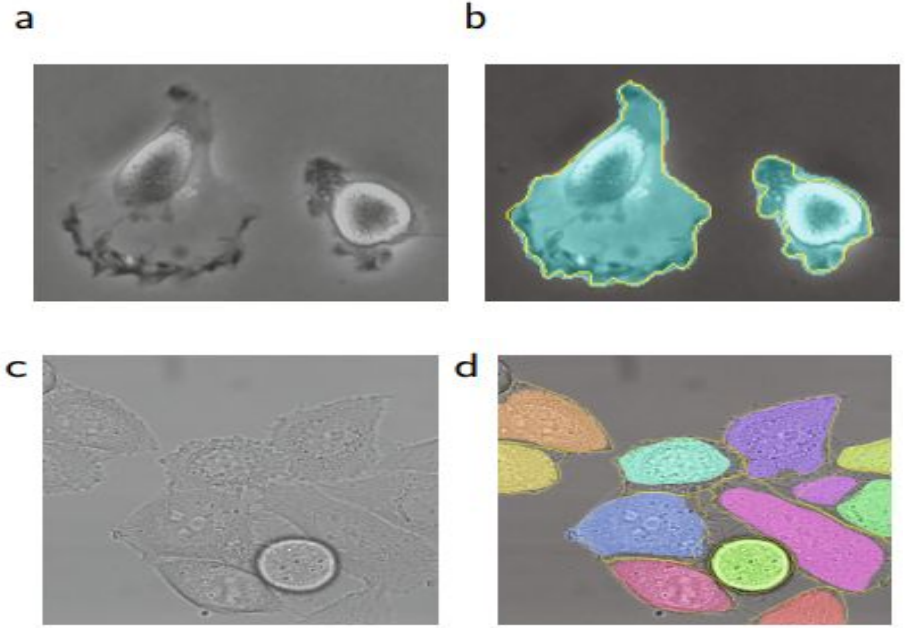
U-Net : Architecture

- Downsampling: two 3x3 convolutions, each followed by a ReLU and a 2x2 max pooling
- Upsampling of the feature map is followed by a 2x2 convolution (up-convolution), a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, followed by ReLU
- Finally, a 1×1 convolution processes the feature maps to generate a segmentation map that categorizes each pixel of the input image.



U-net architecture (each blue box: a multi-channel feature map - size and shape on top and lower left for 32 x 32 pixels)

U-Net: Performance



- The U-net architecture achieves good performance on very different biomedical segmentation applications.
- Results on PhC-U373 and DIC-HeLa dataset for average IoU (intersection over union):

Dataset	Average IOU
PhC-U373	0.9203
DIC-HeLa	0.7756

(a) input image of the “PhC-U373” data set.
(b) Segmentation result (cyan mask) with manual ground truth (yellow border)
(c) input image of the “DIC-HeLa” data set.
(d) Segmentation result.

VNet (for 3D/Volumetric Data)

- For 3D volumes in (Medical)
- Due to more parameters residual connections are used for effective training by avoidir vanishing gradients and faster convergence.
- Residual help to have deeper networks.
- But too many parameters, memory and GPUs.

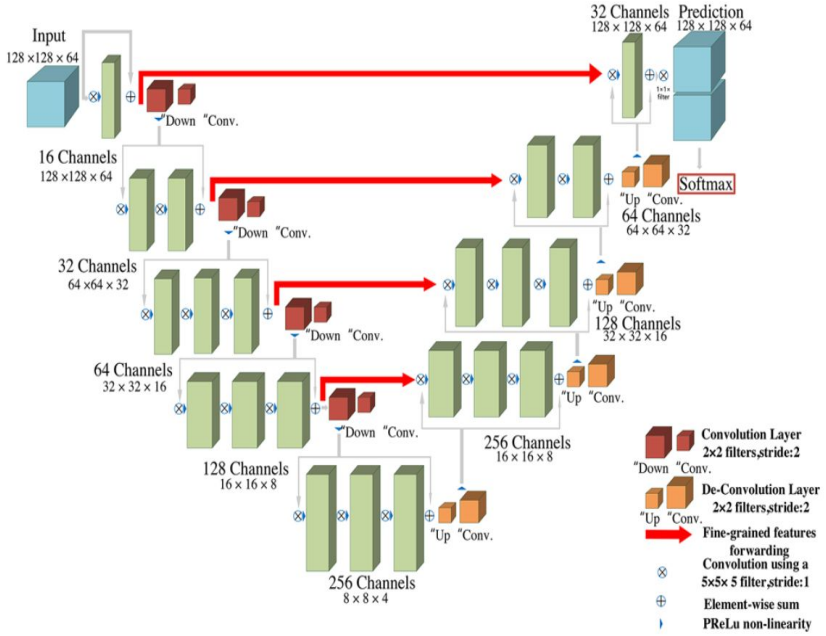


FIGURE 4 The V-Net architecture [35]

Dense Connection Architecture (Improved U-Net)

- Input of each layer comes from the o/p of all previous layers
- Replacing each sub block with dense connections
- Useful for rich features but less robust due to more parameters.

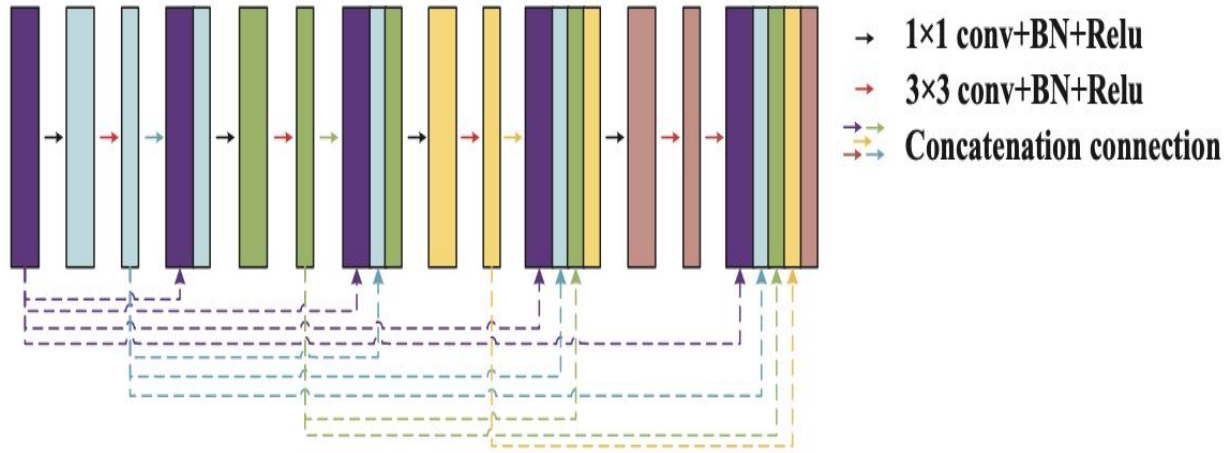


FIGURE 6 Dense connection architecture [70]

U-Net++ Architecture

- Learn feature importance of different layers
- Skip connection are designed to add different types of inputs bridging semantic gap.
- Weight pruning to manage parameters due to dense conn.
- BCE and DC loss.
- Deep Supervision

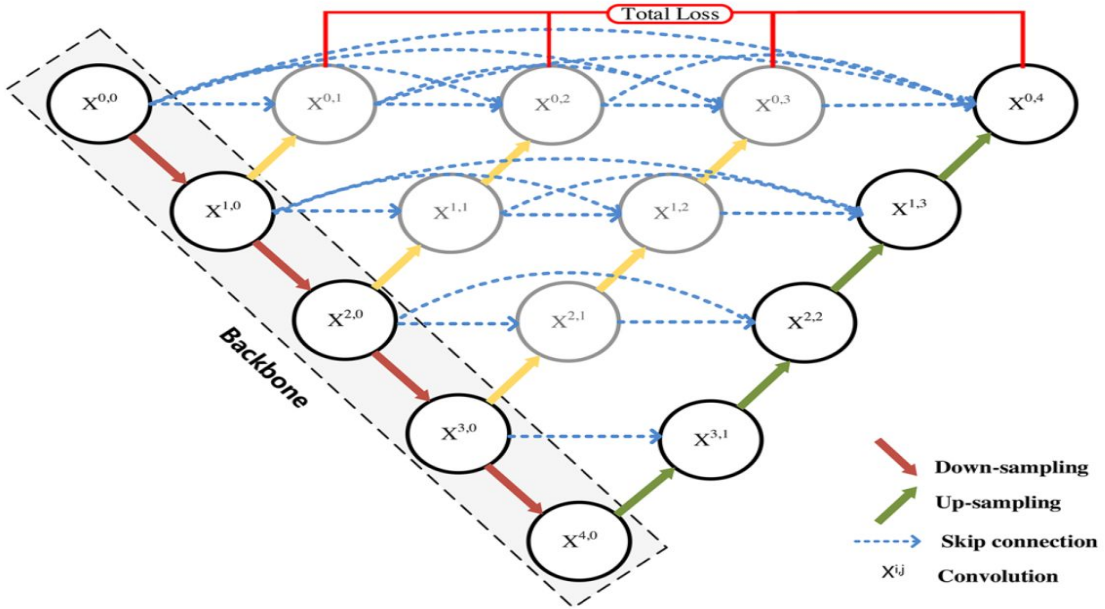


FIGURE 7 The U-Net++ architecture [71]

CE-Net using Inception and Atrous Convolution

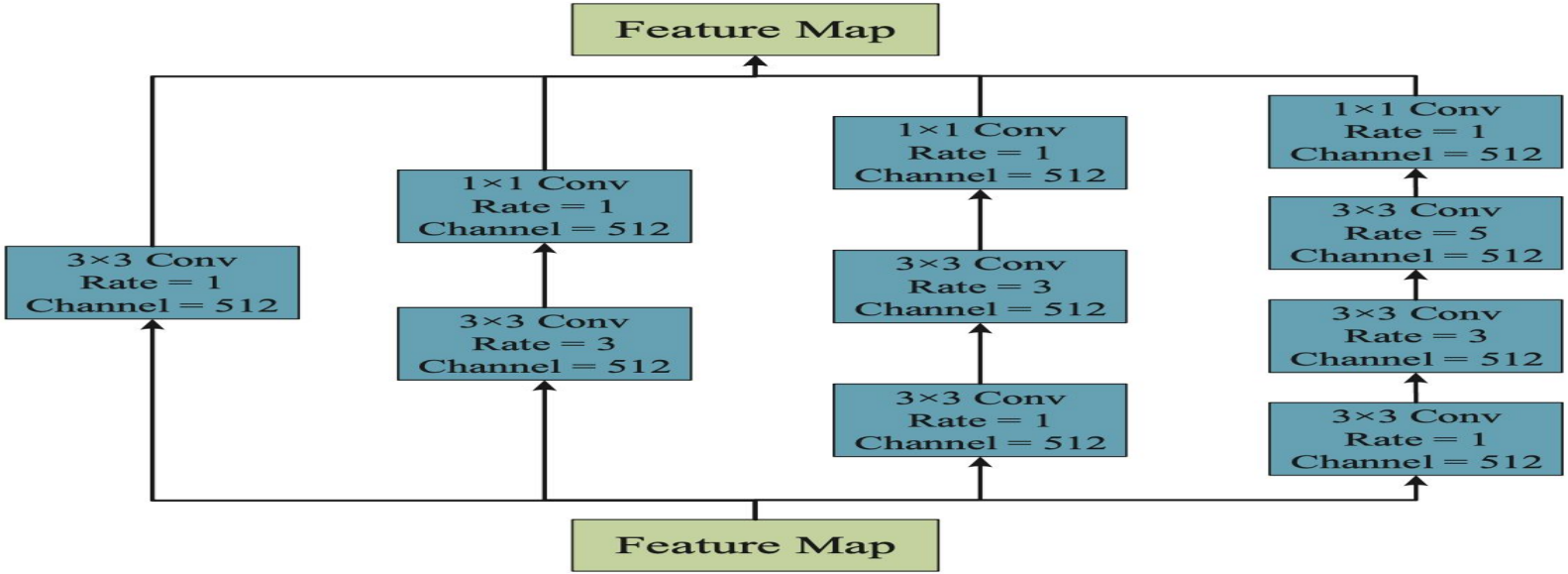
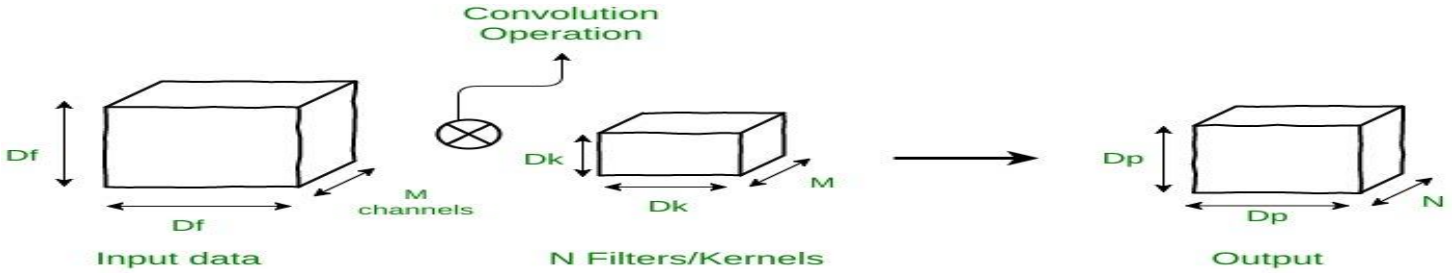
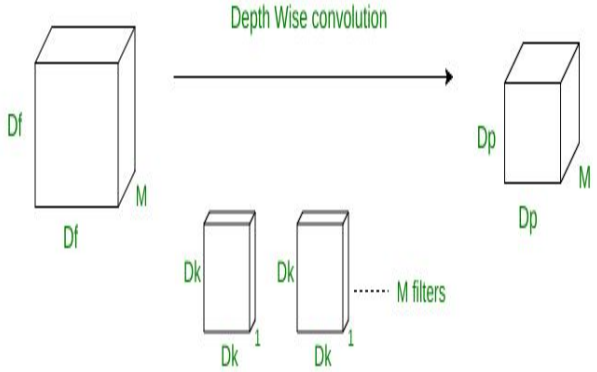
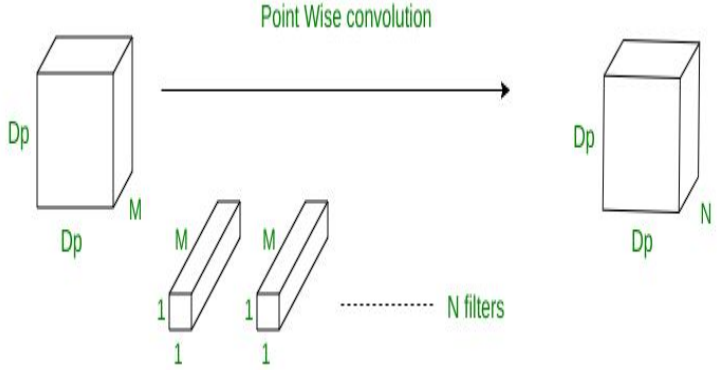


FIGURE 8 The inception architecture [75]. It contains four cascade branches with the gradual increment of the number of atrous convolution, from 1 to 1, 3, and 5, then the receptive field of each branch will be 3, 7, 9, and 19. Therefore, the network can extract features from different scales

MobileNet-V1,V2, LVNet (Depthwise Convolution)

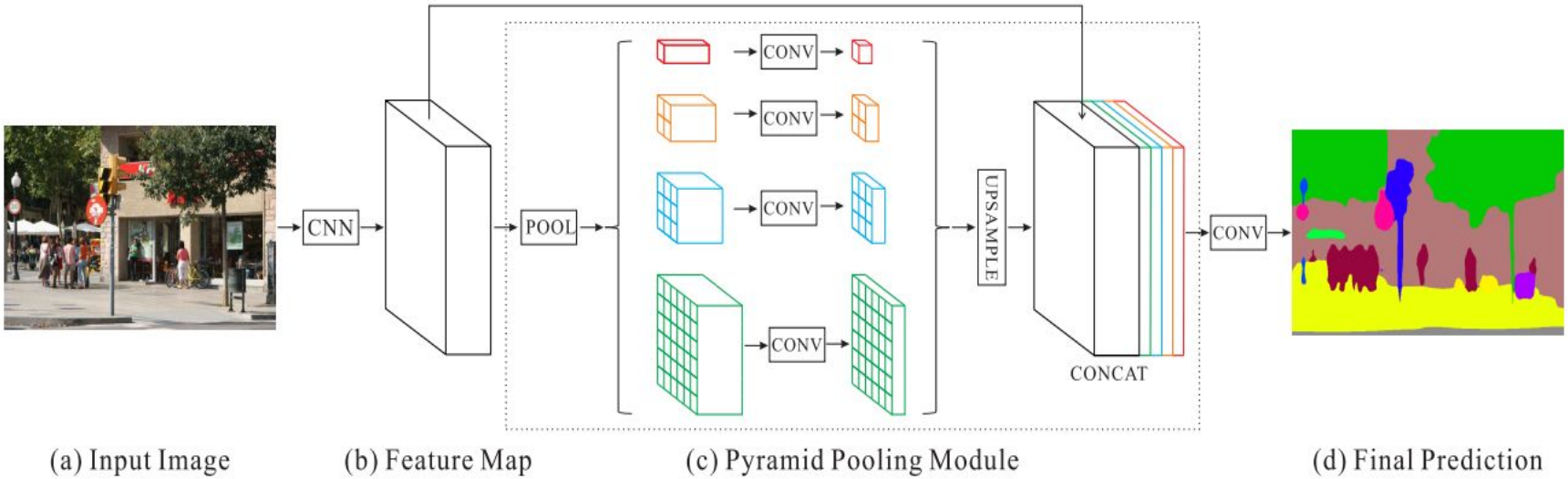






PSPNet : Pyramid Scene Parsing Network

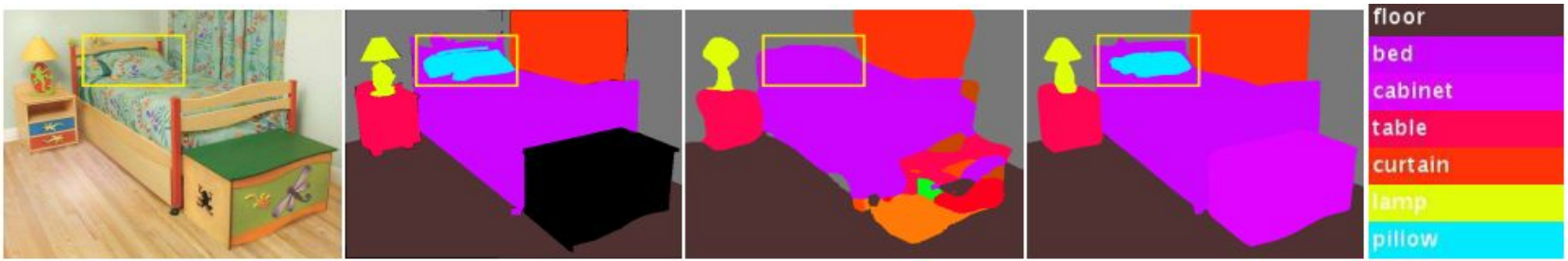
Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-Janua*, 6230–6239. https://doi.org/10.1109/CVPR.2017.660



A pyramid parsing module is applied to get different sub-region representations, followed by upsampling and concatenation layers to get final feature representation

PSPNet : Architecture

- Different patterns are extracted from the input image using a residual network (ResNet) as a feature extractor, with a dilated network.
- These feature maps are then fed into a pyramid pooling module to distinguish patterns of different scales. They are pooled at four different scales, each one corresponding to a pyramid level and processed by a 1 × 1 convolutional layer to reduce their dimensions.
- The outputs of the pyramid levels are up-sampled and concatenated with the initial feature maps to capture both local and global context information. Finally, a convolutional layer is used to generate the pixel-wise predictions

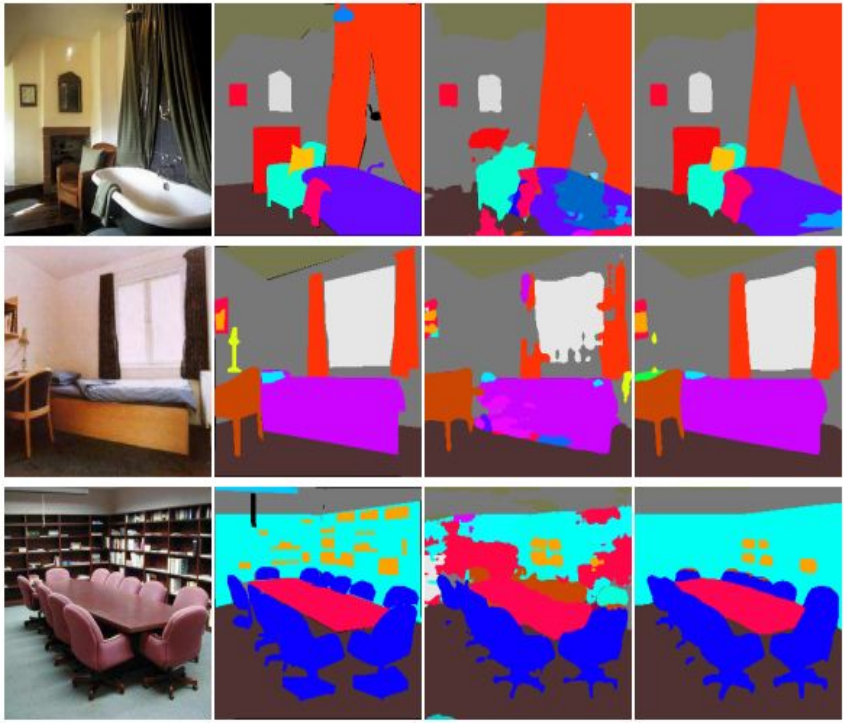


(a) Image (b) Ground Truth (c) FCN (d) PSPNet (e) ColorMap

Performance on ADE20K Dataset

Multi-Scale and Pyramid Network Based Models

PSPNet : Performance



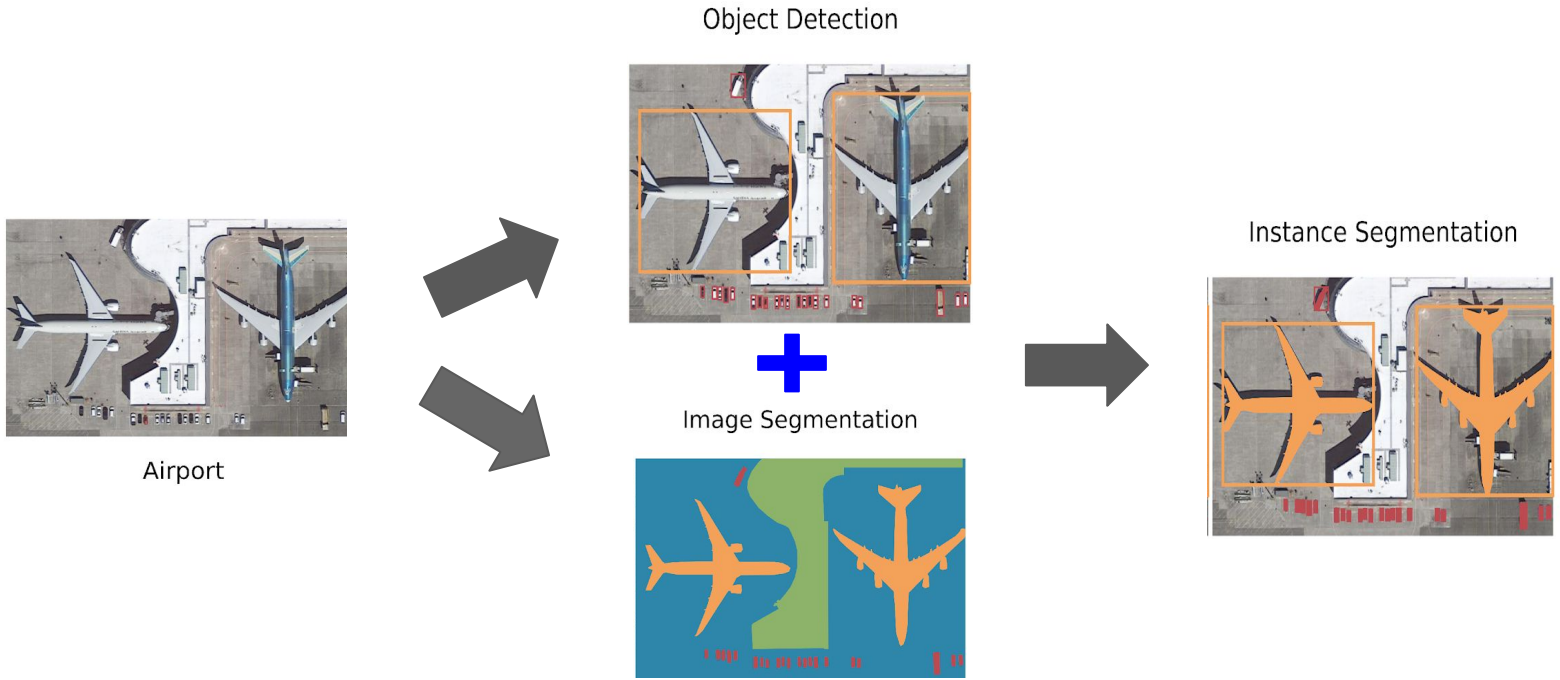
(a) Image (b) Ground Truth (c) Baseline (d) PSPNet

More Results on ADE20K Dataset

(a) Image (b) Ground Truth (c) Baseline (d) PSPNet

Visual improvements on PASCAL VOC

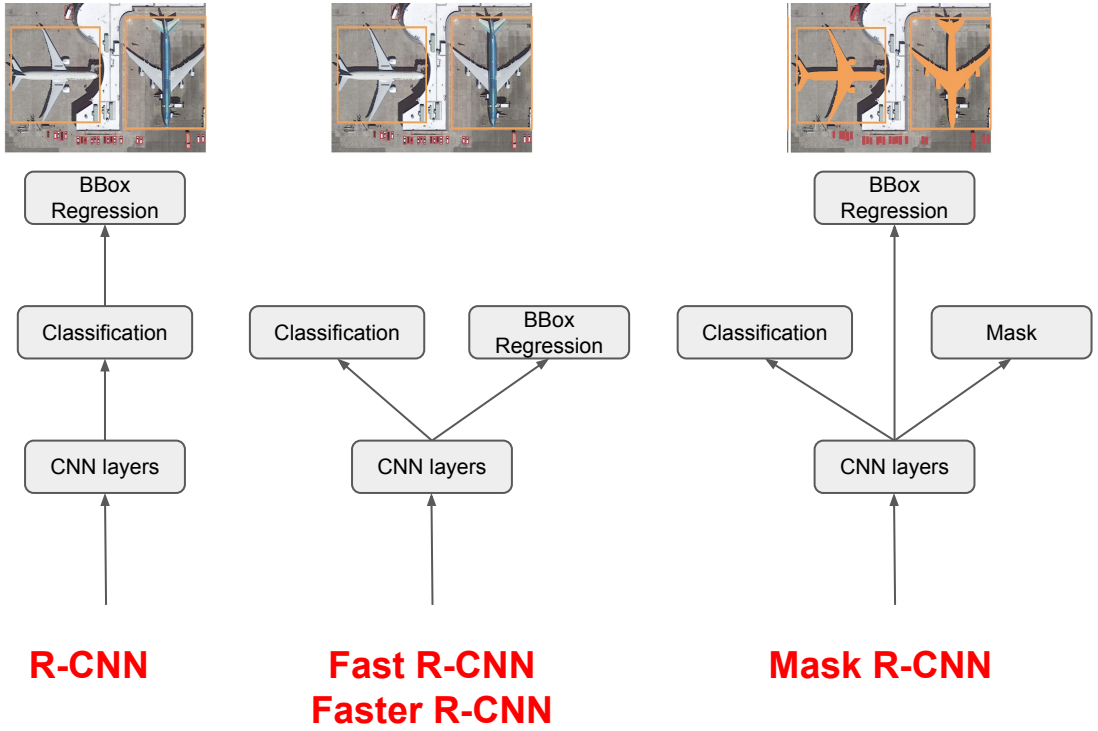
Mask Region based CNN (Mask R-CNN) : Motivation



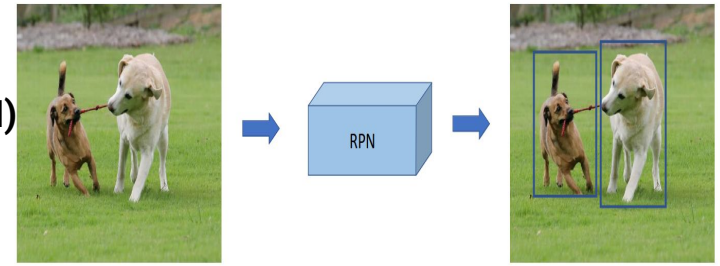
- Main **motivation** is to simultaneously detect and segment individual objects within an image and provide pixel-level segmentation masks for precise object boundary delineation.

Mask R-CNN

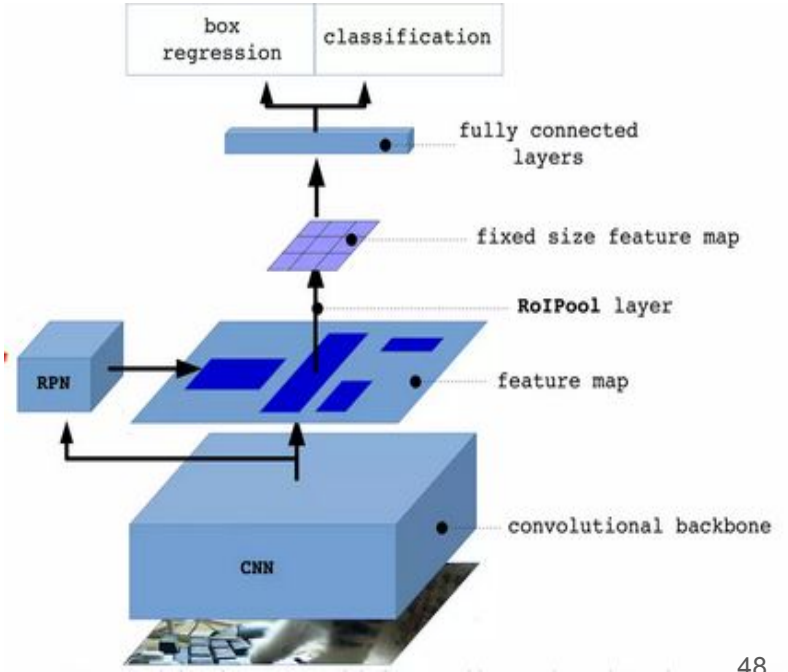
- Approach to instance segmentation.
- Two Subproblems-
  - Object detection
  - Semantic segmentation
- Builds on top of Faster R-CNN by adding a parallel branch.



Mask R-CNN : Background (Faster R-CNN)

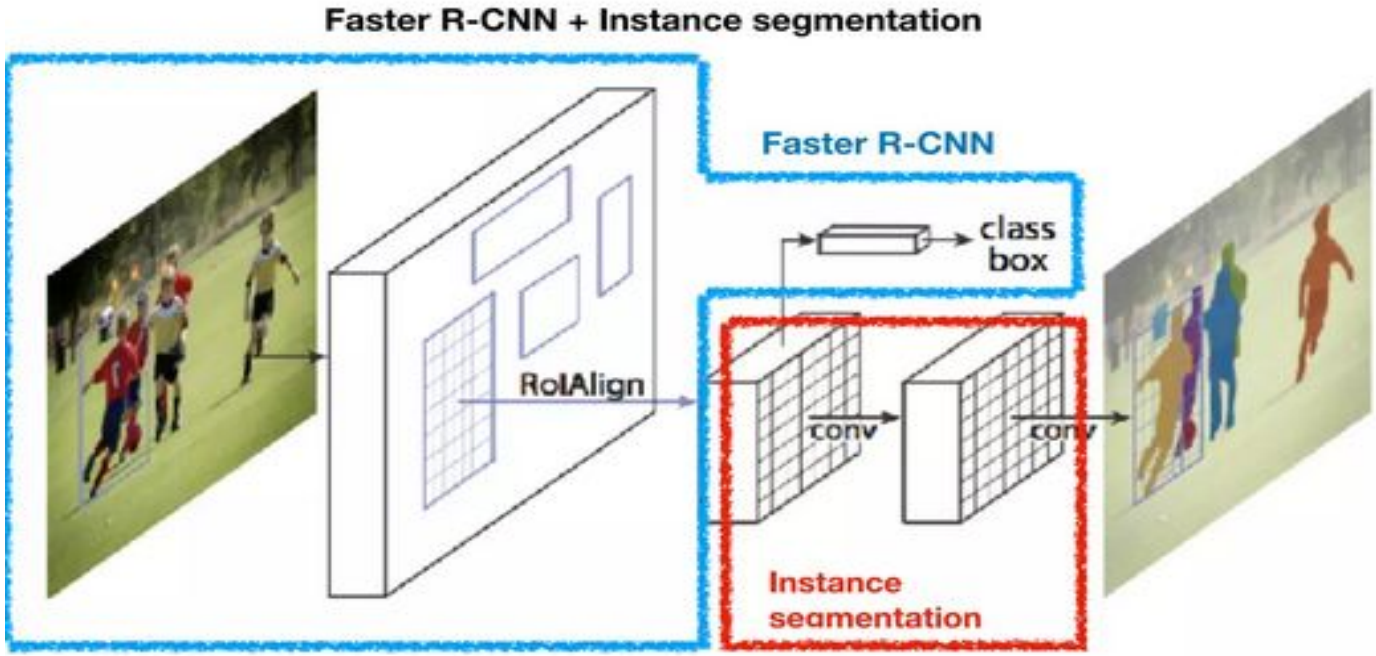


- Convolution backbone take image and results in the Feature map.
- RPN(Region proposal network) generates region proposals by scanning the feature map.
- RoIpool layer takes these region proposals, then divides them into a grid of fixed spatial bins and RoIPool layer performs max pooling within that region.
- Equal size feature map is fed into subsequent fully connected layers to perform object classification and bounding box regression.



Faster RCNN

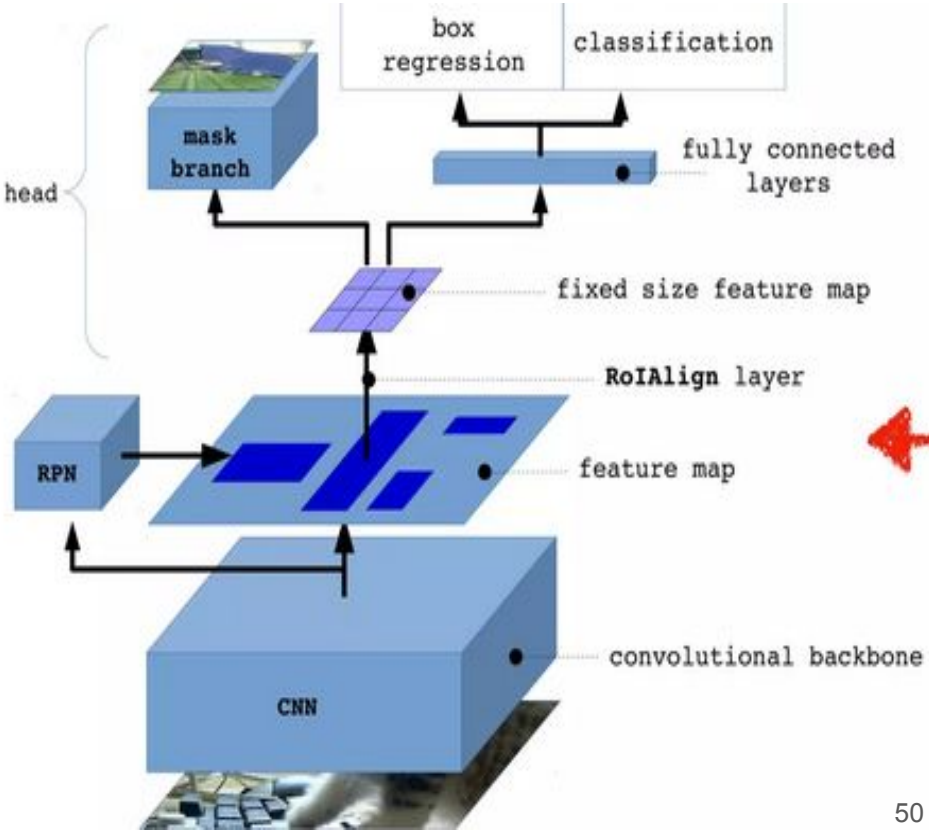
Mask R-CNN



- Builds on top of Faster R-CNN by adding a parallel branch of Fully convolutional Network which results in the masks.

Mask R-CNN

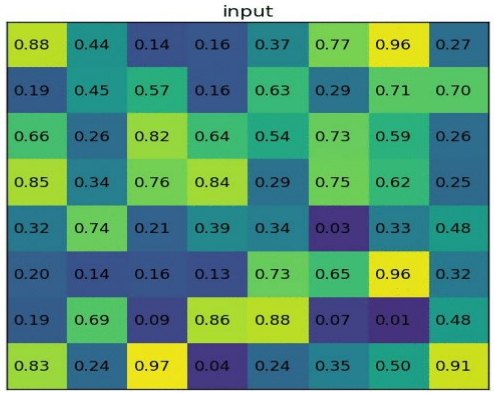
- Convolution backbone results in the feature map.
- RPN(Region proposal network) generates region proposals by scanning feature map.
- RoIpool layer takes these region proposals, then divides them into a grid of fixed spatial bins.
- RoIAlign layer uses bilinear interpolation to sample the exact locations on the feature map for each sub-region.
- Mask R-CNN outputs a binary mask for each RoI in parallel to predict the class and box offset.



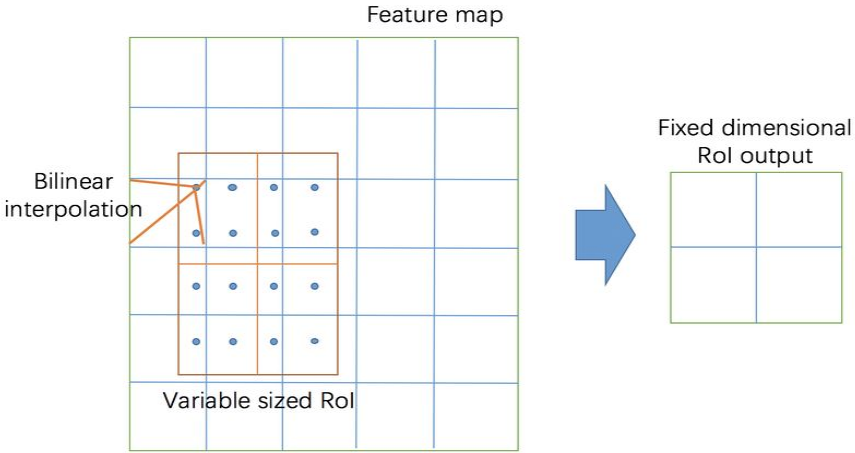
Mask R-CNN : RoIAlign

- RoIAlign (Region of Interest Align) was proposed as an improvement over RoIPool to address the misalignment problem.

- RoIAlign avoids this misalignment by using bilinear interpolation to sample the exact locations on the feature map for each sub-region.



RoIPool Layer



RoIAlign Layer

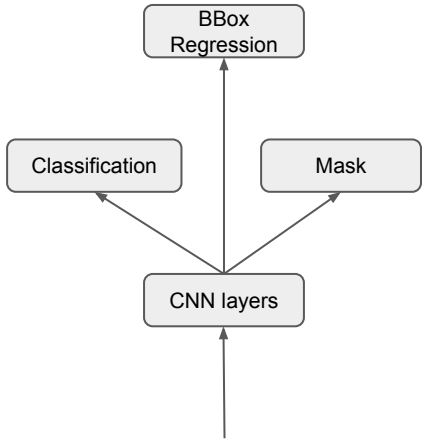
Mask R-CNN : Loss Function

Multitask loss:

- L_{cls}: negative log likelihood
- L_{box} : smooth $L1$ loss
- L_{mask}: mean binary cross-entropy

Loss function for each sampled RoI is:

$$L = L_{cls} + L_{box} + L_{mask}$$



Mask R-CNN

Mask R-CNN : Results

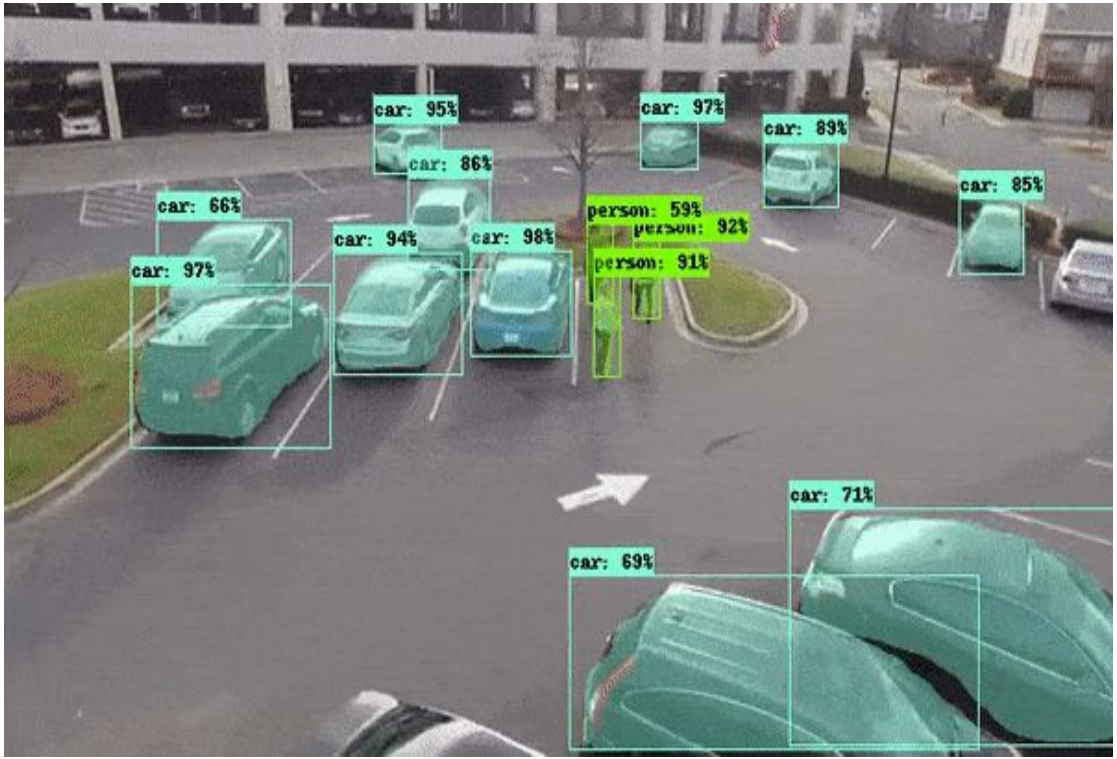
Experiments on Cityscapes

Mask R-CNN for Human Pose Estimation

Mask R-CNN : Some more models for Instance Segmentation

Many other instance segmentation models have been developed based on R-CNN such as:

- R-FCN
- MaskLab
- DeepMask
- PolarMask
- CenterMask

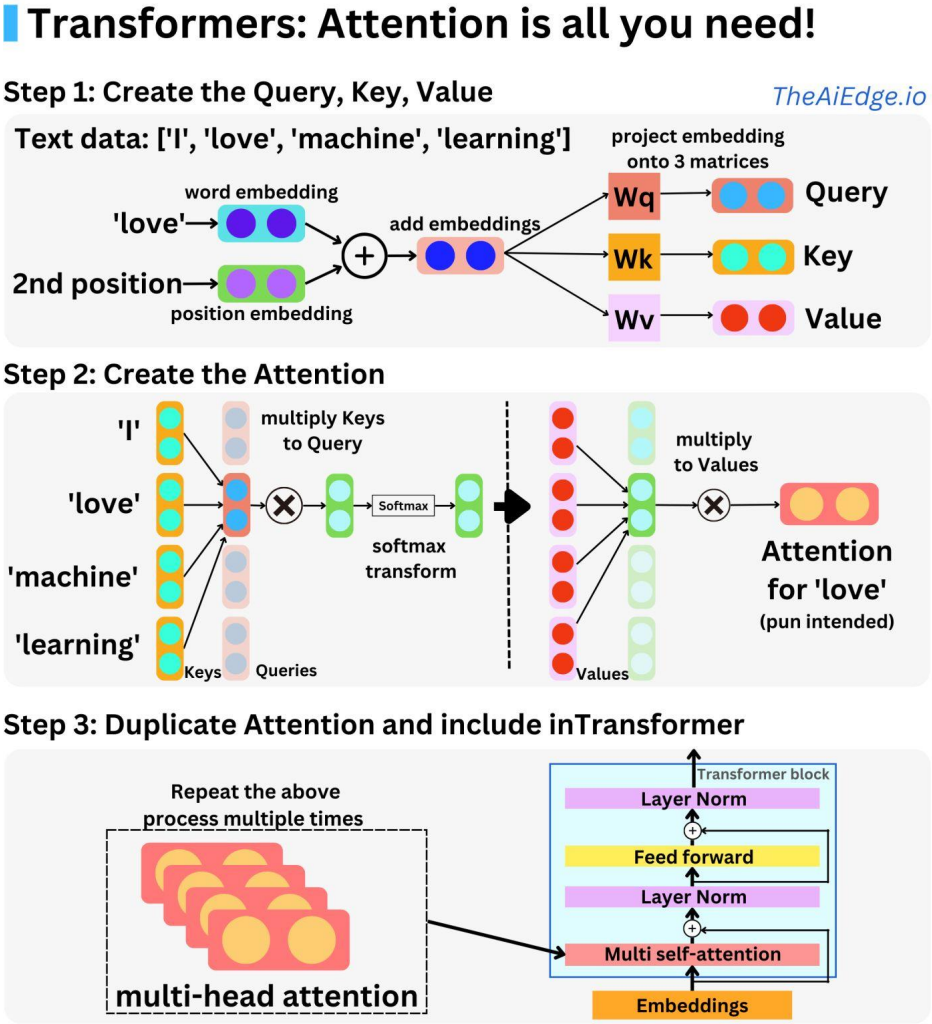


Transformer based

ViT : Vision Transformer

The Transformer Architecture

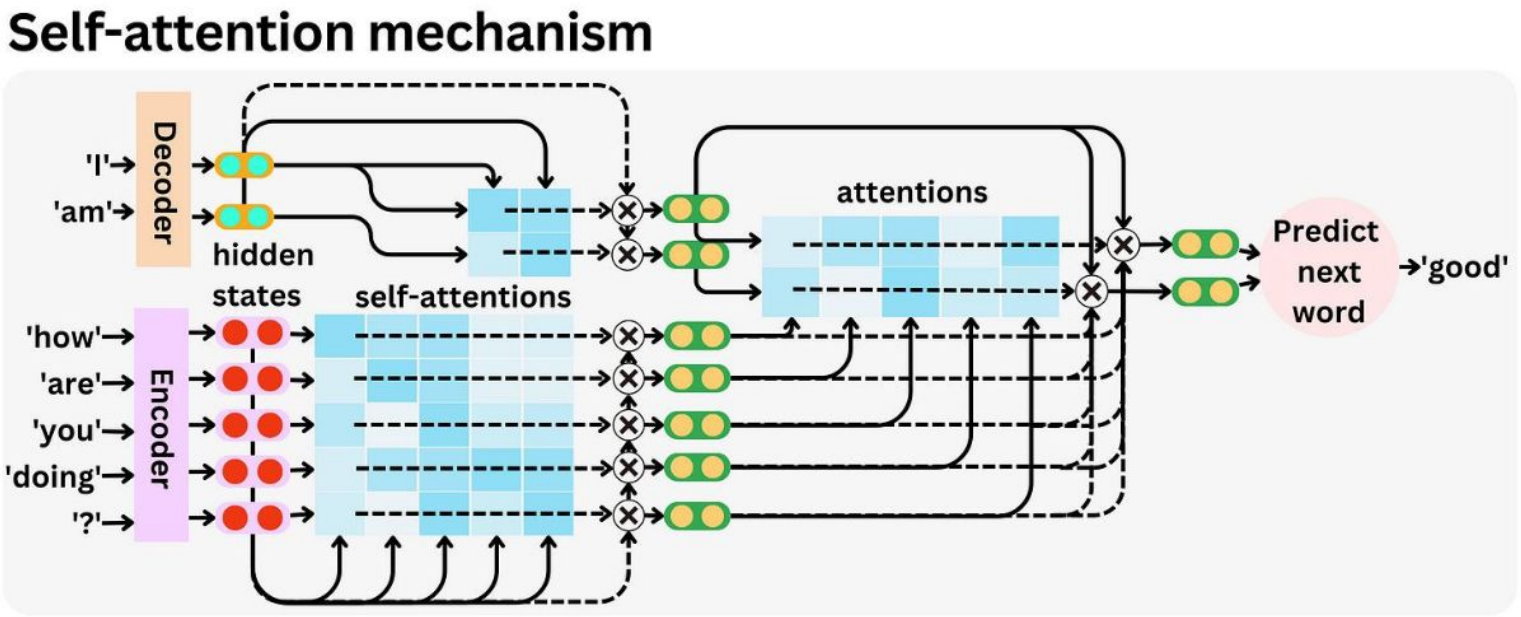
- It was introduced by Vaswani, Ashish, et al. in the paper **"Attention is all you need"** *Advances in neural information processing systems* 30 (2017).



Transformer based

ViT : Vision Transformer

The Self-attention mechanism



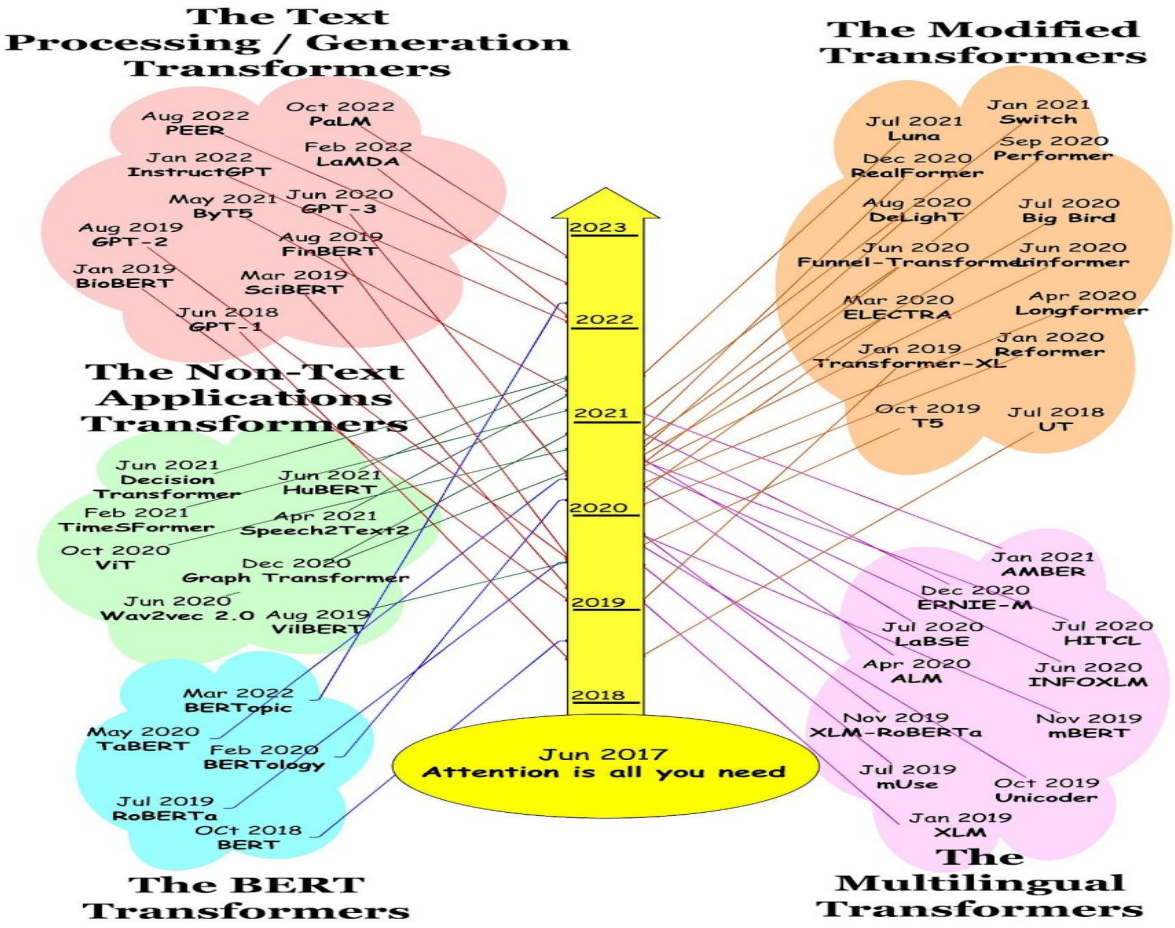
Transformer based

ViT : Vision Transformer

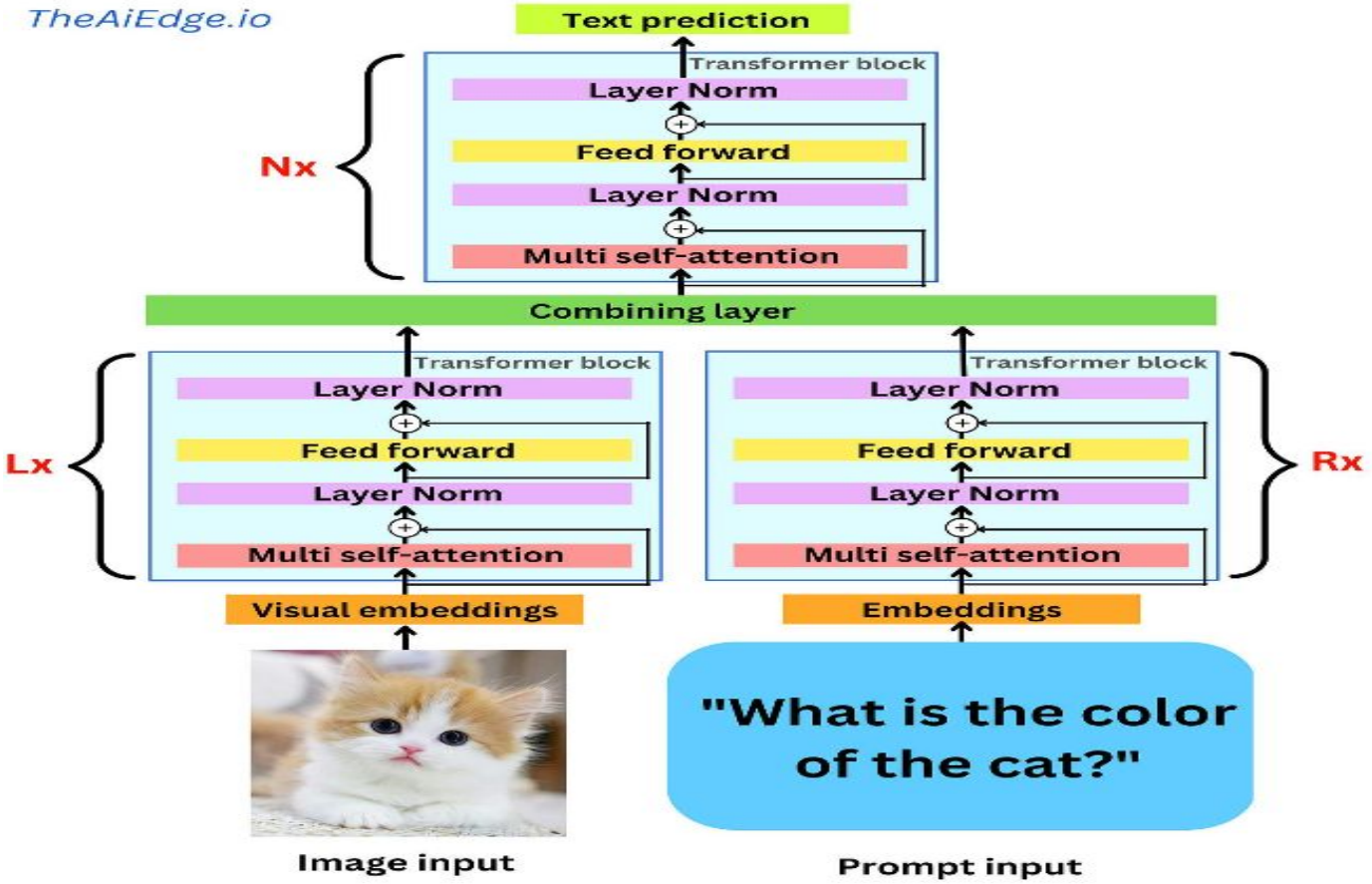
ViT Introduction:

- In natural language processing (NLP) the prevalent architecture today is the Transformer.
- But the modeling in computer vision has long been dominated by convolutional neural networks (CNNs).
- The Vision Transformer (ViT) model was proposed in **An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale**.
- It's the first paper that successfully trains a Transformer encoder on ImageNet, attaining very good results compared to familiar convolutional architectures.

Transformers : History Timeline



GPT 4 : Largest Language Model

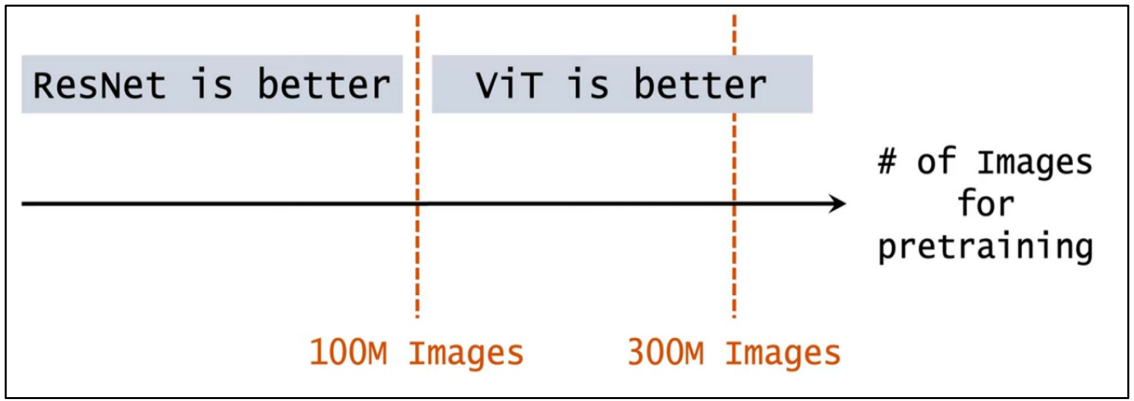


Transformer based

ViT : Vision Transformer

Motivation of using ViT in CV:

- In image classification the best architecture *were* CNNs (ResNet).
- ViT *beats* CNN's when the dataset for pretraining is sufficiently large. (at least 100 million images).
- ViT is the successful application of Transformers in CV.

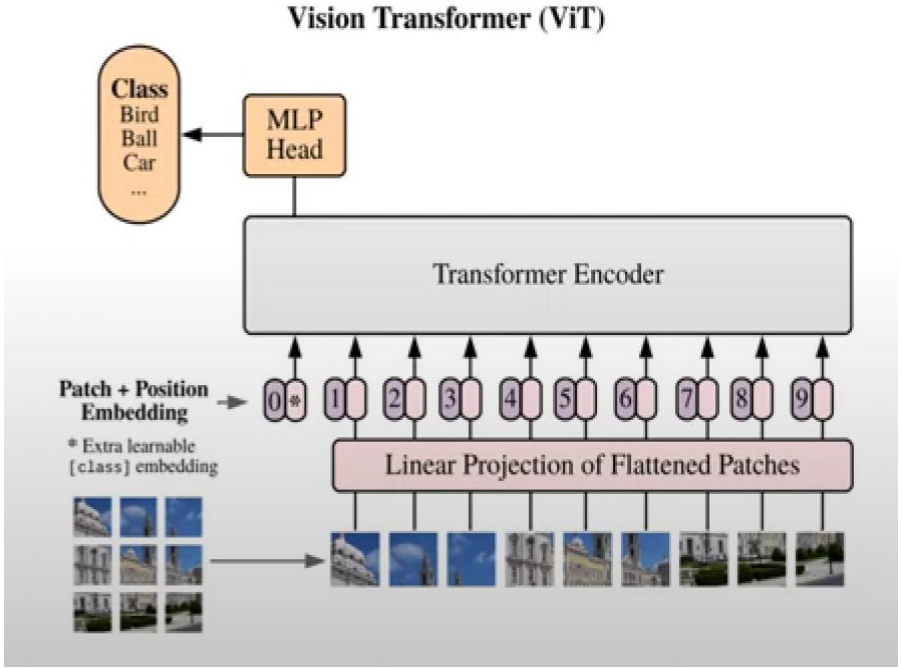


Transformer based

ViT : Vision Transformer

ViT Model overview (for the classification task):

- First, split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder.
- In order to perform classification, use the standard approach of adding an extra learnable “classification token” to the sequence.

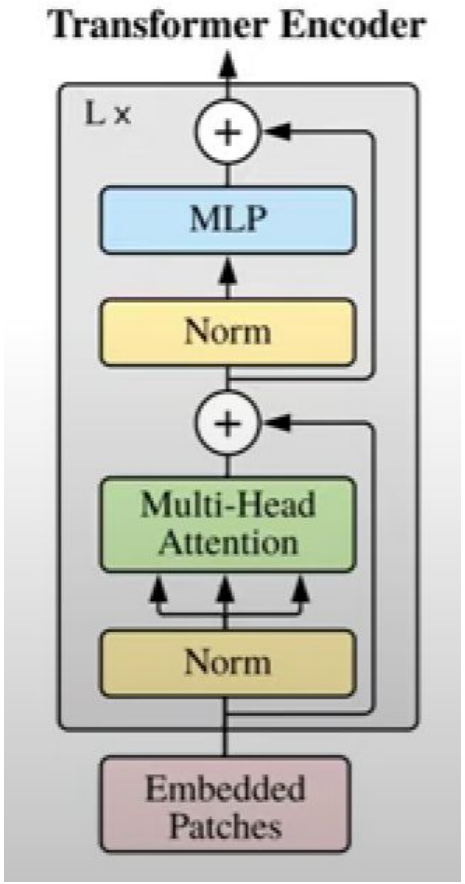


Transformer based

ViT : Vision Transformer

ViT Model overview (for the classification task):

- The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).
- A Transformer Encoder Network contains:
  - Multi-head Self-Attention Layer (MSA)
  - + Dense Layer (MLP)
  - + Skip-Connections
  - + Layer Norm (LN)
- It consists of alternating layers of multi headed self-attention (MSA) and multi-layer perceptron (MLP) blocks. Layernorm (LN) is applied before every block, and residual connections after every block.

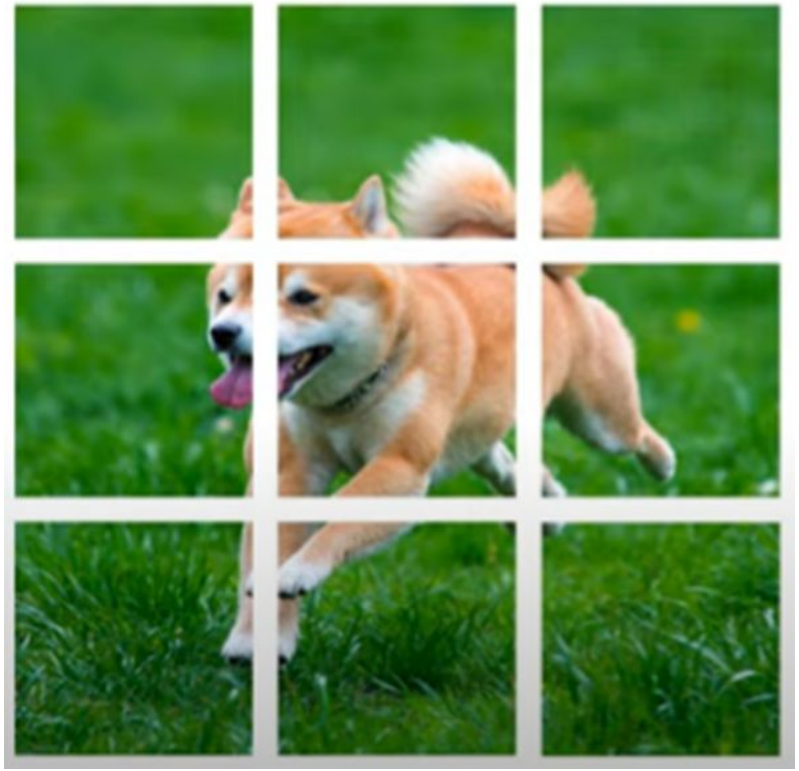


Transformer based

ViT : Vision Transformer

Training ViT as a Classifier:

- Split the image into patches.
- Patches may or may not overlap.
- Which Patch-size to use?
  - The paper uses a patch size of 16 x 16.
- What Stride to use?
  - The paper uses a stride of 16 x 16 (horizontal stride x vertical stride).

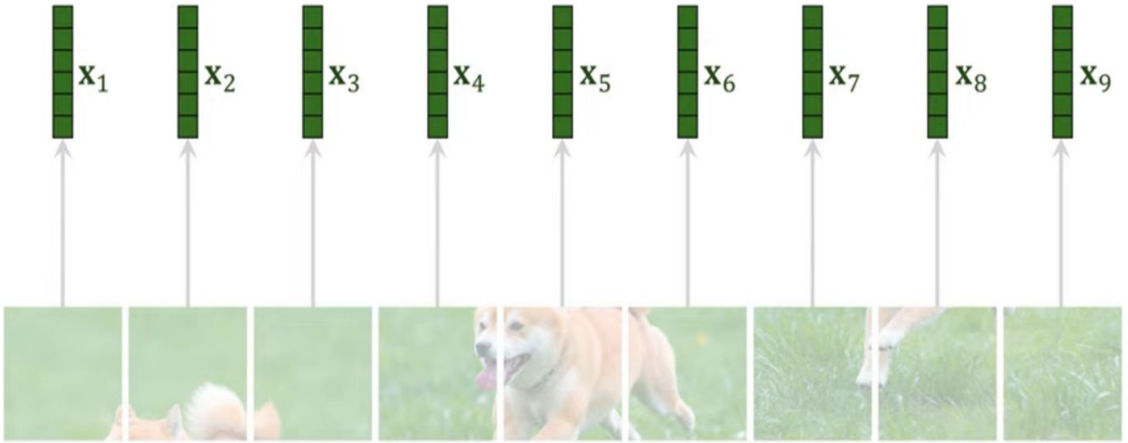


Transformer based

ViT : Vision Transformer

Training ViT as a Classifier:

- The next step is to flatten the patches i.e. to convert patches (h, w, c) to vectors (hwc, 1).

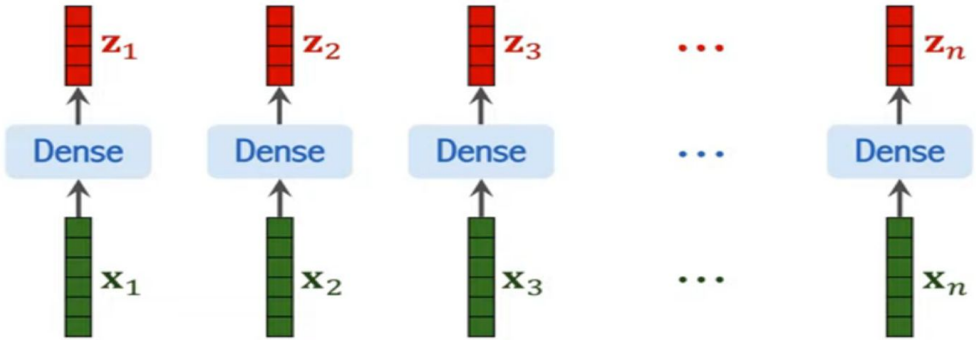


Transformer based

ViT : Vision Transformer

Training ViT as a Classifier:

- Now, pass the reshaped vectors (x1, x2,…xn) from the dense layer to get embeddings. In other words, the first layer of the Vision Transformer linearly projects the flattened patches into a lower-dimensional space.
- Positional encodings are then added to z1, z2, z3,…zn.



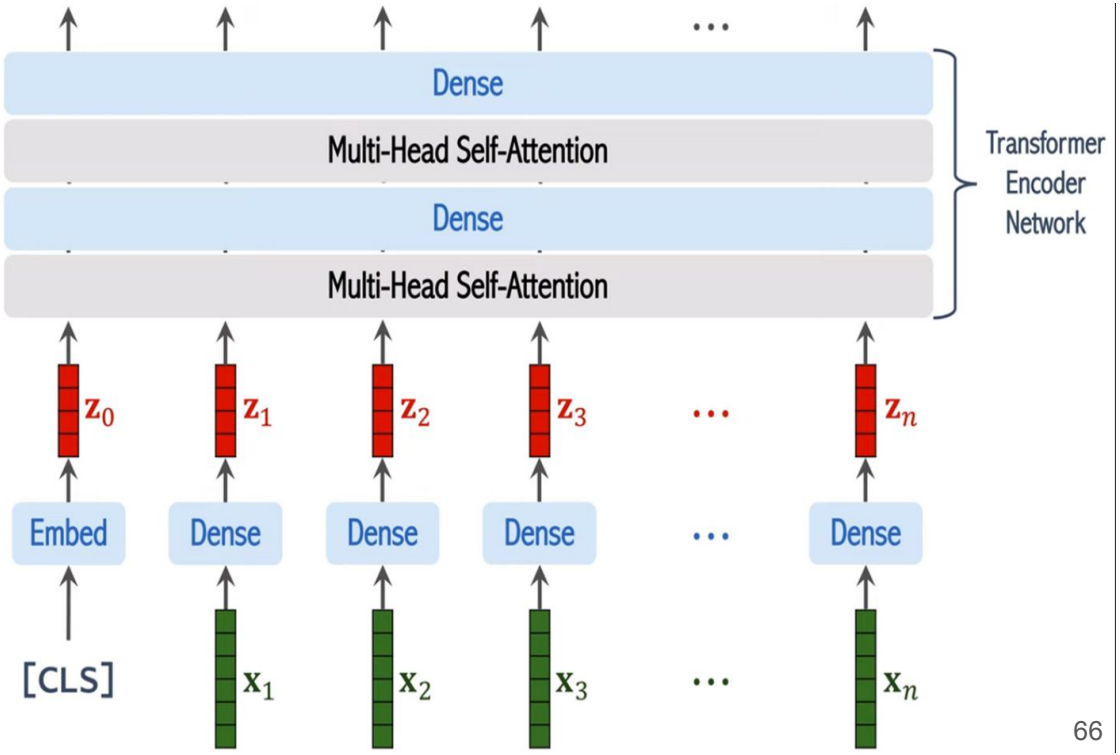
Transformer based

ViT : Vision Transformer

Training ViT as a Classifier:

The CLS token:

- The [CLS] token is a randomly initialized vector which is prepended to the sequence of embedded patches.
- It's embeddings are learned.
- The state of the [CLS] token at the output of the Transformer encoder serves as the image representation.

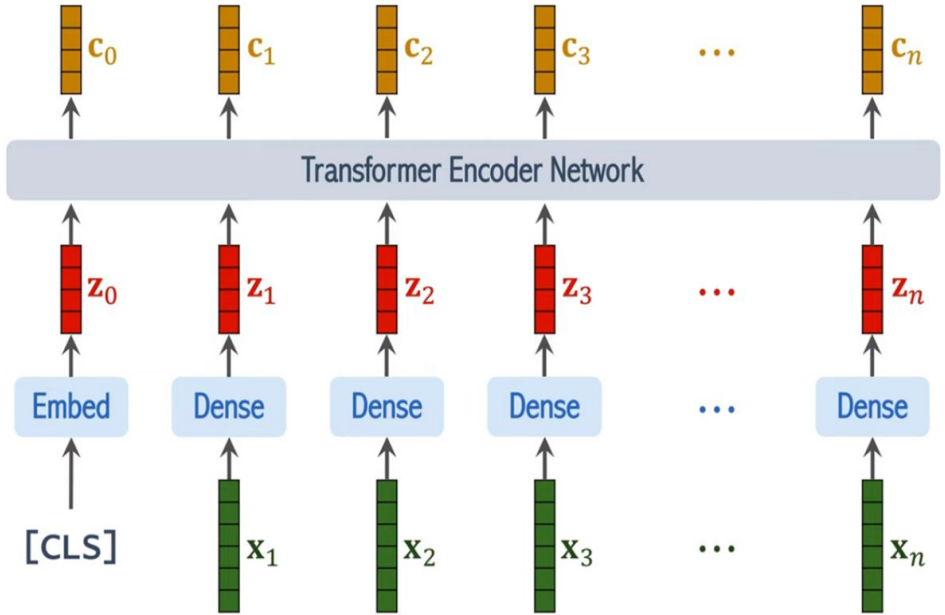


Transformer based

ViT : Vision Transformer

Training ViT as a Classifier:

- C0, C1, …, Cn vectors are output of transformer network.

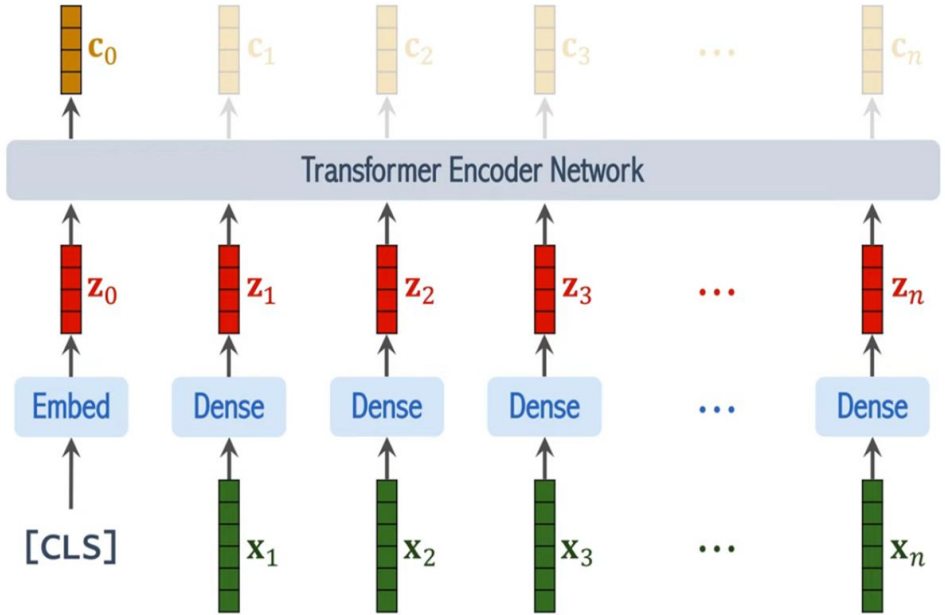


Transformer based

ViT : Vision Transformer

Training ViT as a Classifier:

- The classification is based on C0 only.
- Both during pre-training and fine-tuning, a classification head is attached to C0. The classification head is implemented by a MLP with one hidden layer at pre-training time and by a single linear layer at fine-tuning time.

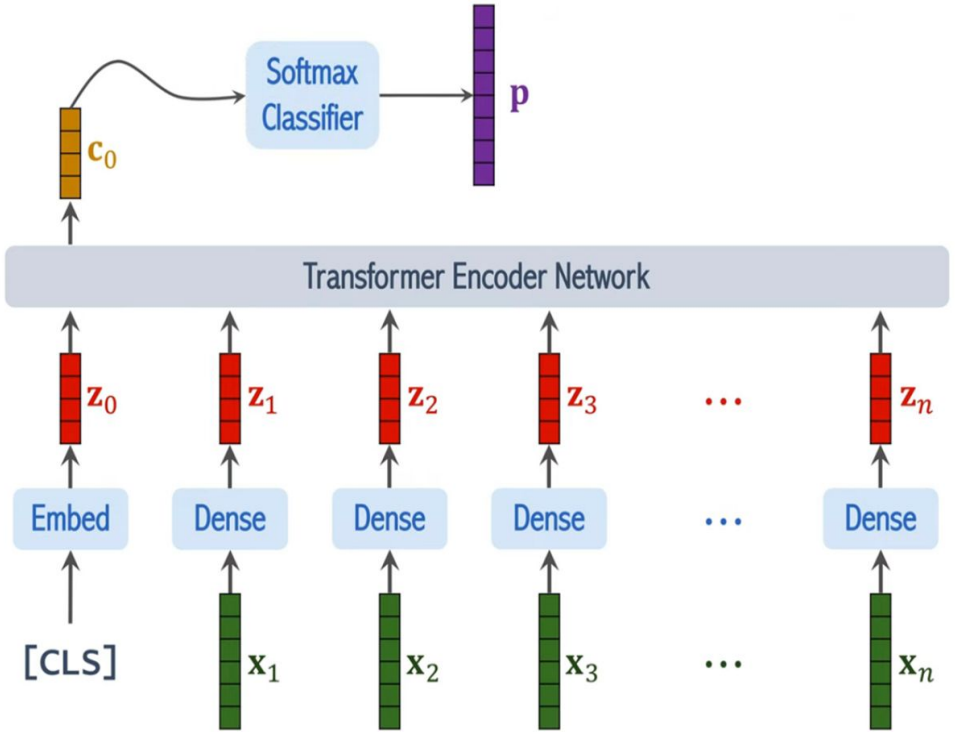


Transformer based

ViT : Vision Transformer

Training ViT as a Classifier:

- C0 vector when passed through softmax classifier gives the vector p having **dim = number of classes**.

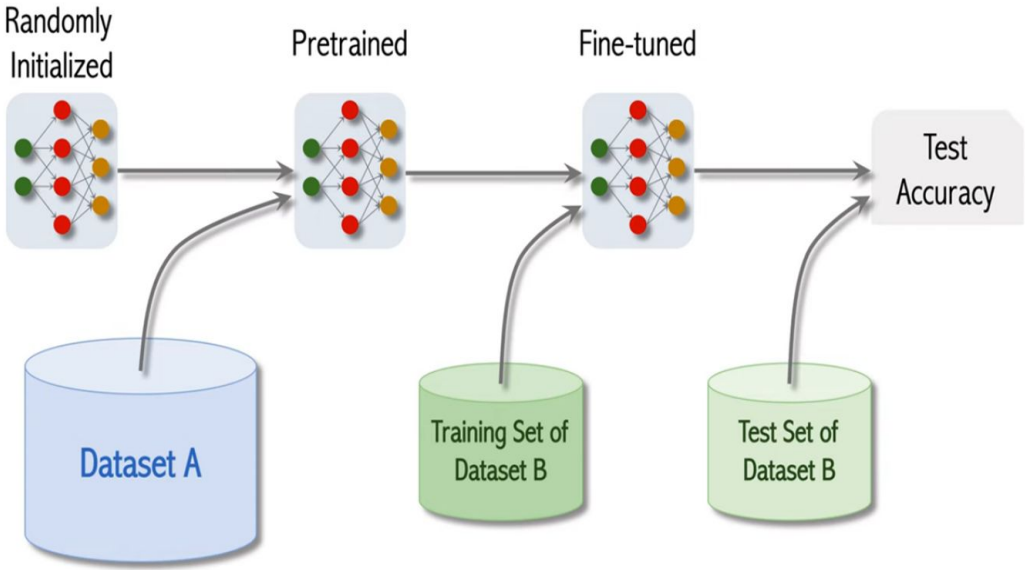


Transformer based

ViT : Vision Transformer

Training ViT as a Classifier (Complete Training):

- Randomly initialize the model.
- Train the model on some large image dataset A. This will give a pretrained model.
- Fine-tune the model on the training dataset.
- Evaluate the dataset on the test-set of dataset B.



Transformer based

ViT : Vision Transformer

Architectural Patterns in Transformer-based Segmentation

- **Vision transformer (ViT)** or its improved variants are the most common backbone networks.
- The improved variants include the following:
  - **Swin transformer:** Its shifted windows concept makes it similar to convolutions, allowing transformer blocks to form a hierarchical pyramid.
  - **Hybrid networks with convolutional blocks**: While transformers can take in more context, they are computationally less efficient due to the quadratic complexity of self-attention. Many variants add convolutional capabilities to make transformers more efficient. They include the convolutional vision transformer, co-scale, and LeViT.
  - **Improved spatial self-attention**: The twins models improve both accuracy and efficiency by optimizing spatial self-attention aspects.

Transformer based

ViT in Image Segmentation

Some of the ViT models specialized for the task of Segmentation:

- SEgmentation TRansformer (SETR)
- **Swin Transformer**
- Segmenter
- Segformer
- Pyramid Vision Transformer (PVT)
- Twins
- Dense Prediction Transformer (DPT)
- High Resolution Transformer (HR Former)
- Masked-Attention Mask Transformer (Mask2Former)

Transformer based

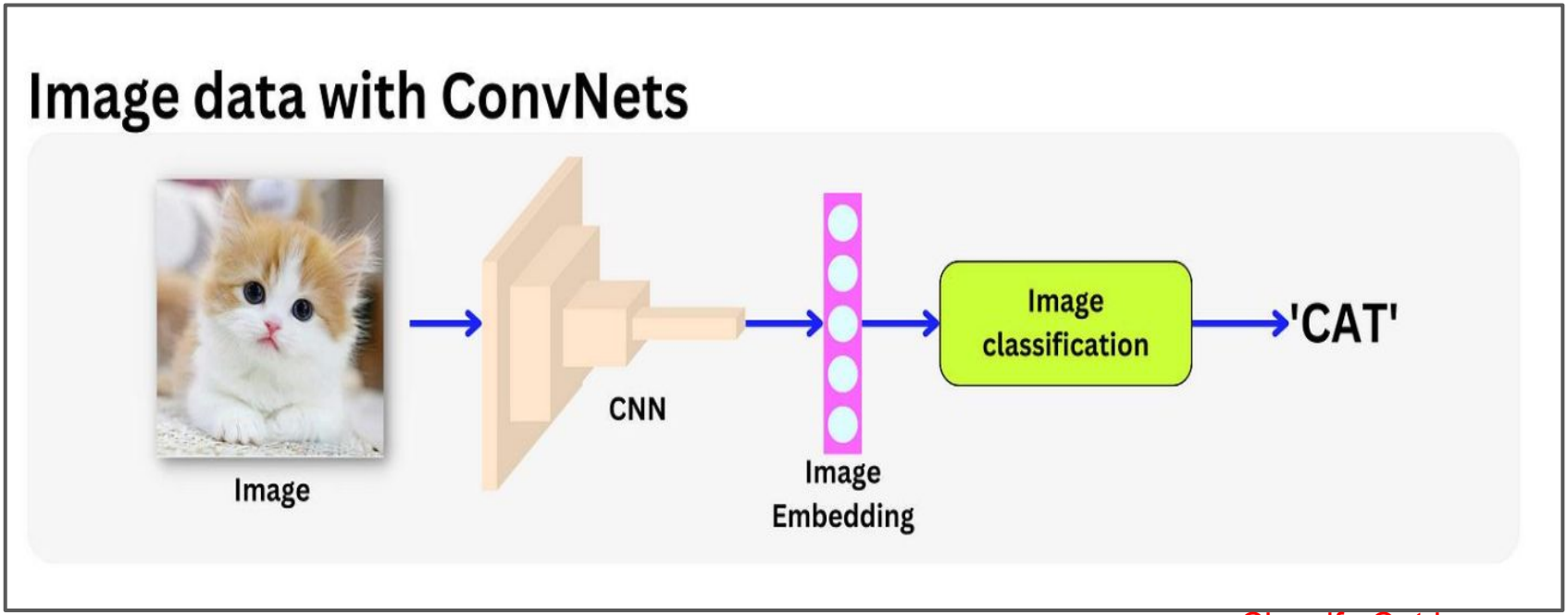
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows (ICCV 2021)

Motivation:

1. The paper seek to expand the applicability of Transformer such that it can serve as a general-purpose backbone for computer vision, as it does for NLP and as CNNs do in vision.
2. The authors observed significant challenges in transferring Transformers' high performance in the language domain to the visual domain.
3. The challenges can be explained by the **differences between the two modalities.**

Transformer based

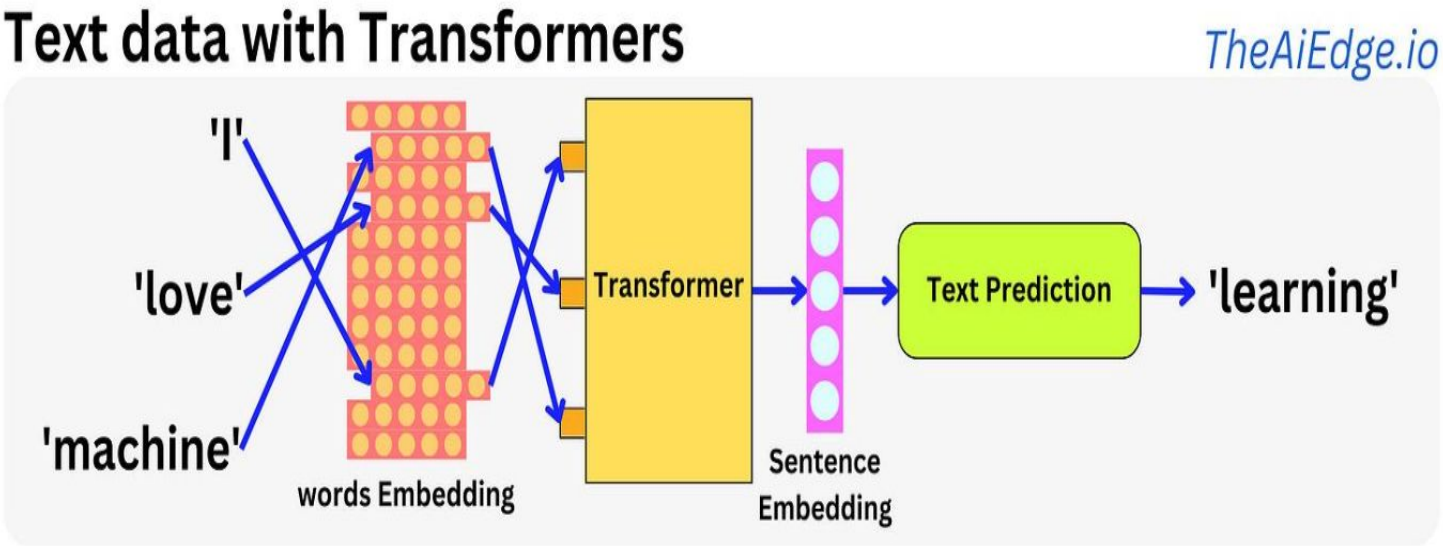
Swin Transformer: Image data vs Text data



Input (Cat image) → CNN (learn local & global features) → Flatten the last layer features / Image Embedding(dim=1024/512) → Classify Cat image as **'CAT'**

Transformer based

Swin Transformer: Image data vs Text data



Input (Words) → Words Embedding → Transformer: extract semantic Representations from Sentences → Text Prediction: Predict the next word of Sentence i.e **'learning'**

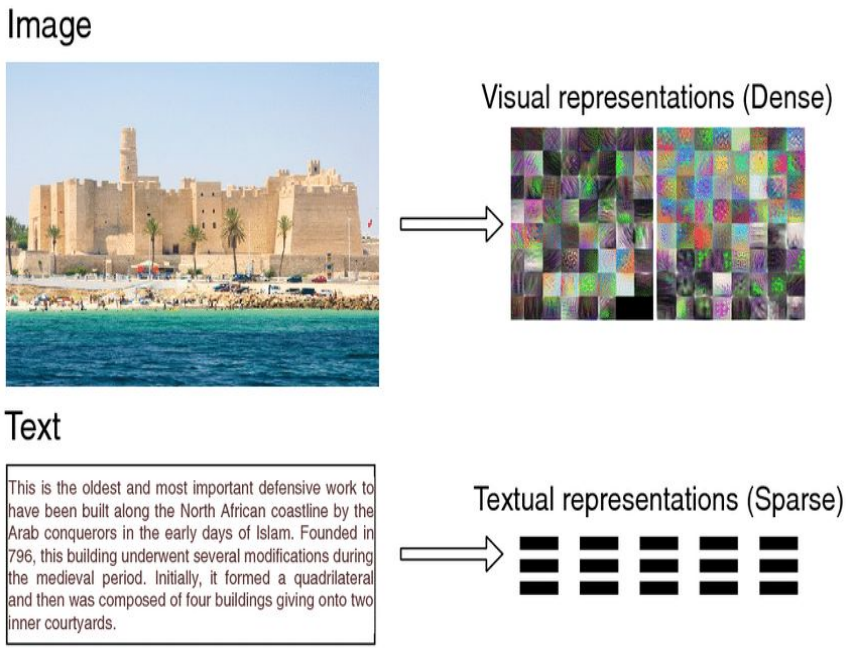
0.8	−0.1	0.8	−0.9	0.8	−0.5	−0.9

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Key Difference between visual and textual signals:

- Basic Elements
  - In Language Models the basic element is a word token.
  - In Computer Vision the basic element for processing may vary from a pixel to a portion of the image sometimes called a patch.
  - Also, An image representation is denser as compared to a textual representation.



Swin transformer takes care of the difference between a textual data and the image data.

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Key Difference between visual and textual signals:

- Translation Invariance
  - A visual object is insensitive to its location.
  - A word is sensitive to its location.

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Key Difference between visual and textual signals:

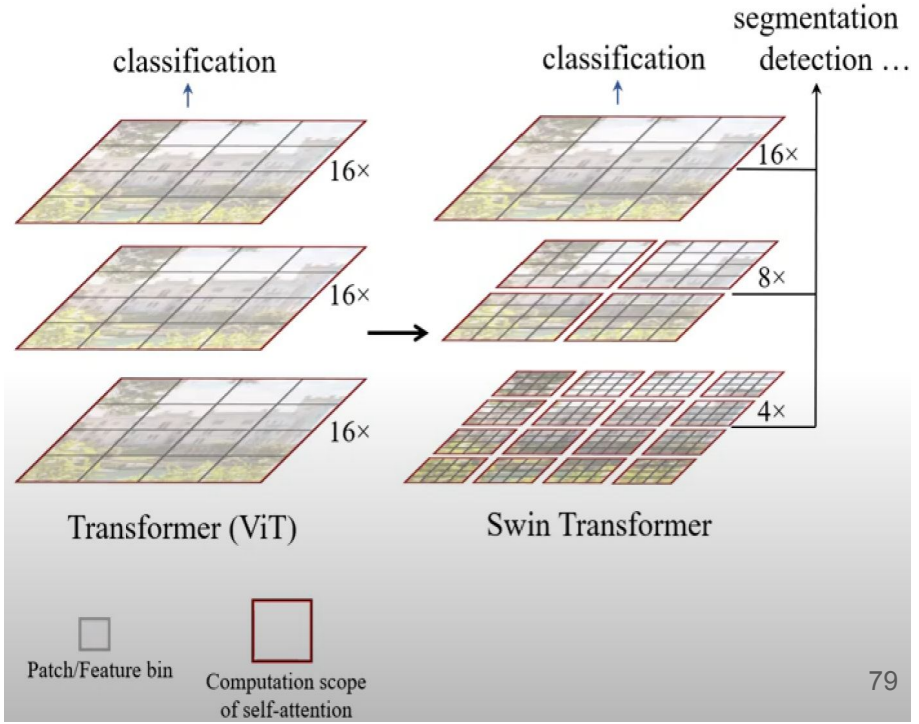
- Spatial Locality
  - Spatial Locality means that all those pixels which are stored nearby to a particular pixel have some correlation.
  - The correlation between the neighboring pixels is more prevalent in images than in the words of a sentence.

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Architectural Details:

- The Figure shows the portion of an image processed at various layers in ViT (left) and in Swin (right).
- **Swin Transformer builds hierarchical feature maps** by merging **image patches (shown in gray)** in deeper layers.
- It computes self-attention only within each **local window (shown in red)**. It can thus serve as a **general-purpose backbone** for both image classification and dense recognition tasks.
- In contrast, **ViT produce feature maps of a single low resolution**. The computation of self-attention is done globally.



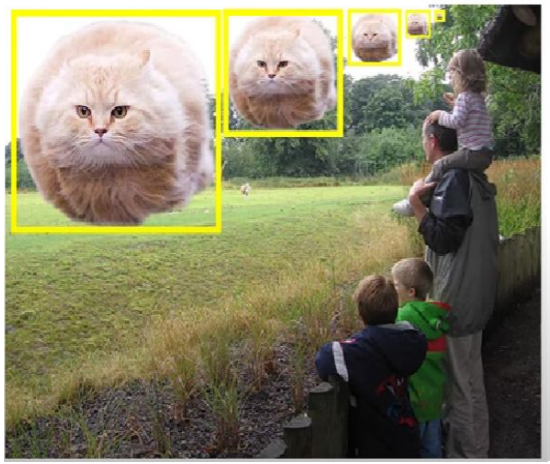
Transformer based

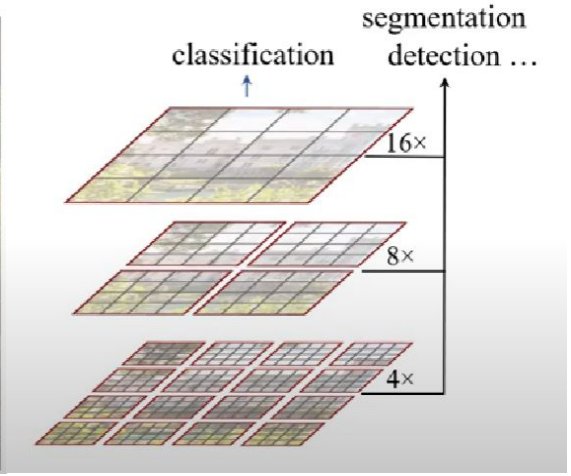
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Architectural Details:

WHY Swin Transformer builds hierarchical feature maps?

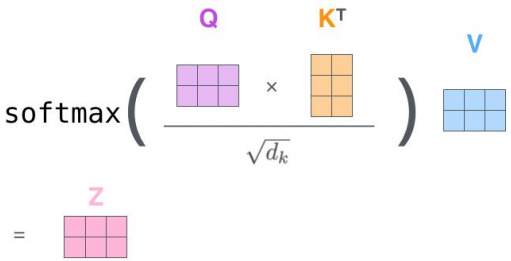
- In existing Transformer-based models, tokens are all of a fixed scale, a property **unsuitable** for the vision applications.
- In Images **No matter how large a fat cat is… it's still a fat cat.**
- Swin processes objects of different scales. This goal is achieved by a hierarchical structure.





Transformer based

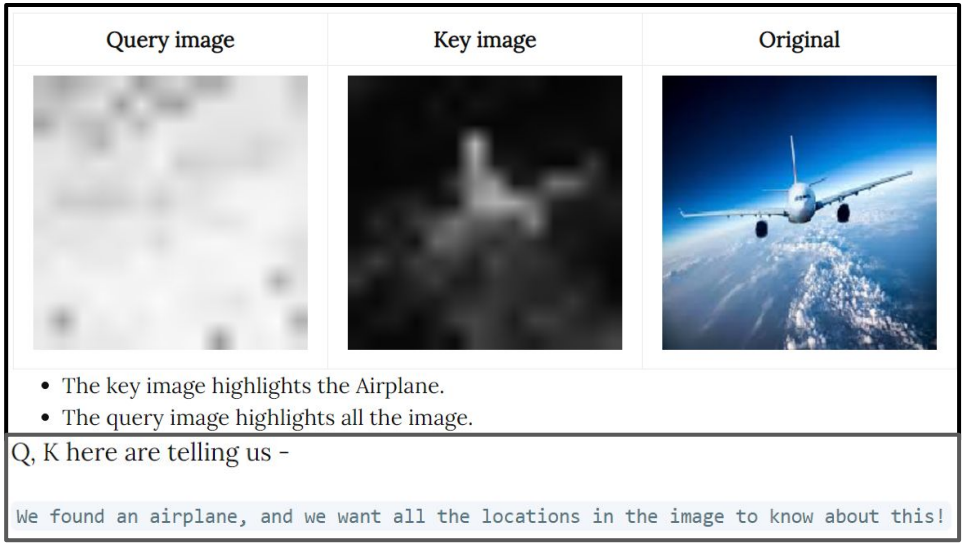
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows



Attention Mechanism in ViT/Swin:

Q, K, V and Attention:

- The unit of attention is a token (image patch).
- The **attention weights** of a Transformer block are computed between the key (K) and the query (Q).
- The weights quantifies how important is the key to the query. **In the ViTs the key and the query comes from the same image, hence the weights determine which part of the image is important.**
- Visualization is done through computing a 0–1 ligthness (i.e. attention weight) at each token.



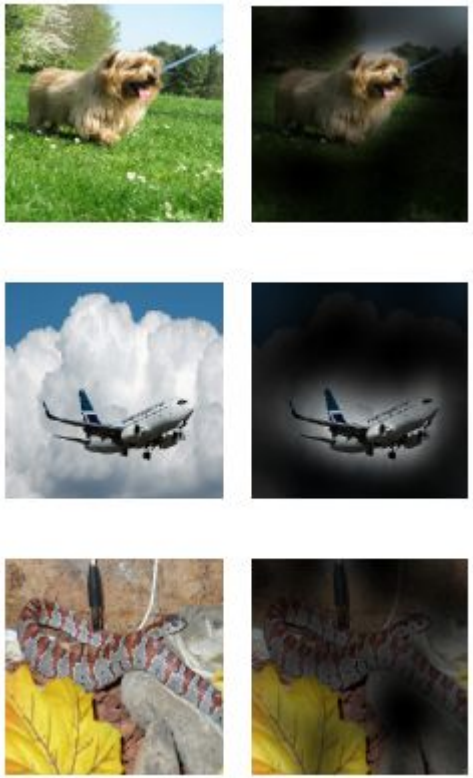
Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Attention Mechanism in ViT/Swin:

Q, K, V and Attention:

- Attention scores are then multiplied by the original image to get the **Attention maps**.
- The **attention map highlights the important region** in the image for the target class.



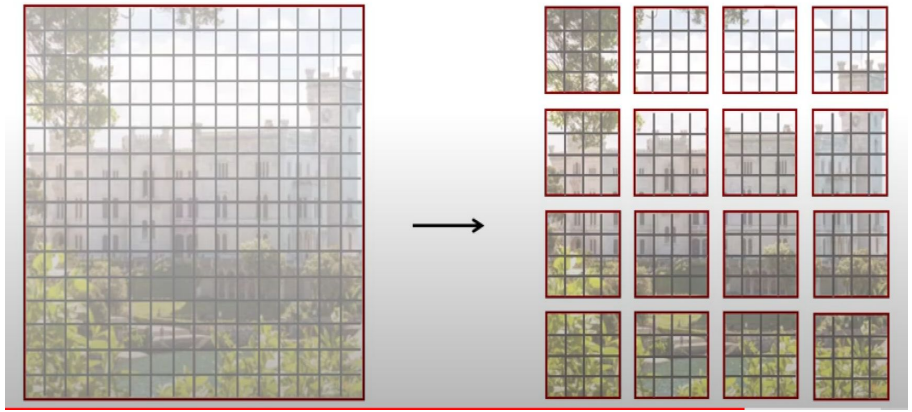
Original Image Attention Maps

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Architectural Details:

- **Swin transformer has linear computation complexity** to input image size due to computation of self-attention only within each local window (shown in red).
- **ViT have quadratic computation complexity** to input image size due to computation of self-attention globally.



Attention score calculation in ViT = 256^2
= 65536 (globally)

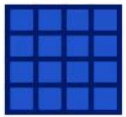
Attention score calculation in Swin = $16 * 16^2$
= 4096
where n_1 is the number of patches

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Standard MSA

Attention for each patch is computed against all patches, resulting in quadratic complexity



Architectural Details:

- **ViT have quadratic computation complexity** to input image size due to computation of self-attention globally.

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Window-based MSA

Attention for each patch is only computed within its own window (drawn in red).
Window size is 2x2 in this example.

Architectural Details:

- **Swin transformer has linear computation complexity** to input image size due to computation of self-attention only within each local window.

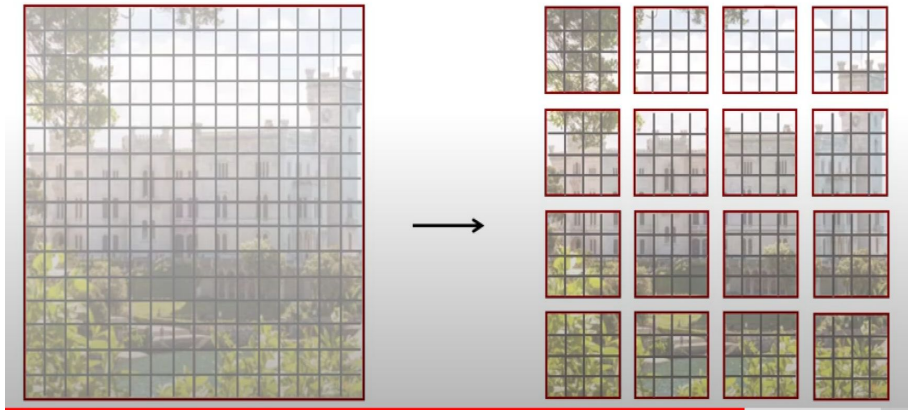
Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Architectural Details:

WHY Swin Transformer needs to reduce the computational complexity?

- It is because of the much higher resolution of pixels in images compared to words in passages of text.
- There exist many vision tasks such as **semantic segmentation that require dense prediction at the pixel level**, and this would be intractable for Transformer on high-resolution images, as the computational complexity of its self-attention is quadratic to image size.



Attention score calculation in ViT = 256^2
= 65536

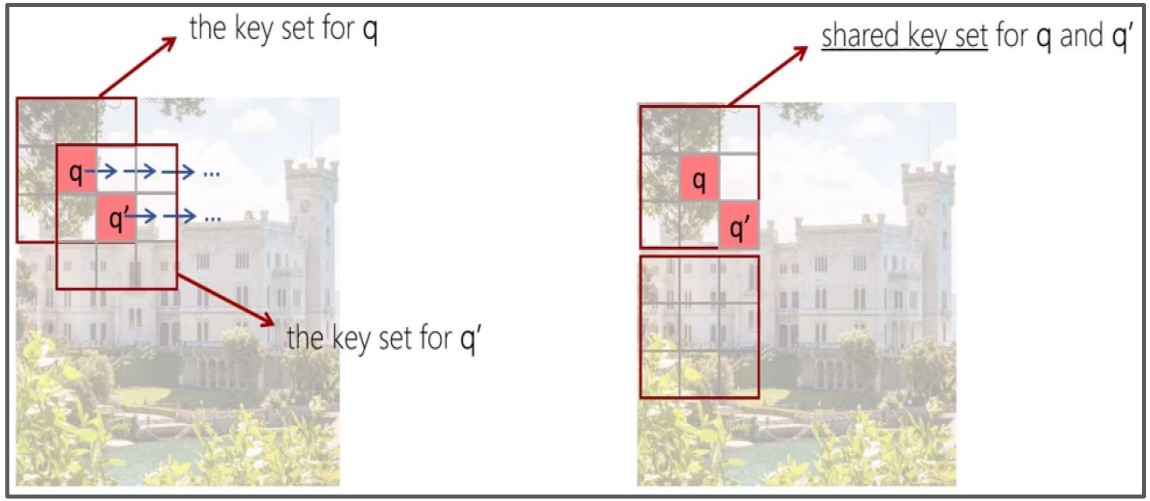
Attention score calculation in Swin = $16 * 16^2$
= 4096
where n_1 is the number of patches

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Architectural Details:

- **Swin Transformer uses non-overlapping windows** as opposed to the traditional sliding window approach.
- **All query patches within a window share the same key set, which facilitates memory access in hardware.**
- Shared keyset enables friendly memory access and is thus good for speed (larger than 3x).



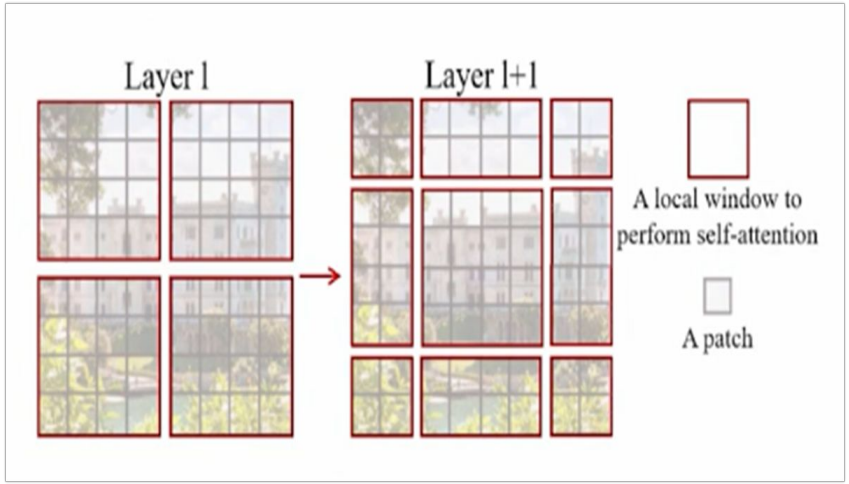
Sliding window approach (left) **Non-overlapping window approach (right)**

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Architectural Details:

- A key design element of Swin Transformer is its **shift of the window partition** between consecutive self-attention layers.
- In layer l (left), a regular window partitioning scheme is adopted which **starts from the top-left pixel**, and the 8 × 8 feature map is evenly partitioned into 2 × 2 windows of size 4 × 4 (M = 4). The self-attention is computed within each window. M is the number of patches within each window.
- In the next layer l + 1 (right), the window partitioning is shifted, resulting in new windows. **It adopts a windowing configuration that is shifted from that of the preceding layer, by displacing the windows by (⌊M/2⌋, ⌊M/2⌋) pixels (forward and backward) from the regularly partitioned windows.**
- The self-attention computation in the new windows crosses the boundaries of the previous windows in layer l, providing connections among them.

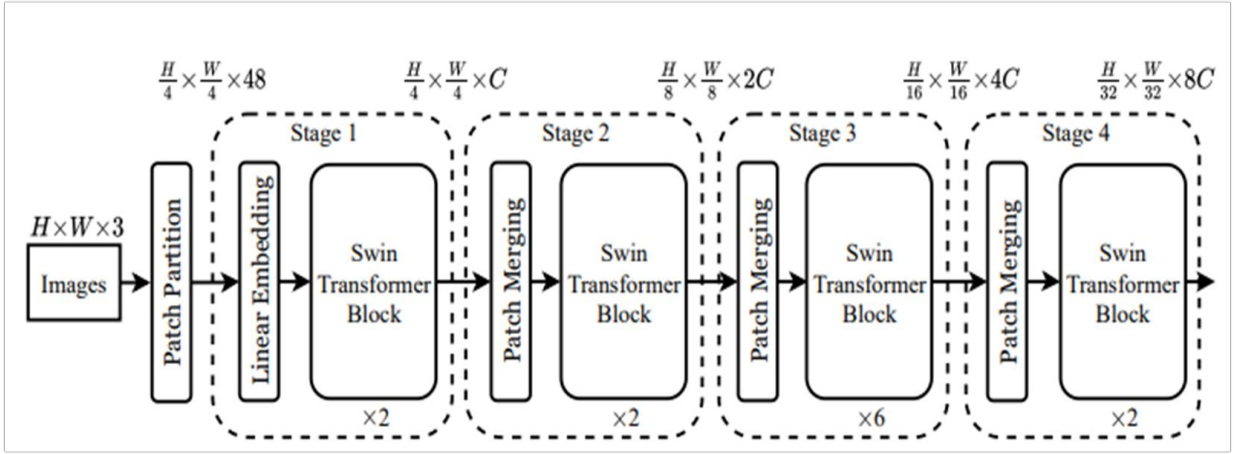


Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Input:

- Swin Transformer first splits an input RGB image into non-overlapping patches by a patch partition module, (like ViT**).**
- Each patch is treated as a “token” and its feature is set as a concatenation of the raw pixel RGB values.
- **A patch size of 4×4** is used and thus the feature dimension of **each patch is 4×4×3=48**.
- A linear embedding layer is applied on this raw-valued feature to **project it to an arbitrary dimension *C***.
- C is the channel number of the hidden layers in the first stage.



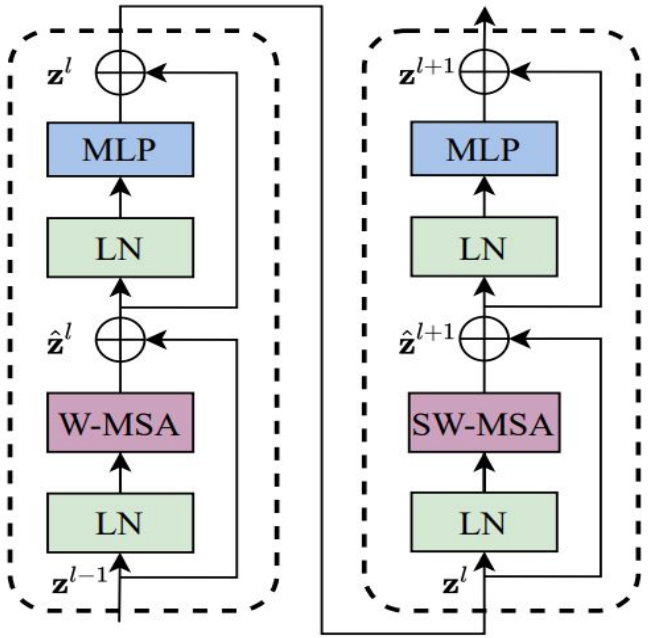
Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Swin Transformer Block:

- A Swin Transformer block consists of a **shifted window based MSA module**, followed by a 2-layer **MLP** with **GELU** nonlinearity in between.

- A **LayerNorm (LN)** layer is applied before each MSA module and each MLP, and **a residual connection** is applied after each module.



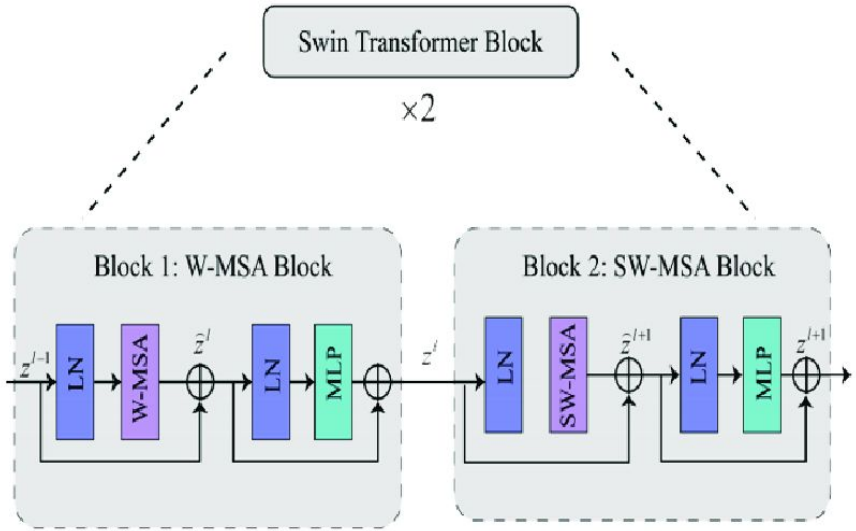
Two successive Swin transformer blocks.

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Swin Transformer Block (another view):

- Different versions of Swin Transformer differ in the number of Swin Transformer block layers, the size of hidden dimensions, the number of attention heads used by the W-MSA and SW-MSA layers, and the size of the MLP classifier.
- The "Swin-T" model (discussed here) has 12 Swin Transformer block layers, with hidden size 768, and uses 24 attention.



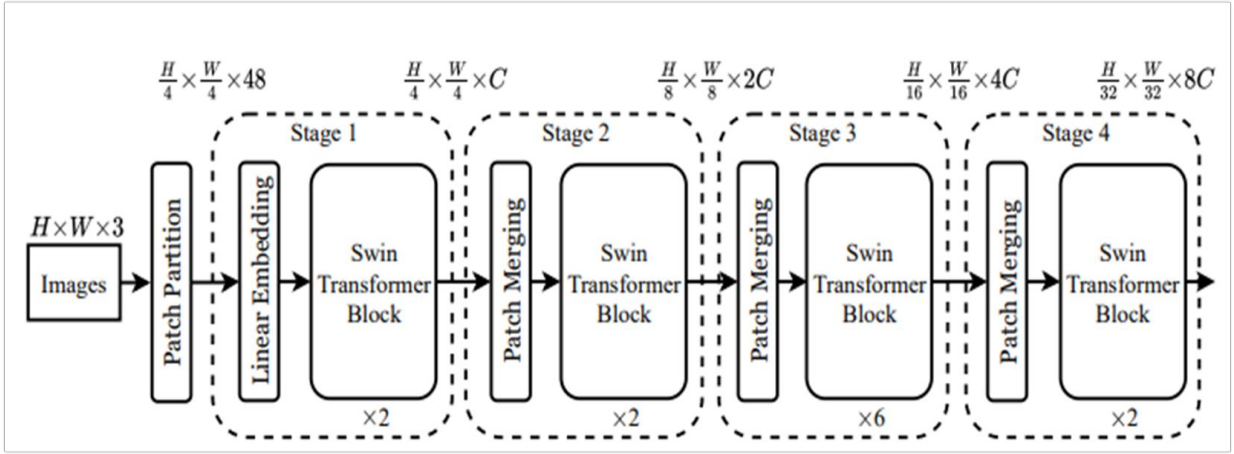
Two successive Swin transformer blocks.

Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Stage 1:

- In stage-1 two **Swin Transformer blocks are applied on the patch tokens.**
- **A patch size of 4×4** is used and thus the feature dimension of **each patch is 4×4×3=48**.
- A linear embedding layer is applied on this raw-valued feature to **project it to an arbitrary dimension *C***.
- In other words, C is the channel number of the hidden layers in the first stage. (C=96 in Swin-T)
- The Transformer blocks maintain the number of tokens ($H/4 \times W/4$), and together with the linear embedding are referred to as **"Stage 1"**.

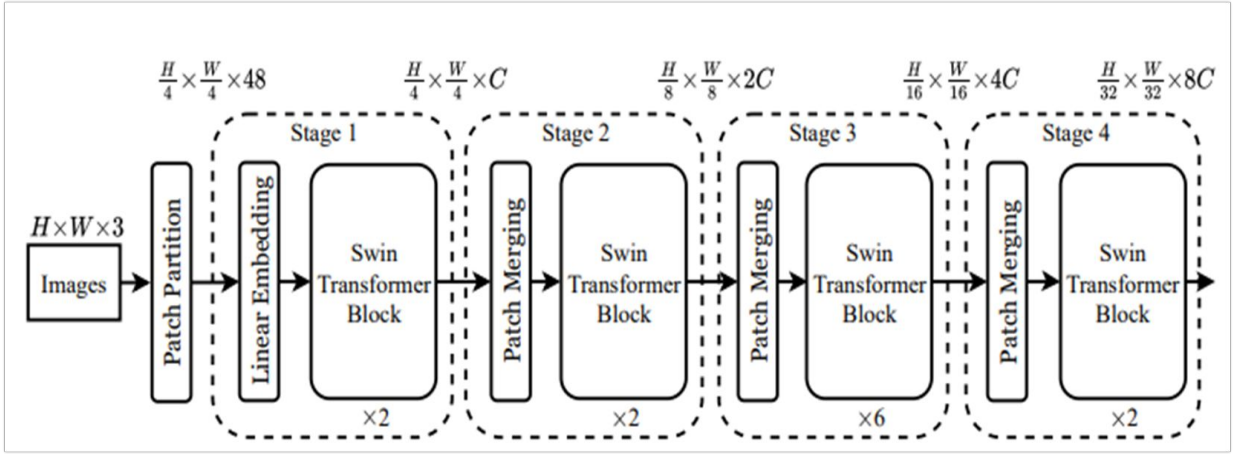


Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Stage 2:

- To produce a **hierarchical representation**, **the number of tokens is reduced by patch merging layers** as the network gets deeper.
- The **output dimension** is set to **2*C***, and the resolution is kept at ***H*/8×*W*/8**. This first block of patch merging and feature transformation is denoted as **"Stage 2"**.

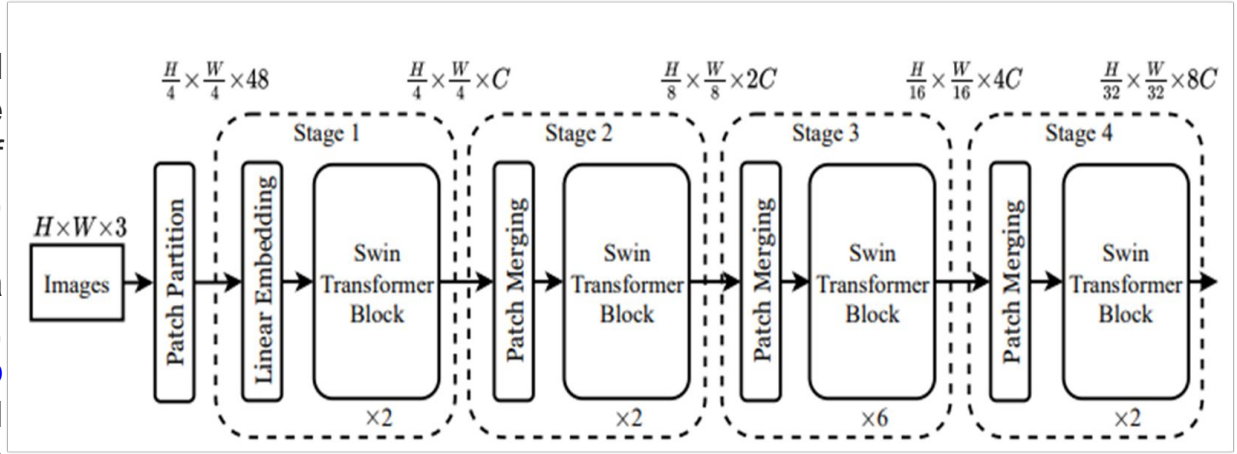


Transformer based

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Stage 3 & 4:

- The procedure is **repeated twice**, as **“Stage 3”** and **“Stage 4”**, with output resolutions of ***H*/16×*W*/16** and ***H*/32×*W*/32**, respectively.
- **These stages jointly produce a hierarchical representation**, with the **same feature map resolutions** as those of typical convolutional networks, such as VGGNet and ResNet, which can **conveniently replace the backbone networks in existing methods** for various vision tasks.



Transformer based

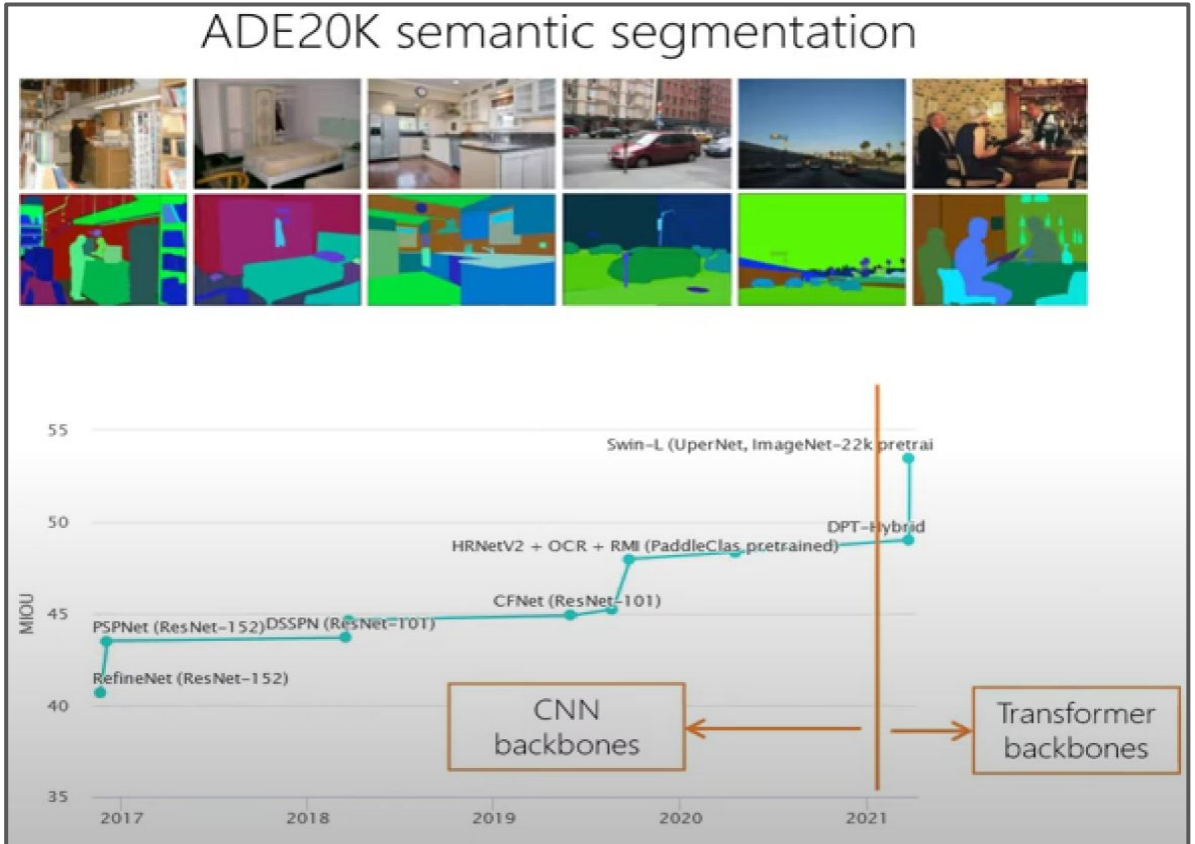
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

Performance:

- On **ADE20K dataset, as shown above** (20,000 training images, 150 categories) : #1
- Swin Transformer achieves SOTA on major video benchmarks.

Transformer based

ViT in Image Segmentation : Performance



LLM and Prompt Engineering based

SAM: Segment Anything

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... & Girshick, R. (2023). Segment anything. arXiv preprint arXiv:2304.02643.

- Foundation Model
  - Trained on SA-1B (1 billion masks and 11 M images) can be fine-tuned to downstream tasks
- **Promptable** Segmentation Model
- **Zero-shot** generalization
- **SA-1B dataset**
- **Inspired from models like ChatGPT**
- **Zero Shot Experiments**
  - Zero shot Single Point Valid Mask Evaluation
  - Zero-shot Edge Detection
  - Zero-shot Object Proposals
  - Zero-shot Instance Segmentation
  - Zero-shot Text-to-Mask
  -

Dataset [link]

11M images as the dataset for segment anything

SAM:Segment Anything

Abstract

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... & Girshick, R. (2023). Segment anything. arXiv preprint arXiv:2304.02643.

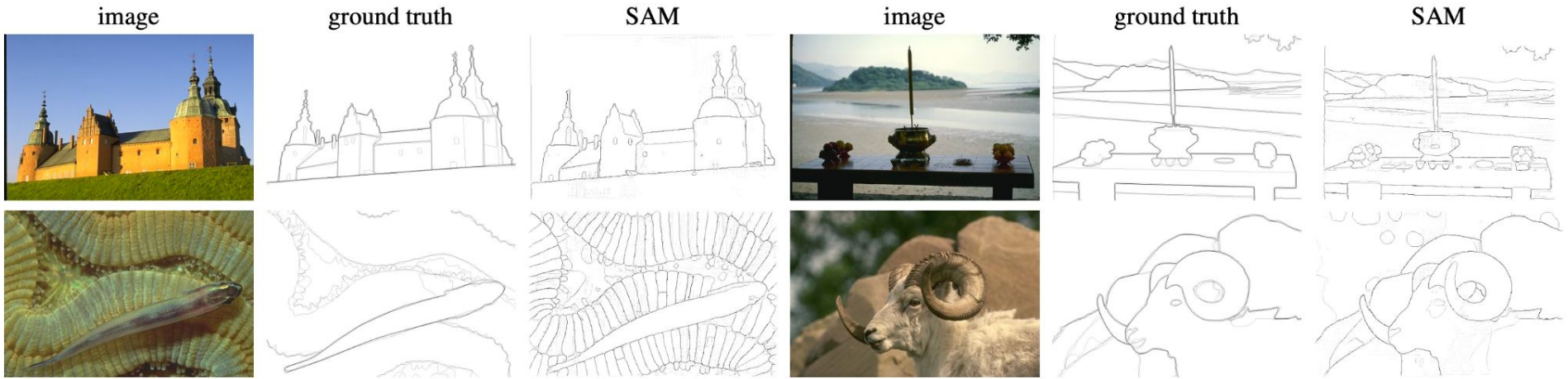


Figure 15: Additional visualizations of zero-shot edge predictions on BSDS500. Recall that SAM was not trained to predict edge maps and did not have access to BSDS images and annotations during training.

SAM:Segment Anything

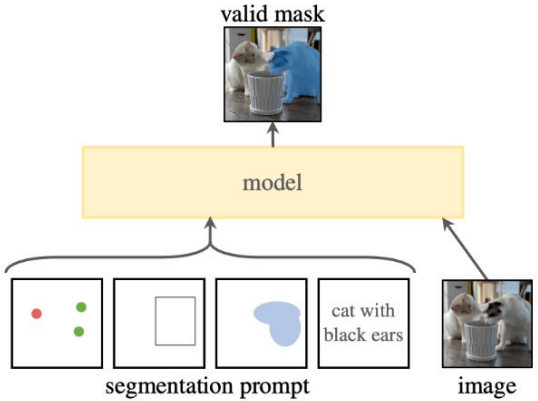
Abstract

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... & Girshick, R. (2023). Segment anything. arXiv preprint arXiv:2304.02643.

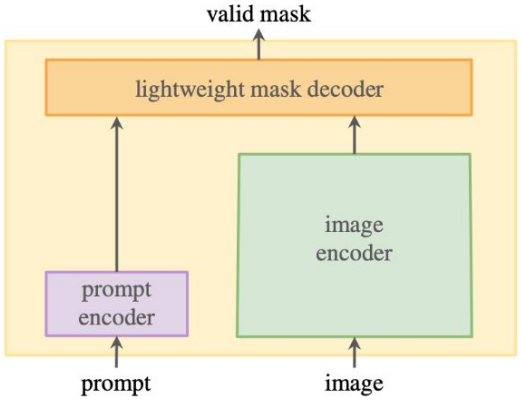
Task	Model	Data
Something with analogous capabilities	What is the corresponding **Model** Architecture?	What **Data** can power this task and model?
Promptable Segmentation Task where the objective is predict valid segmentation masks given (point + bbox + mask + text) as a complete prompt Pre-training Task	Prompt is encoded along with the image encoding and is fed to the lightweight mask decoder, to generate mask. Where pre-training task was mentioned to be generate valid masks	**Large Dataset,** which can help in generalization to downstream tasks. Meta generated SA-1B data to pretrain the whole model

Methods

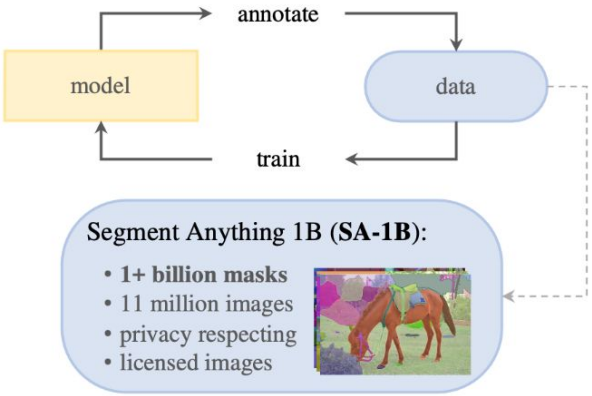
Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... & Girshick, R. (2023). Segment anything. arXiv preprint arXiv:2304.02643.



(a) **Task**: promptable segmentation



(b) **Model**: Segment Anything Model (**SAM**)



(c) **Data**: data engine (top) & dataset (bottom)

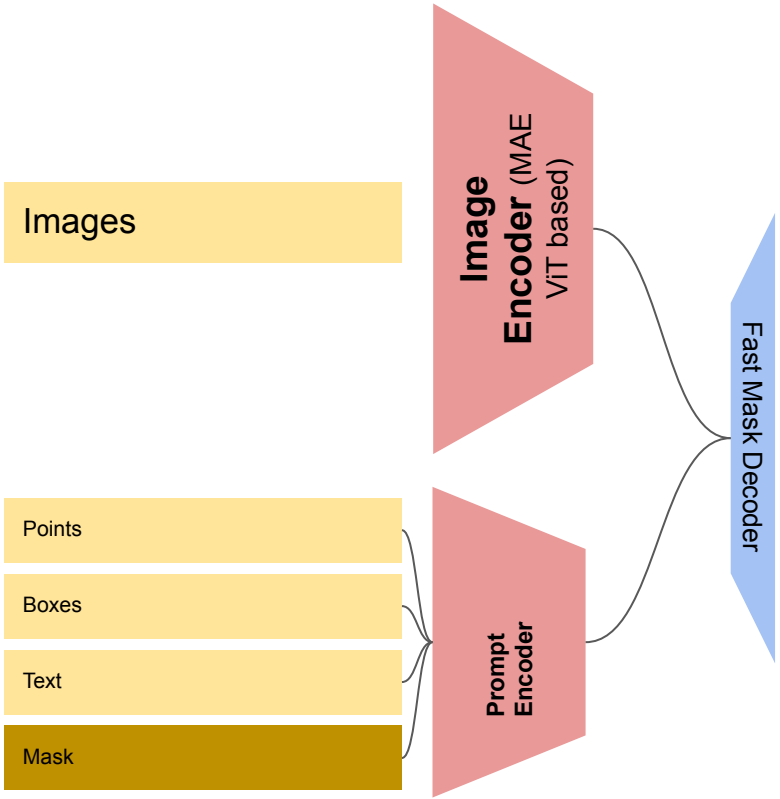
Task

Model

Data

Light Weight Mask Decoder + Heavy Weight Encoder

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... & Girshick, R. (2023). Segment anything. arXiv preprint arXiv:2304.02643.



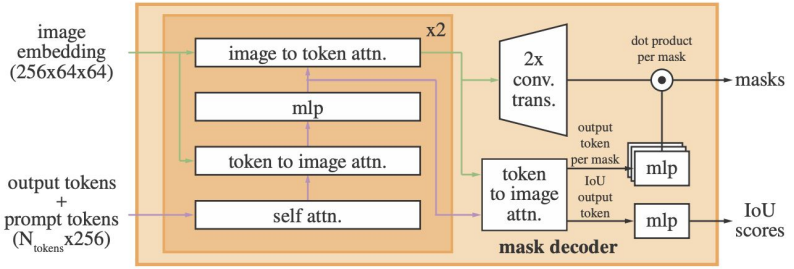


Figure 14: Details of the lightweight mask decoder. A two-layer decoder updates both the image embedding and prompt tokens via cross-attention. Then the image embedding is upscaled, from which the updated output tokens are used to dynamically predict masks. (Not illustrated for figure clarity: At every attention layer, positional encodings are added to the image embedding, and the entire original prompt token (including position encoding) is re-added to the token queries and keys.)

SAM:Segment Anything

Image Embeddings

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... & Girshick, R. (2023). Segment anything. arXiv preprint arXiv:2304.02643.

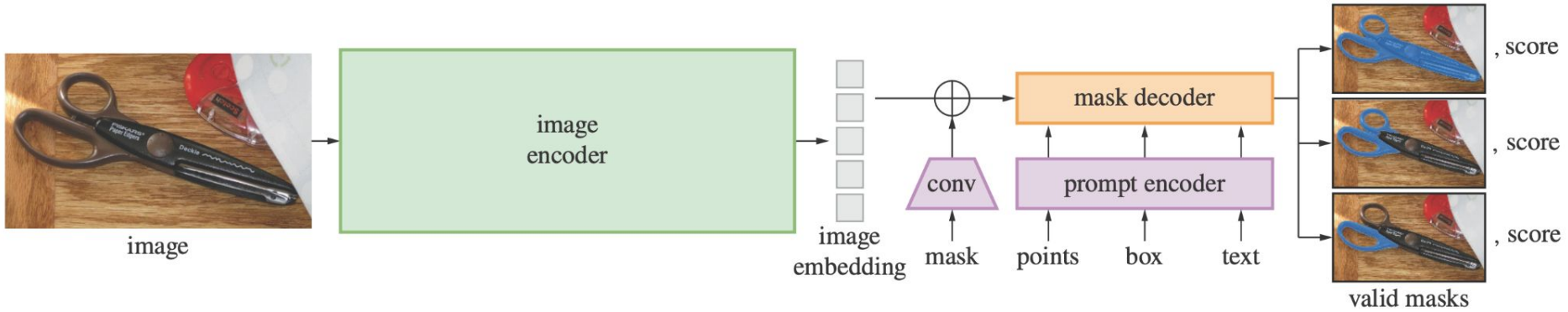


Figure 4: Segment Anything Model (SAM) overview. A heavyweight image encoder outputs an image embedding that can then be efficiently queried by a variety of input prompts to produce object masks at amortized real-time speed. For ambiguous prompts corresponding to more than one object, SAM can output multiple valid masks and associated confidence scores.

Results

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... & Girshick, R. (2023). Segment anything. arXiv preprint arXiv:2304.02643.

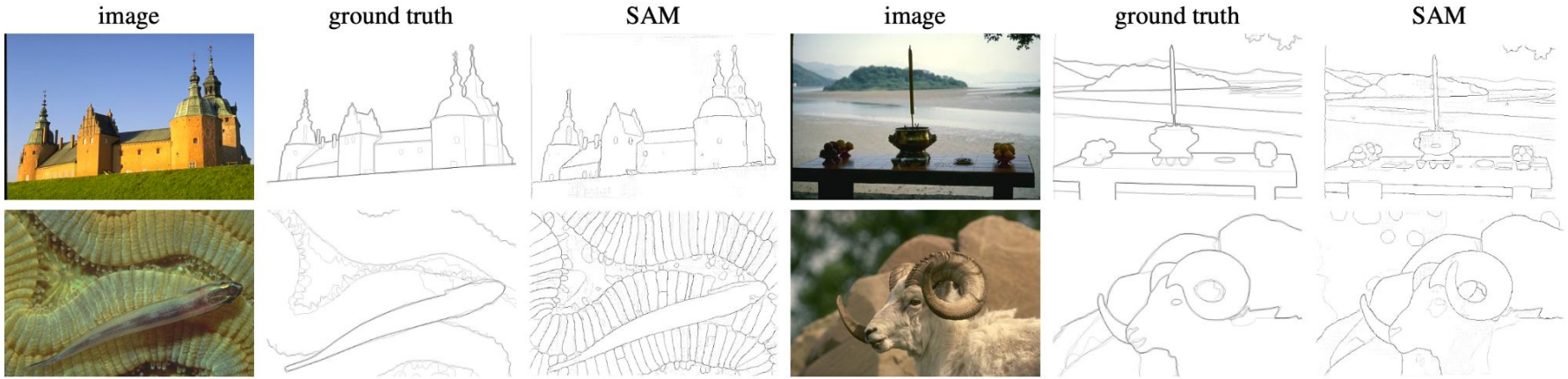


Figure 15: Additional visualizations of zero-shot edge predictions on BSDS500. Recall that SAM was not trained to predict edge maps and did not have access to BSDS images and annotations during training.

Results

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... & Girshick, R. (2023). Segment anything. arXiv preprint arXiv:2304.02643.

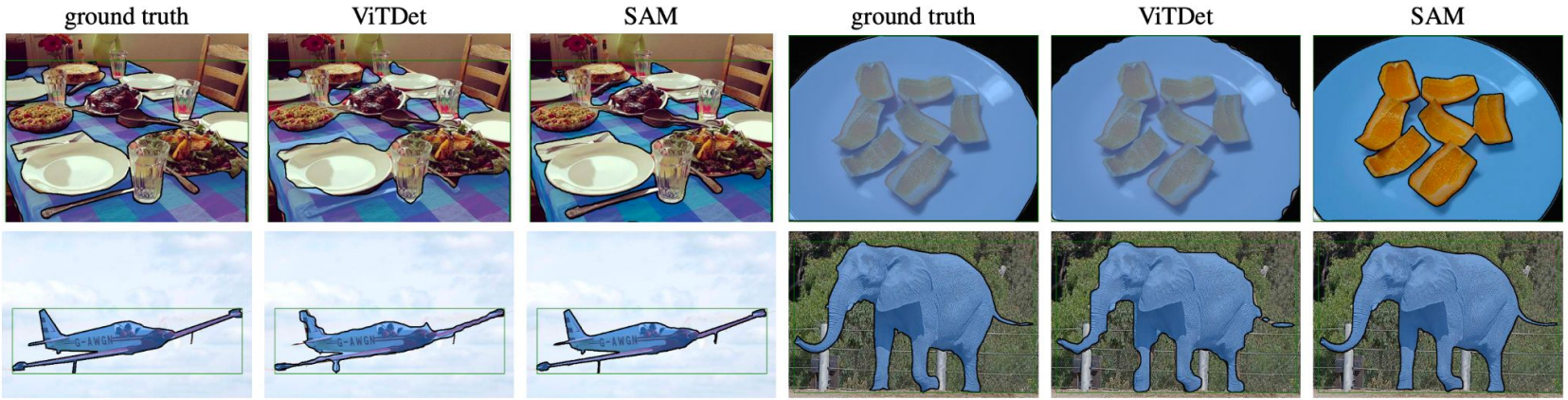
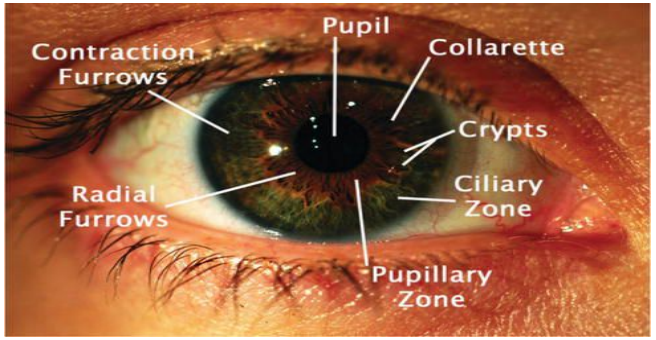


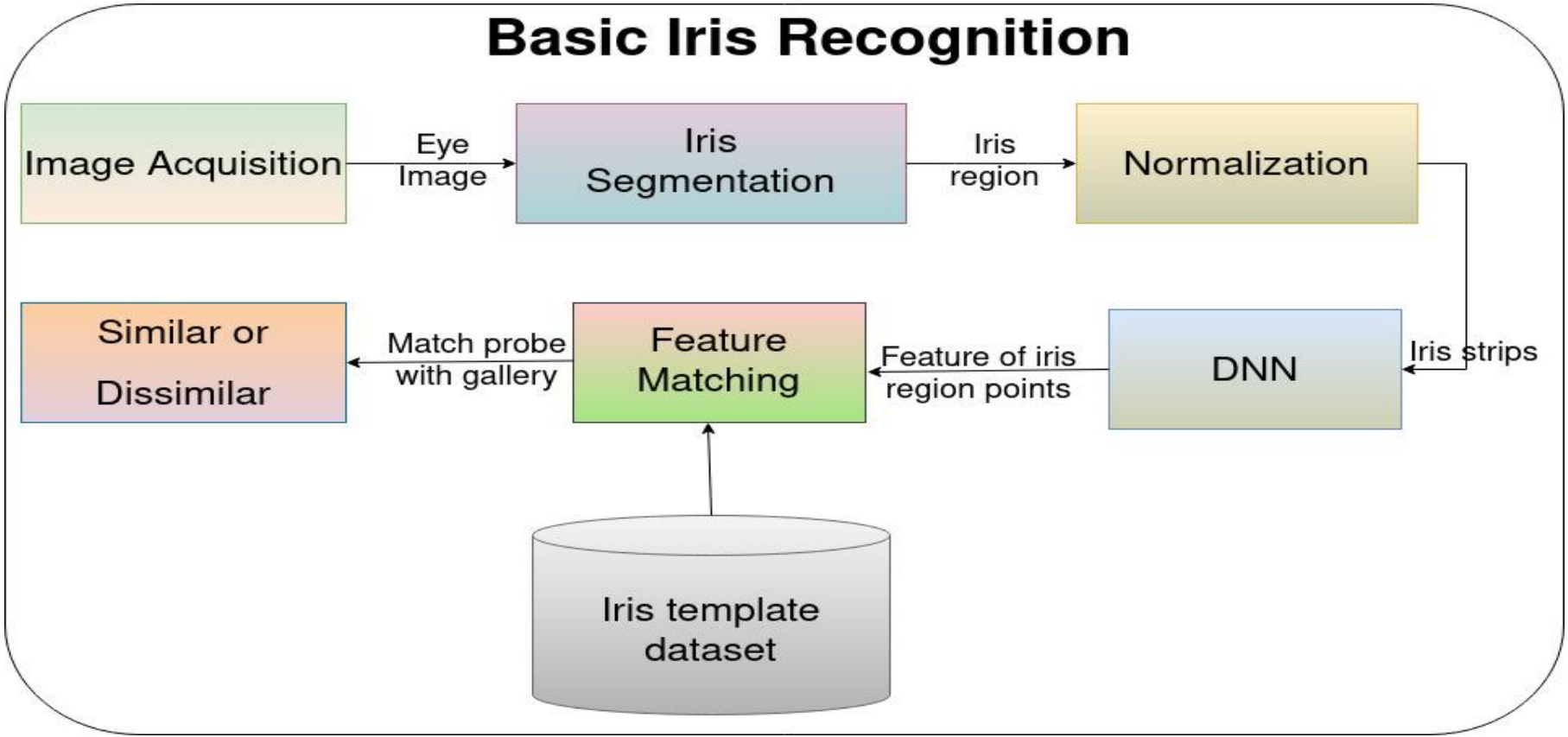
Figure 16: Zero-shot instance segmentation on LVIS v1. SAM produces higher quality masks than ViTDet. As a zero-shot model, SAM does not have the opportunity to learn specific training data biases; see top-right as an example where SAM makes a modal prediction, whereas the ground truth in LVIS is amodal given that mask annotations in LVIS have no holes.

Iris Recognition



- Iris recognition is a biometric technology that uses the unique patterns in the iris of the human eye for identification and authentication purposes.
- The most secure biometric modalities, as the patterns in the iris are highly stable throughout a person's lifetime and do not change with age.
- Iris recognition systems capture an image of the iris using specialized cameras and then extract and analyze the unique patterns within the iris.
- Iris templates are used for comparison and matching against a database of pre-enrolled templates to identify or authenticate individuals.
- Iris recognition offers several advantages over other biometric modalities, such as fingerprint or face recognition, including resistance to spoofing attacks and the ability to work in low light conditions.
- Challenges associated with iris recognition, such as variations in image quality, occlusions, and the need for cooperative subjects during image capture.

Recognition : Iris Biometric trait



Iris Recognition using Traditional and Deep neural network

Stages of Biometric Iris Recognition System

- **Sample Acquisition :** Initially the raw data is captured using the data sensor. It may require a setup and its own software routine. This is very critical and important stage as whatever is acquired in this stage will be used as the input to all the subsequent stages.

- **ROI Extraction :** From the raw data the actual region of interest (ROI) is extracted in this stage.

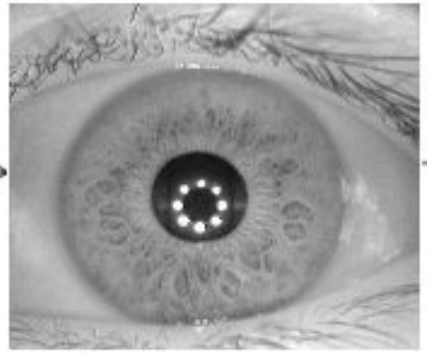
- **Quality Estimation :** The quality assessment of the sample ROI extraction is done in this stage. If the quality of the obtained ROI is poor then re-acquisition of the sample is done.

- **Sample Preprocessing** : In this stage the obtained ROI is preprocessed using several enhancement techniques. Also some transformations can be performed over it to get robust sample image representation.

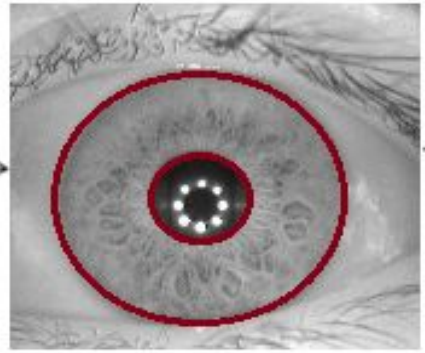
- **Feature Extraction and Matching :** In this stage robust feature vectors are computed and stored. Then they are used to compute a score for any given matching which can decide whether a it is genuine or imposter.

Iris Recognition Flow

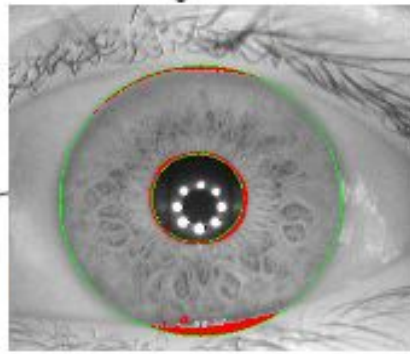
Image Acquisition



Localization

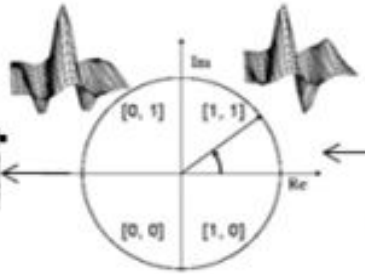


Segmentation



Noise masking

Polar representation



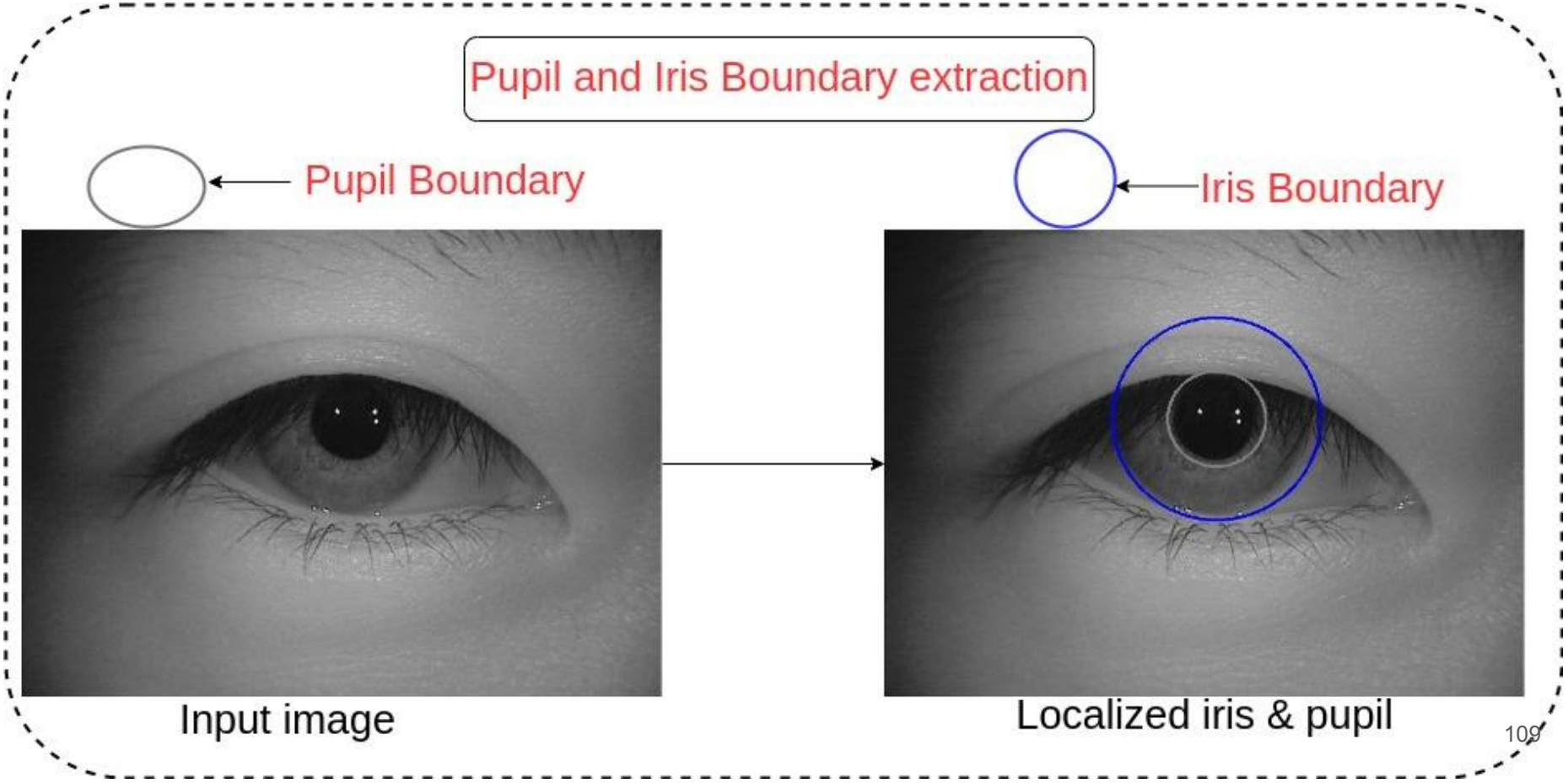
Feature extraction

IrisCode

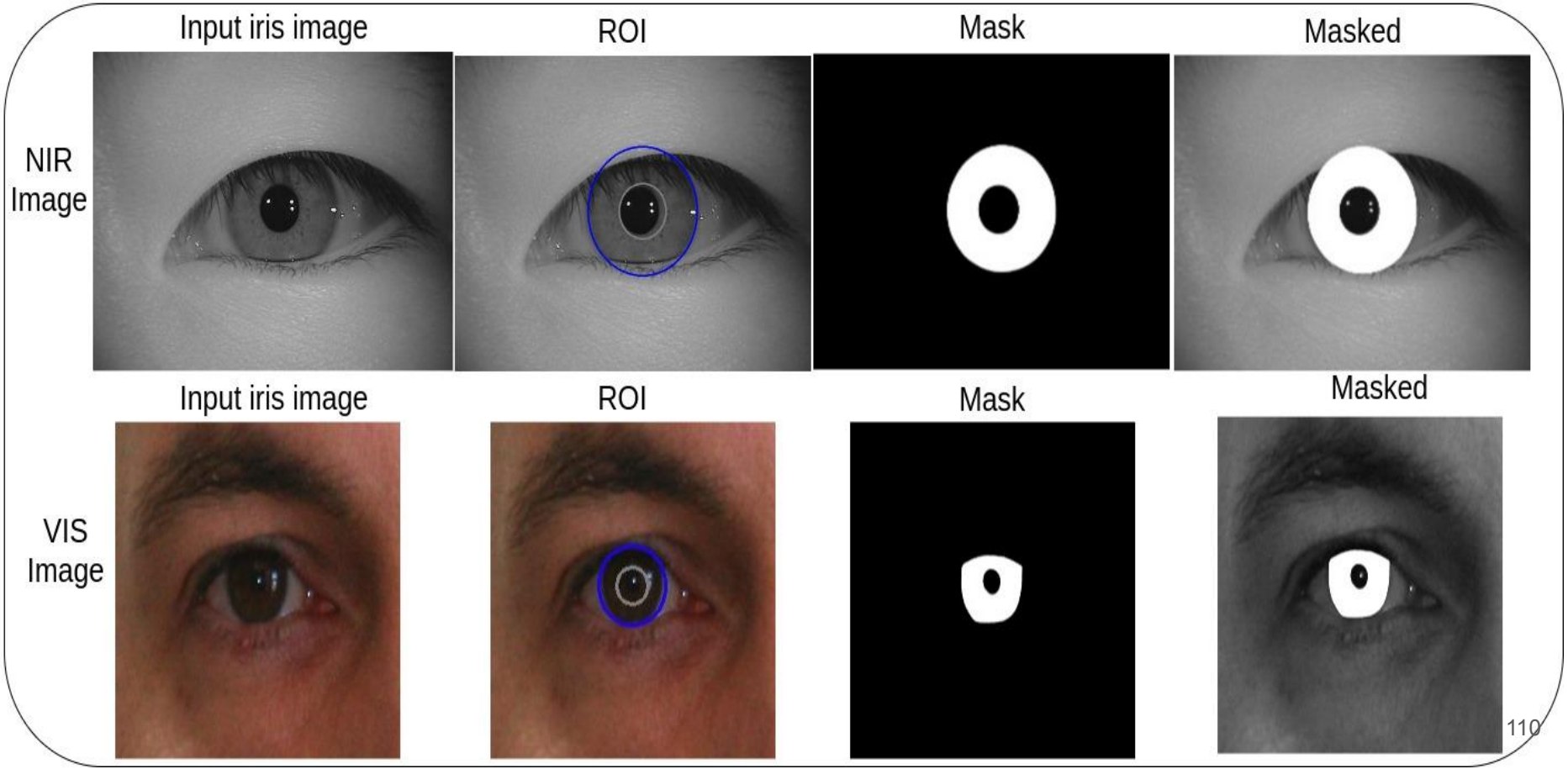
Pattern Matching

Database

Segmentation (Iris and Pupil)



Segmentation (Iris and Pupil)



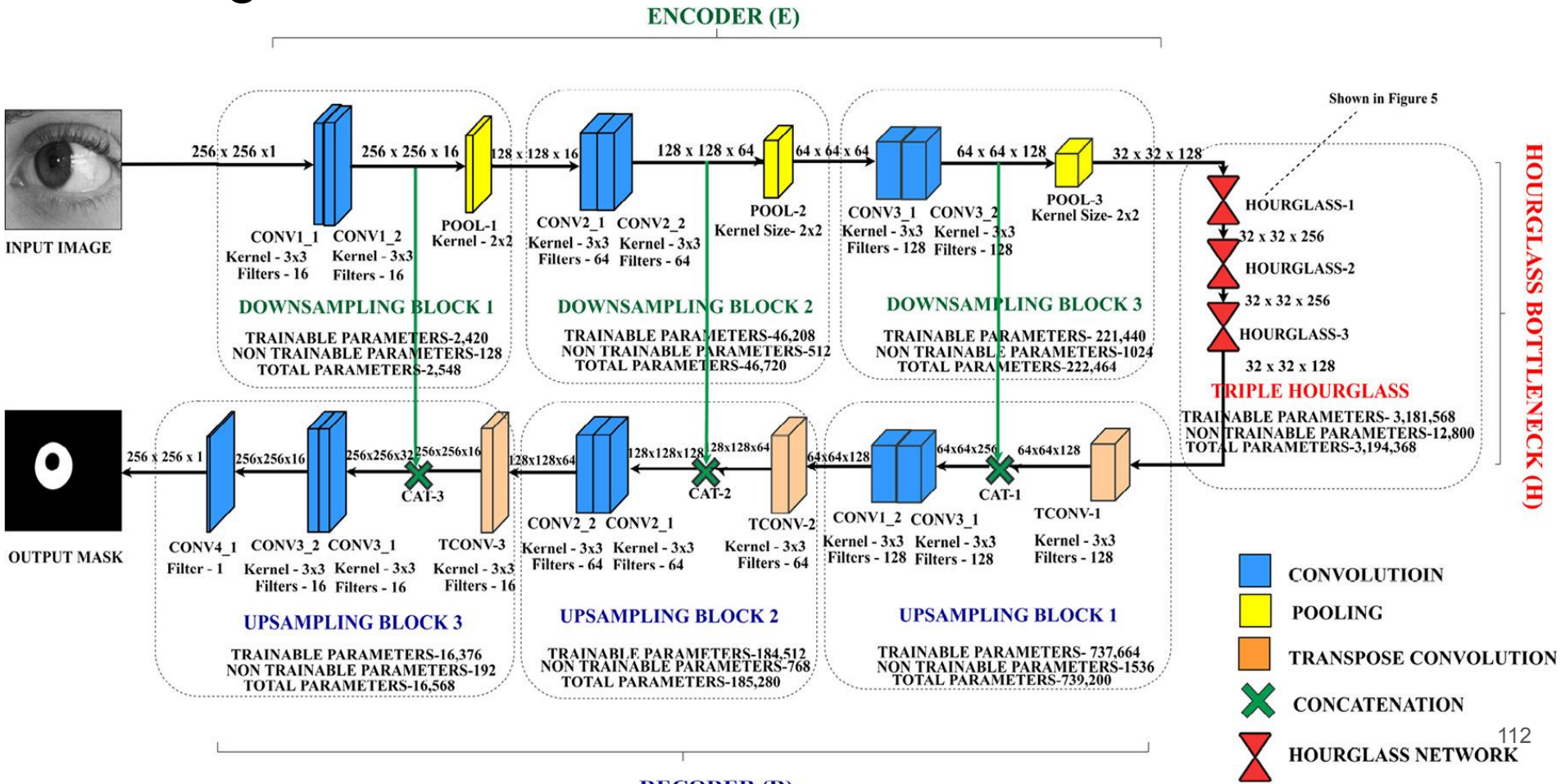
PixelSegNet: Pixel Level Iris Segmentation using Convolutional Encoder-Decoder with stacked Hourglass Bottleneck

Jha, Ranjeet Ranjan, Gaurav Jaswal, Divij Gupta, Shreshth Saini, and Aditya Nigam. *IET biometrics* 9, no. 1 (2020): 11-24.

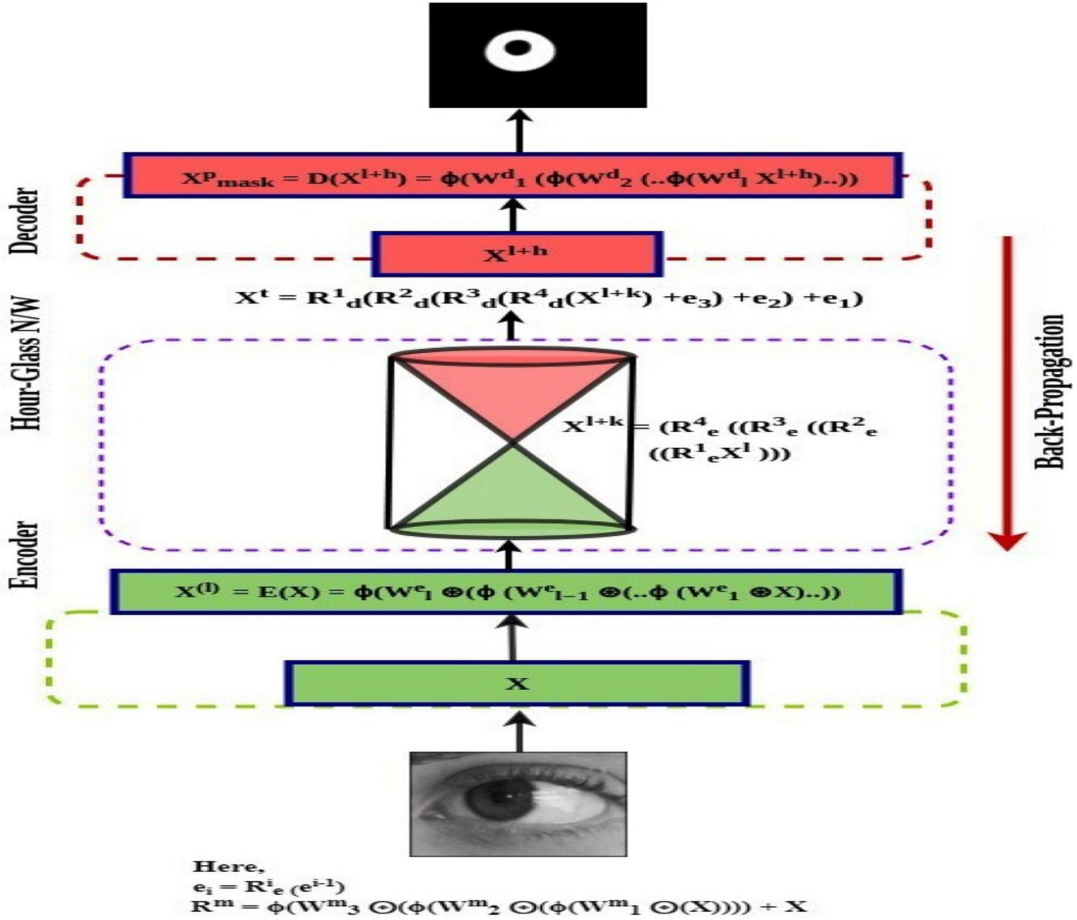
- **Aim :** To segmenting the images with variation in scale, off-angle, occlusion along with poor contrast, blurriness, etc.
- **PixlSegNet** consists majorly of three portions, namely encoder, decoder, and hourglass as a bottleneck of the encoder–decoder combination.
- Encoder takes input image to give feature map representation holding the contextual information taken by the decoder as input to produce corresponding segmentation mask.
- Long skip connections are formed between the encoder and the decoder wherein the corresponding feature maps from the encoder before downsampling are concatenated with the corresponding feature maps of the decoder after upsampling.

Convolutional Encoder - Decoder Network

PixelSegNet:



PixelSegNet: Segmentation Network



Convolutional Encoder - Decoder Network

PixelSegNet:

- The encoder network *E* maps the input image *X* to a latent representation , x^l .

$$X^l = E(X) = \phi(W_l^e \circledast (\phi(W_{l-1}^e \circledast (\cdots \phi(W_1^e \circledast X) \cdots))))$$

- Here $W_1^e, W_2^e, \ldots, W_l^e$ are the weights of the neurons at layers 1, 2, …, *l*, respectively, and X is the input vector, including the bias.

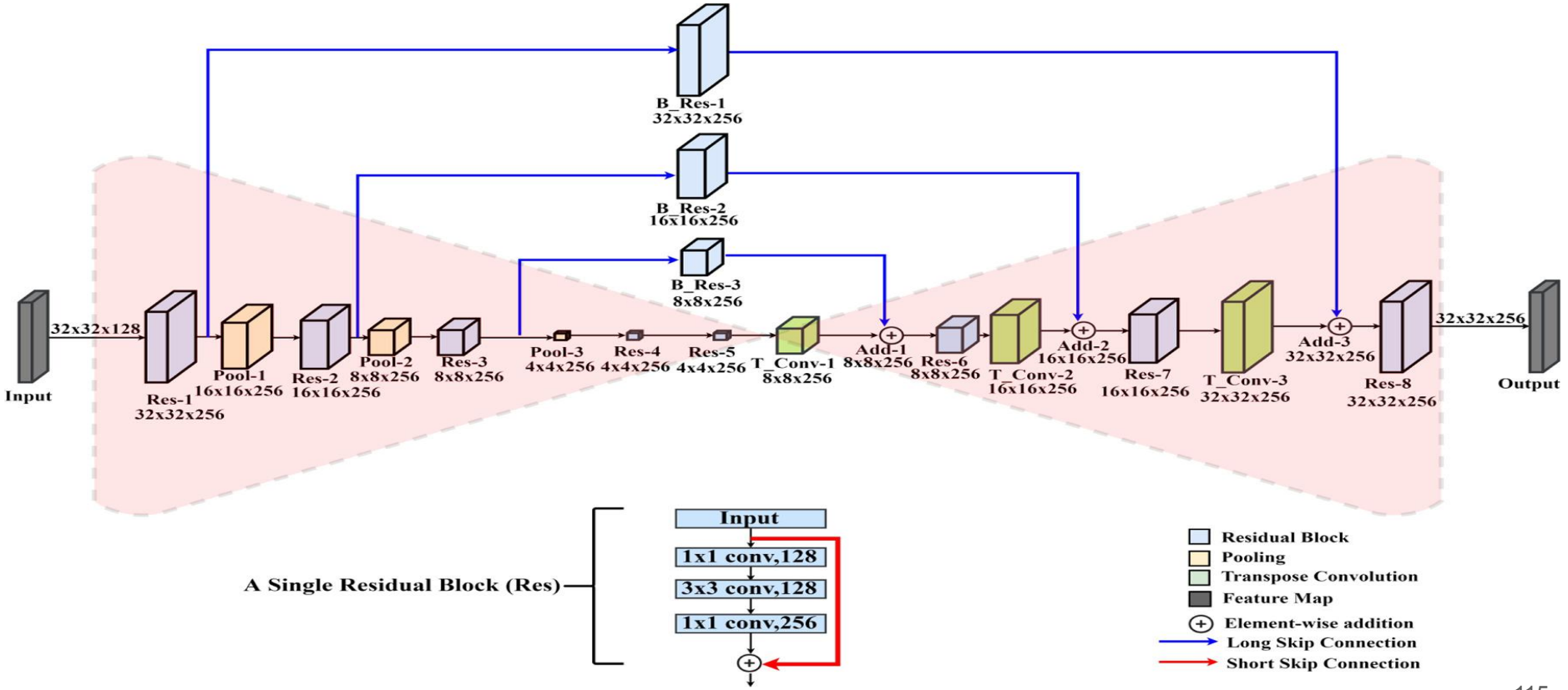
- Activation function (typically *Sigmoid, Relu*) has been applied after each layer to introduce the non-linearity.

- ***Encoder network architecture:*** It consists of repeated sets of dual convolutions (with padding 1 and stride 1) succeeded by ReLU activation and batch normalisation. Subsequently, each set is succeeded by a max-pool operation (size and stride 2).

- Each set is repeated but with filters having varied channel depths of 16, 64, 128.

- The majority of iris images contain the iris in a regular and an equally distributed (circular or elliptical) manner.

Hourglass Network



Hourglass Network along with Residual Block

Hourglass Network

Residual module *:* Each layer in the Encoder/Decoder of *H* has been realised as a residual module. mth residual block with three layers

$$R^m(X) = \phi(W_3^m \circledast (\phi(W_2^m \circledast (\phi(W_1^m \circledast (X)))))) + \boldsymbol{X}$$

Hourglass encoder : This network receives the input X^d from the encoder network E.

$$X^{d+k} = R_e^4(R_e^3(R_e^2(R_e^1(X^d))))$$

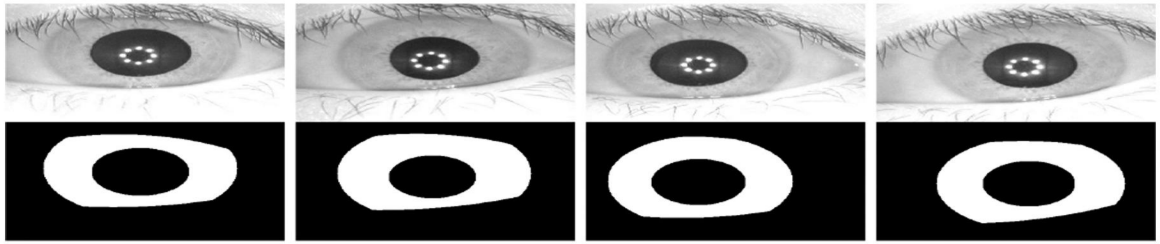
Hourglass Decoder : Hourglass decoder has been modelled with three long-term skip connections in order to preserve the spatial information.

$$X^{d+h} = R_d^1(R_d^2(R_d^3(R_d^4(X^{d+k}) + e_3) + e_2) + e_1)$$

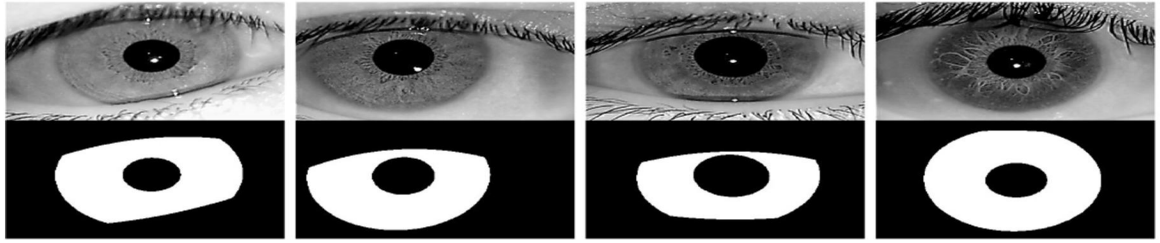
$$\text{where} \quad e_1 = R_s^1(R_e^1(X^d))$$
$$e_2 = R_s^2(R_e^2(R_e^1(X^d)))$$
$$e_3 = R_s^3(R_e^3(R_e^2(R_e^1(X^d))))$$

Hourglass Network : Dataset

UBIRIS-v2 images and corresponding ground-truth masks



CASIA-V3- Interval and corresponding ground-truth masks



IITD images and corresponding ground-truth masks

UBIRIS-V2

- 11,101 images of 261 distinct subjects
- Image size = 400*300
- captured from 4 to 8 m
- Canon EOS 5D camera with focal length 400 mm

CASIA-Iris V3 Interval

- 2655 images of 249 subjects
- collected using an iris sensor in two sessions with a time interval of one month
- Image size = 320*280

IITD data set

- 2240 images of the resolution 320x240
- captured using digital CMOS, JPC1000, JIRIS camera
- total of 224 subjects
- 48 and 176 are females and males

Convolutional Encoder - Decoder Network

PixelSegNet: Parameters

Network	Number of Parameters
Encoder network	314,048
Hourglass network	1,141,889
Decoder network	827,841
Total	**2,283,778**

Important points:-

1. Skip connections in encoder (E)–decoder (D) network.
2. Significance of using hourglass (H) network as a bottleneck.
3. Stacked hourglass (H) as a bottleneck.
4. Residual connections in the hourglass (H).
5. Skip connections in Hourglass (H)

Convolutional Encoder - Decoder Network (Contd.)

PixelSegNet Network : Loss Function

PixelSegNet network using two types of loss functions viz.

$$L(I^{p},\ I^{gt}) = L_{ce}(I^{p},\ I^{gt}) + L_{cl}(I^{p},\ I^{gt})$$

[i] cross-entropy and [ii] content loss. Hence our total loss *L* can be defined as

Cross-entropy Loss:

$$L_{ce}(I^{p},\ I^{gt}) = \frac{-1}{m * n} \sum_{j=1}^{n} \sum_{i=1}^{m} \Big[I^{gt}(i,\ j) * \log(I^{p}(i,\ j)) + (1 - I^{gt}(i,\ j)) * \log(1 - I^{p}(i,\ j)) \Big]$$

Content Loss :

$$L_{cl}(I^{p},\ I^{gt}) = \sum_{i=1}^{c} || E(I^{gt}(i)) - E\left((I^{p}(i)\right) ||^{2}$$

Regularisation:

$$J = L(Y^{p},\ Y^{gt}) + \lambda \Omega(\theta)$$

Hourglass Network : Performance parameters

Mean Segmentation Error :

$$PCL_k(I^p, I^{gt}) = \frac{1}{m*n}\sum_{j=1}^{n}\sum_{i=1}^{m} I^p(i,\ j) \oplus I^{gt}(i,\ j) \qquad E_1 = \frac{1}{N}\sum_{k=1}^{N} PCL_k$$

Pixel-wise classification error PCL_k of any kth test image.

Type II Error : $E_2^k = 0.5 * \mathrm{FPR} + 0.5 * \mathrm{FNR}$

$$\mathrm{FPR} = \frac{1}{m*n}\sum_{j=1}^{n}\sum_{i=1}^{m}[((I^{gt}(i,\ j) \cdot I^p(i,\ j)) \oplus I^p(i,\ j)]$$

$$\mathrm{FNR} = \frac{1}{m*n}\sum_{j=1}^{n}\sum_{i=1}^{m}[((I^{gt}(i,\ j) \cdot I^p(i,\ j)) \oplus I^{gt}(i,\ j)]$$

Jaccard index (JI)/intersection over union (IOU) :

$$\mathrm{JI} = \frac{1}{L}\sum_{i=1}^{L}\frac{C_{ii}}{G_i + P_i - C_{ii}}$$

Precision:-

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

Recall : -

$$R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

F-Measure : -

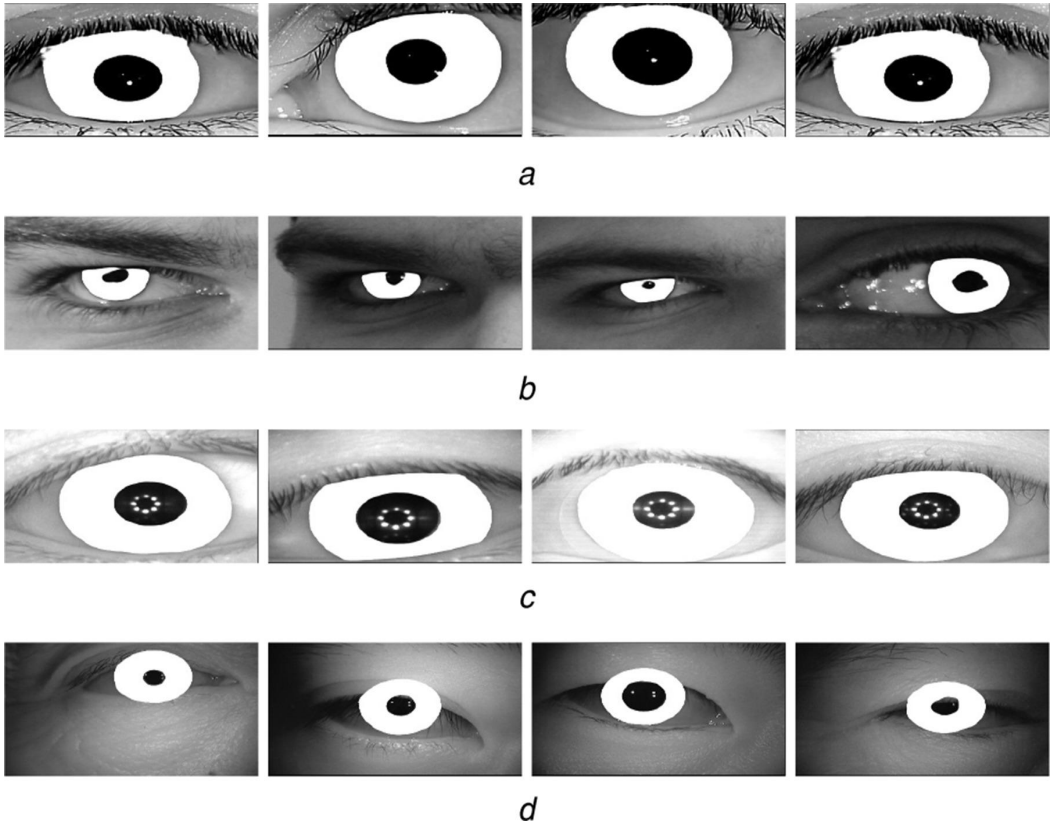
$$\mathrm{F-measure} = \frac{2RP}{R+P}$$

Accuracy : -

$$\mathrm{Accuracy} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

Convolutional Encoder - Decoder Network (Contd.)

Hourglass Network : Result



Results on challenging images (images containing variation, in contrast, rotation, blurriness etc.)

(a) **IITD iris images,**

(b) **UBIRIS iris images,**

(c) **CASIA-iris-Interval images,**

(d) **CASIA-iris-Lamp images**

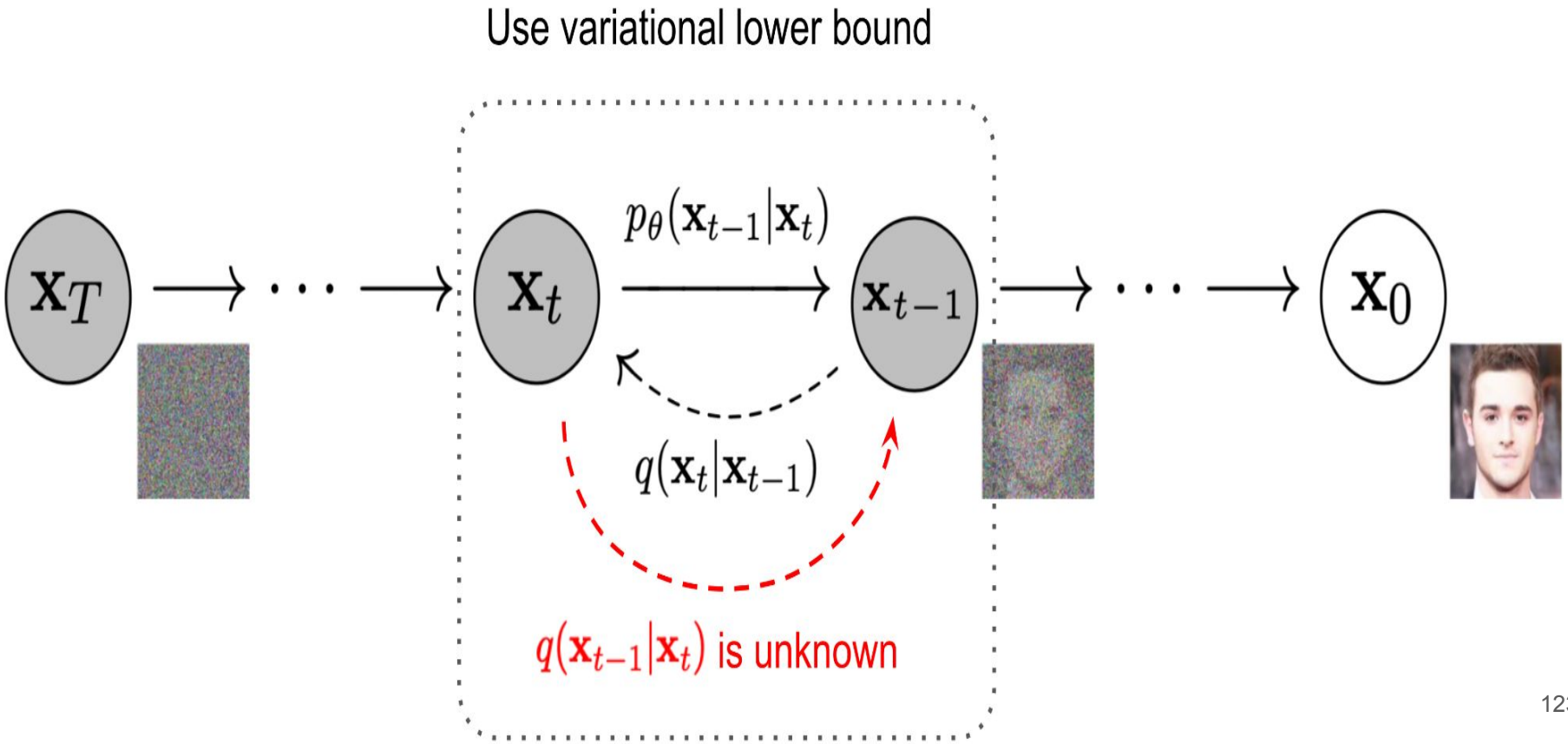
DiffusionInst: Diffusion Model

Gu, Zhang Xuan, Haoxing Chen, Zhuoer Xu, Jun Lan, Changhua Meng, and Weiqiang Wang. *arXiv preprint arXiv:2212.02773*

- Diffusion models usually use two Markov chains: a forward chain that perturbs the image to noise and a reverse chain that refines noise back to the image.
- Given a data distribution x0 ~ q(x0), the forward noise perturbing process at time t is defined as q(xt|xt−1).
- It gradually adds Gaussian noise to the data according to a variance schedule β1, · · · , βT : q(xt|xt−1) = N (xt; p 1 − βt xt−1, βt I).
- Given x0, we can easily obtain a sample of xt by sampling a Gaussian vector ~ N (0, I) and applying the transformation as follows: xt = √ α¯tx0 + (1 − α¯t), where α¯t = Qt s=0(1 − βs).
- During training, a neural network is trained to predict x0 from xt for different t ∈ {1, · · · , T}.
- While performing inference, we start from a random noise xT and iteratively apply the reverse chain to obtain x0.

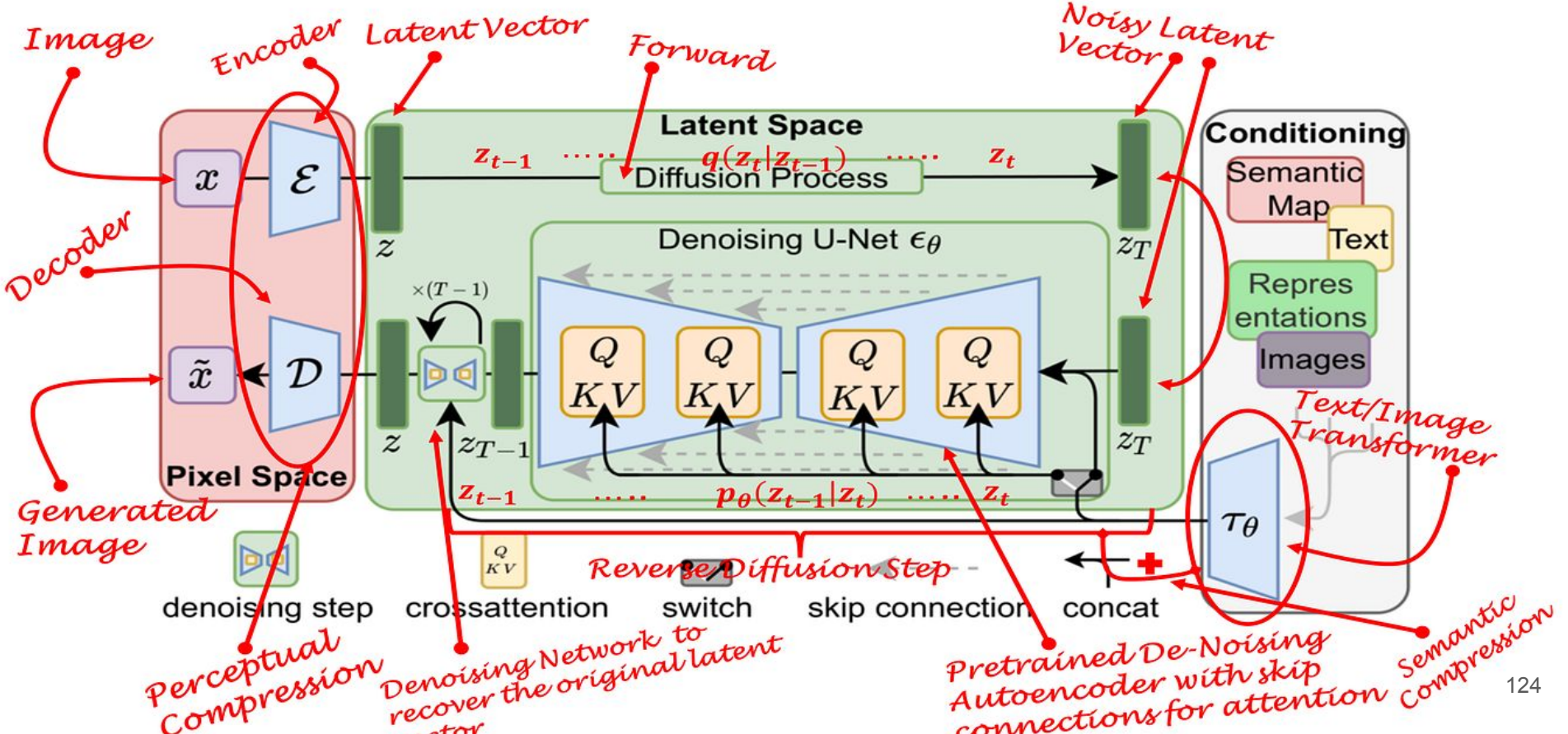
Diffusion Based Network

Diffusion Model



Diffusion Based Network

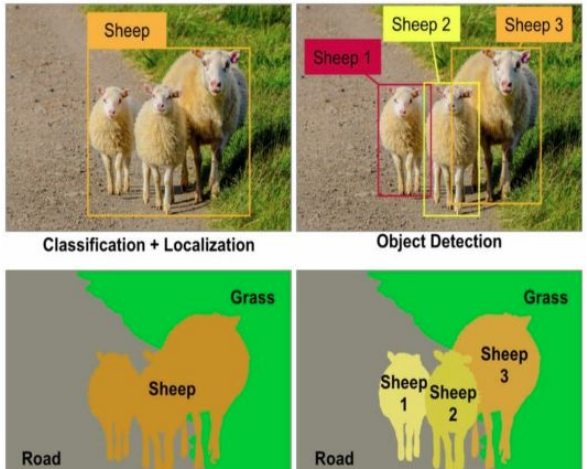
Diffusion Model



DiffusionInst: Diffusion Model for Instance Segmentation:

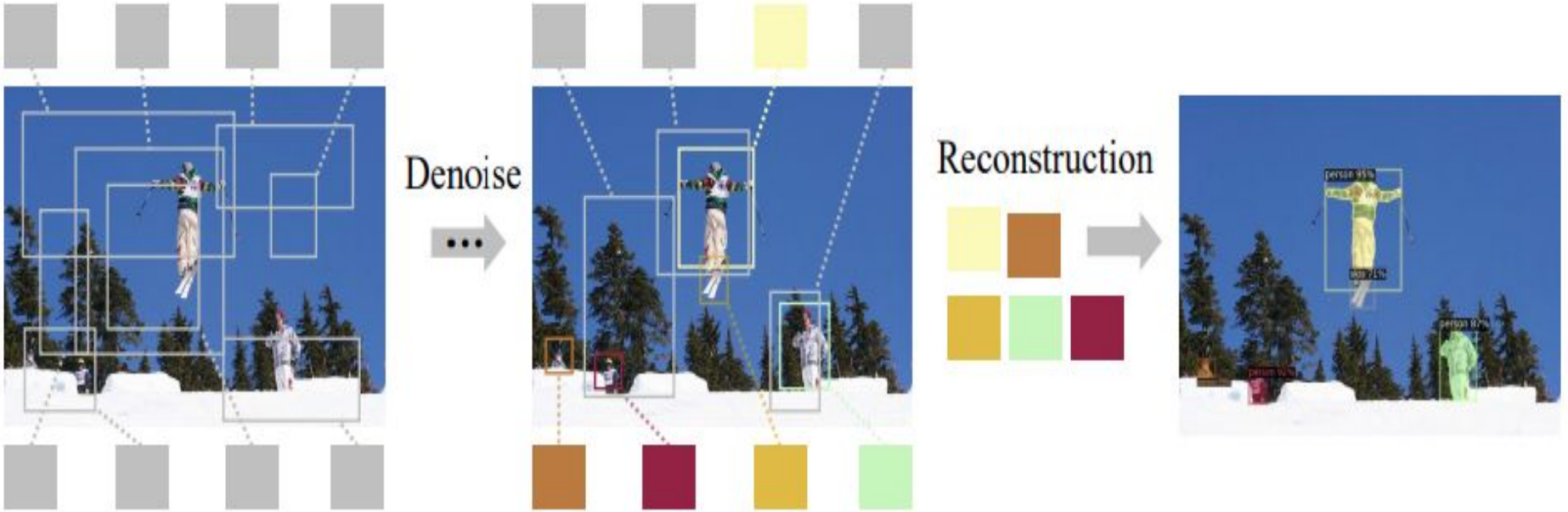
Gu, Zhang Xuan, Haoxing Chen, Zhuoer Xu, Jun Lan, Changhua Meng, and Weiqiang Wang. *arXiv preprint arXiv:2212.02773*

- Instance segmentation aims to represent objects with binary masks, which is a finer-grained representation compared to the bounding boxes of object detection.
- Instance segmentation is a computer vision task that involves identifying objects individually within an image, assigning a unique label to each object pixel.
- It combines object detection, which localizes objects, with semantic segmentation, which classifies pixels, enabling detailed understanding and separation of multiple instances within an image.
- Two-stage methods first detect objects, then crop their region features with RoI alignment to further classify each pixel.
- To adapt the diffusion model in instance segmentation is still an open problem.



DiffusionInst: Diffusion Model for Instance Segmentation:

Gu, Zhangxuan, Haoxing Chen, Zhuoer Xu, Jun Lan, Changhua Meng, and Weiqiang Wang. *arXiv preprint arXiv:2212.02773*



Diffusion model for instance segmentation: Author's propose to regard instance segmentation as a denoising diffusion process from noisy bounding boxes and filters to instance masks with a dynamic mask head for mask reconstruction.

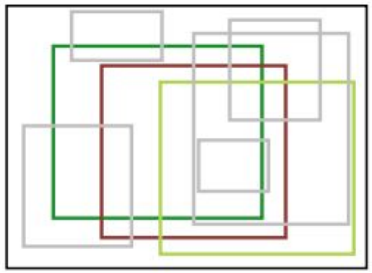
DiffusionInst: Diffusion Model for Instance Segmentation:

Gu, Zhangxuan, Haoxing Chen, Zhuoer Xu, Jun Lan, Changhua Meng, and Weiqiang Wang. *arXiv preprint arXiv:2212.02773*

GroundTruth

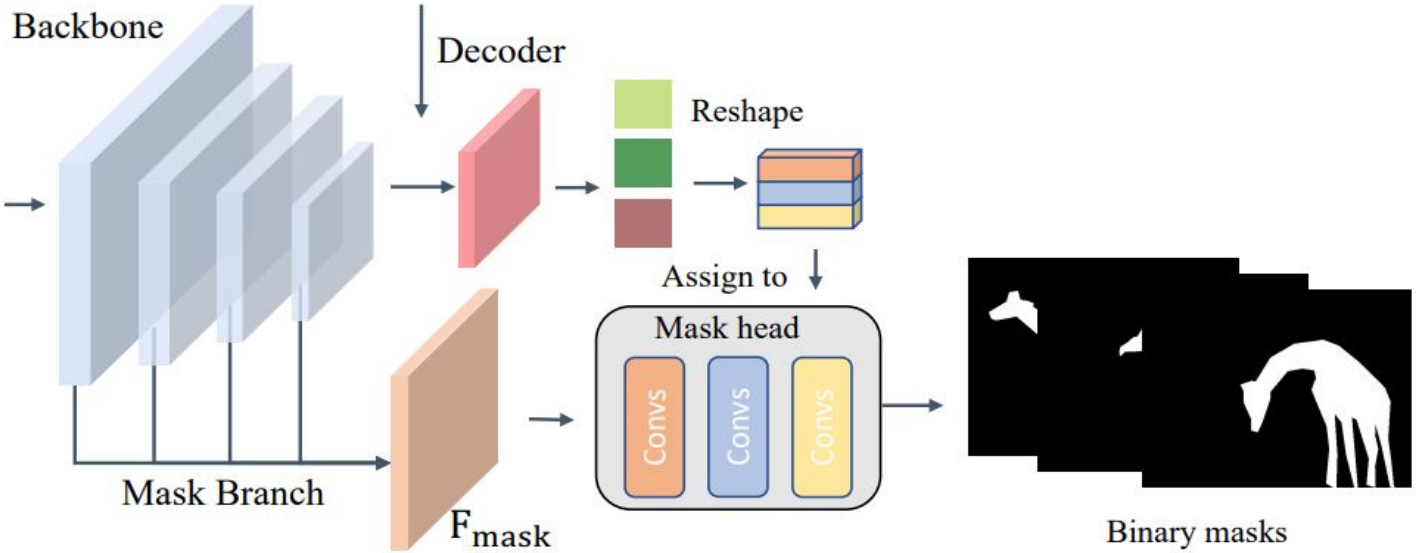
Gaussian Noise & Pad

With $t \in [0, T]$

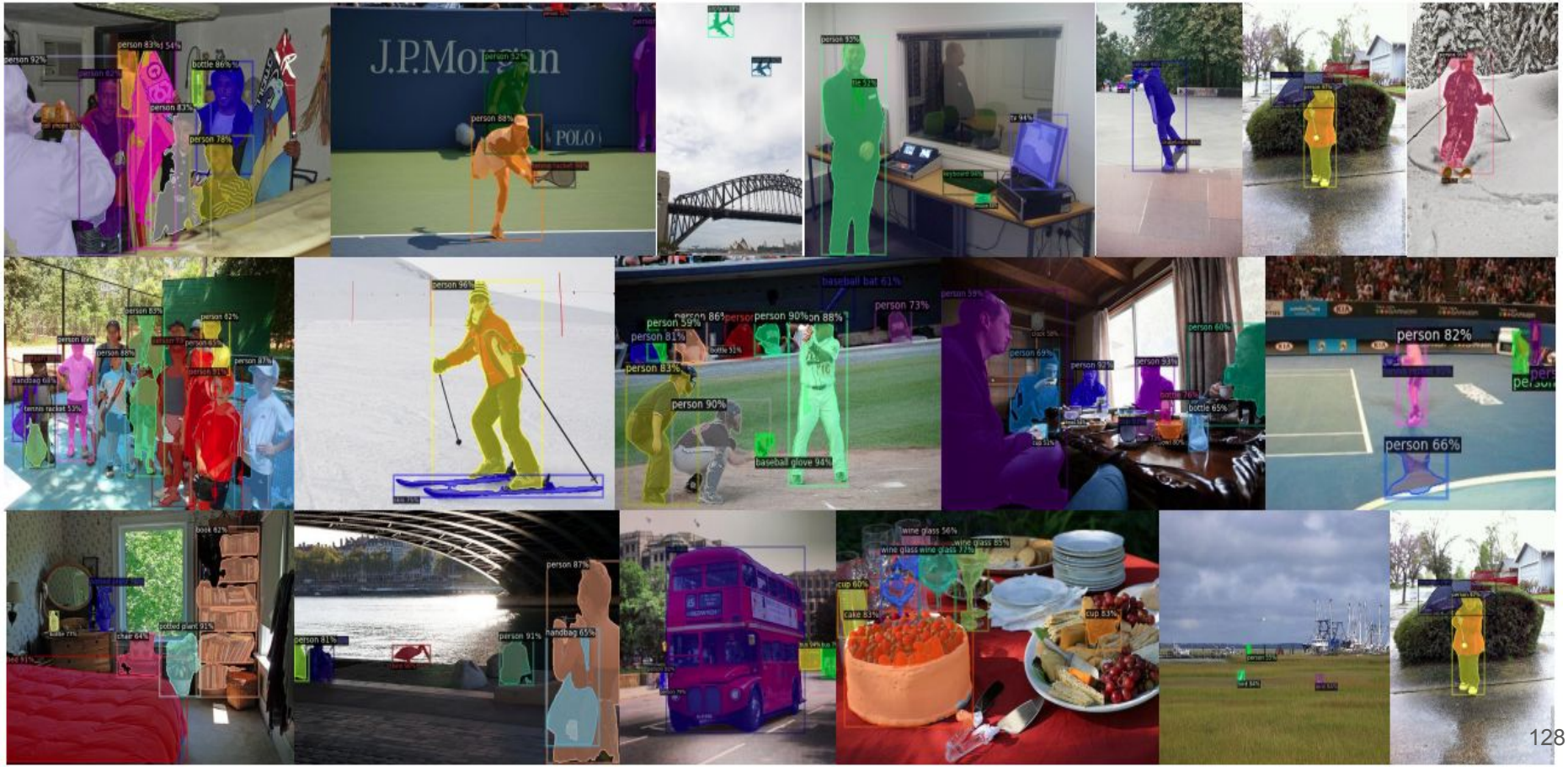


Instance-aware Filters θ

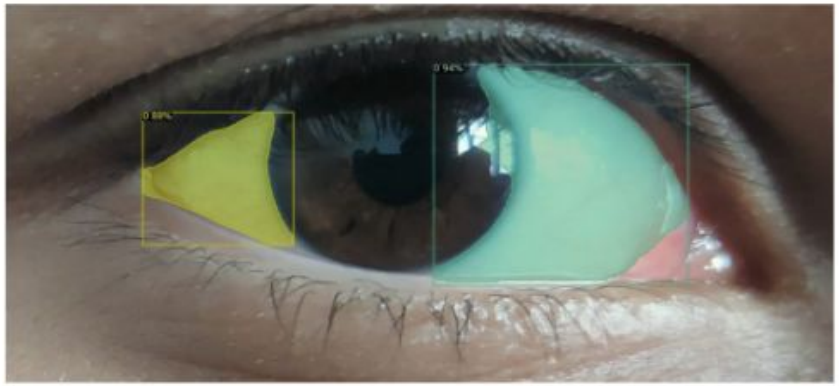
Input Image

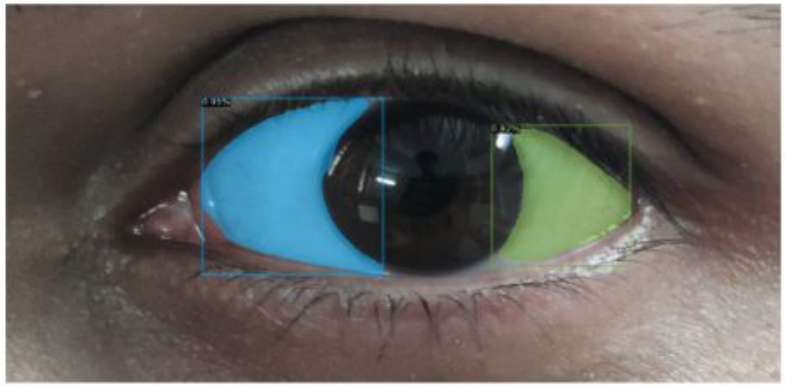


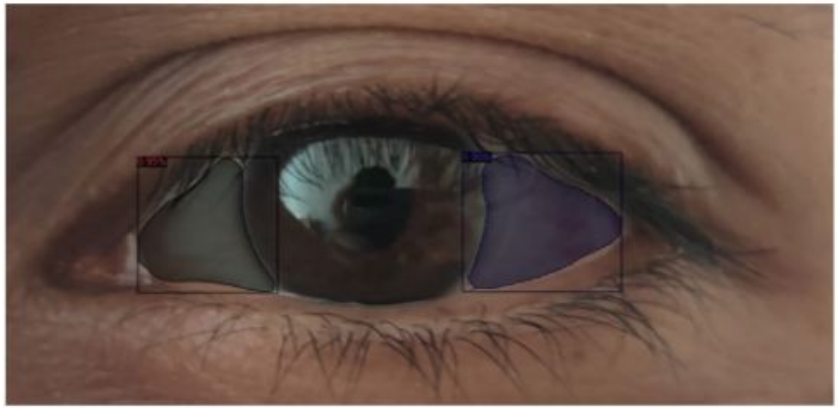
DiffusionInst: Performance on COCO 2017 Dataset

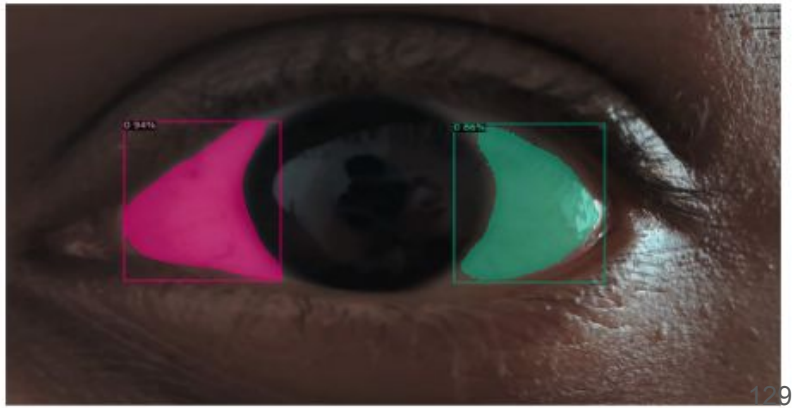


DiffusionInst: Performance on Sclera Dataset



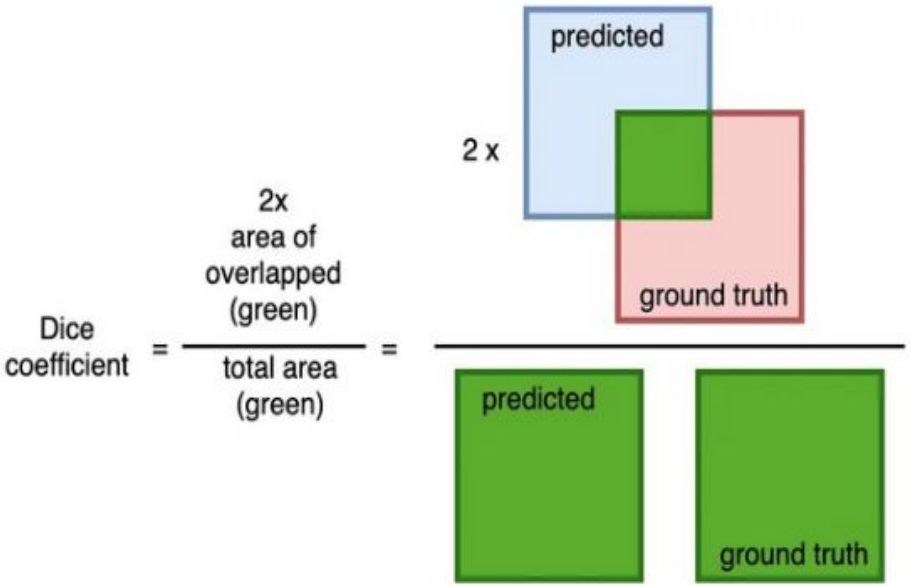






DiffusionInst: Loss Function

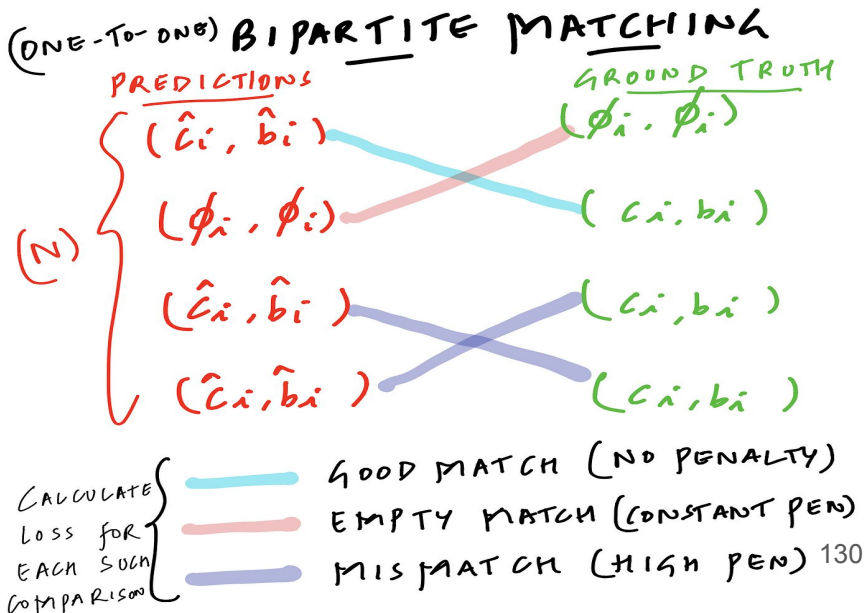
Dice Loss : Dice coefficient is a measure of overlap between two masks. 1 indicates a perfect overlap while 0 indicates no overlap.



Set Prediction Loss :

- It produces an optimal bipartite matching between predicted and ground truth objects, and then optimize object-specific (bounding box) losses.

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\min} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}),$$



DiffusionInst: Evaluation Metrics

Evaluation metrics refer to the evaluation metrics used for object detection and segmentation detection tasks. These include :

- **Average Precision (AP):** Average Precision measures the accuracy of object detection and instance segmentation by considering the precision and recall trade-off. It calculates the area under the precision-recall curve.
- **Average Precision at IoU (Intersection over Union) thresholds:** This metric computes the Average Precision at specific IoU thresholds.
- **Average Precision for different object sizes:** COCO evaluation metrics also provide Average Precision values for objects of different sizes.
- **AP** - Average Precision
- **AP50** - Average Precision at IoU threshold of 0.5
- **AP75** - Average Precision at IoU threshold of 0.75
- **APs** - Average Precision for small objects
- **APm** - Average Precision for medium objects
- **APl** - Average Precision for large objects

Iris Segmentation: Applications

Iris Recognition : The iris region from an eye image, features such as texture, patterns, and unique characteristics can be extracted for identification or verification purposes. Iris recognition is widely used in access control systems, national identification programs, and secure authentication processes.

Ocular Disease : It can aid in the diagnosis and monitoring of ocular diseases. Segmenting the iris, ophthalmologists can analyze the structure and characteristics of the iris, which can provide valuable insights into conditions like glaucoma, diabetic retinopathy, and iris melanoma.

Gaze Tracking : Iris segmentation is essential for gaze tracking systems.

Eye Tracking in Behavioral research : It is used for eye tracking studies. By tracking the position and movement of the iris, researchers can gain insights into visual attention, perception, reading patterns, and user experience.

Contact lens Fitting : Measurements can be made to determine the appropriate size and shape of contact lenses for a person's eye

Iris based Human Computer Interaction : It is used used in iris-based human-computer interaction systems, where the iris movements and patterns are used as input for controlling devices, interacting with virtual environments, or playing games.

Slides Overview and References

- **Image Segmentation** : Definition, Types, Algorithms, Datasets, Loss Functions, Evaluation Metrics
  - https://www.width.ai/post/semantic-segmentation-vs-instance-segmentation
  - S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz and D. Terzopoulos, "Image Segmentation Using Deep Learning: A Survey," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 7, pp. 3523-3542, 1 July 2022.

- **Fully Convolutional Networks (FCNs)**
  - ParseNET
    - J. Long, E. Shelhamer and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 3431-3440, doi: 10.1109/CVPR.2015.7298965.
    - Liu, W., Rabinovich, A., & Berg, A. C. (2015). *ParseNet: Looking Wider to See Better*. 1–11. http://arxiv.org/abs/1506.04579
    - https://blog.paperspace.com/global-pooling-in-convolutional-neural-networks/

Slides Overview and References (Contd)

- **Encoder-Decoder Models for Image Segmentation**
  - SegNet
    - Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.
    - Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017): 2481-2495.
    - https://www.youtube.com/@nptel-nociitm9240/semanticsegmentation
  - U-Net
    - Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18 (pp. 234-241). Springer International Publishing.
    - https://theaisummer.com/unet-architectures/
    - https://theaisummer.com/skip-connections/

- **Multi-Scale and Pyramid Network Based Models**
  - PSPNet
    - Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, *2017-Janua*, 6230–6239. https://doi.org/10.1109/CVPR.2017.660

Slides Overview and References (Contd)

- **CNN-based Image Segmentation Architectures**
  - Mask R-CNN
    - Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. https://doi.org/10.1109/tpami.2016.2577031
    - https://www.slideshare.net/hitheone/maskrcnn-for-instance-segmentation-117485267
    - He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). *Mask R-CNN*. https://doi.org/10.1109/iccv.2017.322
    - https://www.cs.utexas.edu/~robertom/cs391r_fall2022/slides/CS391R%20Presentation%20Mask%20RCNN.pdf
- **Transformer-based Image Segmentation Architectures**
  - ViT
    - Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).
    - Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." *arXiv preprint arXiv:2010.11929* (2020).
    - https://www.youtube.com/watch?v=HZ4j_U3FC94
    - https://huggingface.co/transformers/v4.12.5/model_doc/vit.html
  - Swin
    - Liu, Ze, et al. "Swin transformer: Hierarchical vision transformer using shifted windows." *Proceedings of the IEEE/CVF international conference on computer vision*. 2021.
    - https://keras.io/examples/vision/probing_vits/
    - https://jacobgil.github.io/deeplearning/vision-transformer-explainability
    - http://jalammar.github.io/illustrated-transformer
    - https://github.com/microsoft/Swin-Transformer
    - https://sh-tsang.medium.com/review-swin-transformer-3438ea335585
    - https://towardsdatascience.com/a-comprehensive-guide-to-swin-transformer-64965f89d14c#:~:text=The%20Swin%20Transformer%20block%20consists,(SW%2DMSA)%20module.

Slides Overview and References (Contd)

- **LLM and Prompt Engineering based**
  - SAM
    - Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., ... & Girshick, R. (2023). Segment anything. arXiv preprint arXiv:2304.02643.
    - Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dolla´r, and Ross Girshick. Masked autoencoders are scalable vision learn- ers. CVPR, 2022. 5, 8, 12, 16, 17
- **Recent Works:**
  - Iris Recognition
    - Daugman, J., 2009. How iris recognition works. In *The essential guide to image processing* (pp. 715-739). Academic Press.
  - Segmentation (Iris and Pupil)
    - PixelSegNet and DiffusionInst
    - Gu, Zhangxuan, Haoxing Chen, Zhuoer Xu, Jun Lan, Changhua Meng, and Weiqiang Wang. "Diffusioninst: Diffusion model for instance segmentation." *arXiv preprint arXiv:2212.02773* (2022).
    - Jha, Ranjeet Ranjan, Gaurav Jaswal, Divij Gupta, Shreshth Saini, and Aditya Nigam. "PixISegNet: pixel‐level iris segmentation network using convolutional encoder–decoder with stacked hourglass bottleneck." *IET biometrics* 9, no. 1 (2020): 11-24.