Image Segmentation Losses Classification

Introduction

Convolutional Neural Networks (CNN) are primarily used in image-related tasks like image recognition and classification. Image Segmentation involves classifying each pixel into an object instance, that is going to each pixel and asking which object class does it belong to.

The CNN architecture outputs SoftMax probability matrix and we classify the pixel corresponding to the highest SoftMax value. We have a likelihood loss function between the SoftMax values generated by the CNN architecture and the ground truth labels. The likelihood loss function plays a key role in the performance and needs to be chosen carefully because

- Loss function can help you to adapt to your end goal. You can augment your own devised loss function with general loss functions like cross-entropy or dice score to get superior results compared to general loss functions used alone.
- The loss function can help in faster convergence

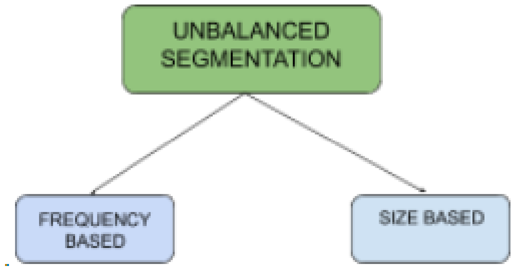
So in this article, we will discuss different challenges in segmentation and how different research papers, given their end goal tried to modify existing loss function or start from scratch devising their loss function, to get superior results. Also, we will try to classify the proposed loss function into three different categories, Boundary-based, Region-based and Topology preserving.

Challenges in segmentation

- Hard to collect data
    - The ground truth data is labelled by annotators, who mark the segmentation boundary of different object instances.

    - Accurate boundaries are important in segmentation, but it is expensive and time consuming to manually annotate accurate boundaries. We will discuss some of the methods on how to get accurate boundaries later in the article

- Unbalanced dataset segmentation



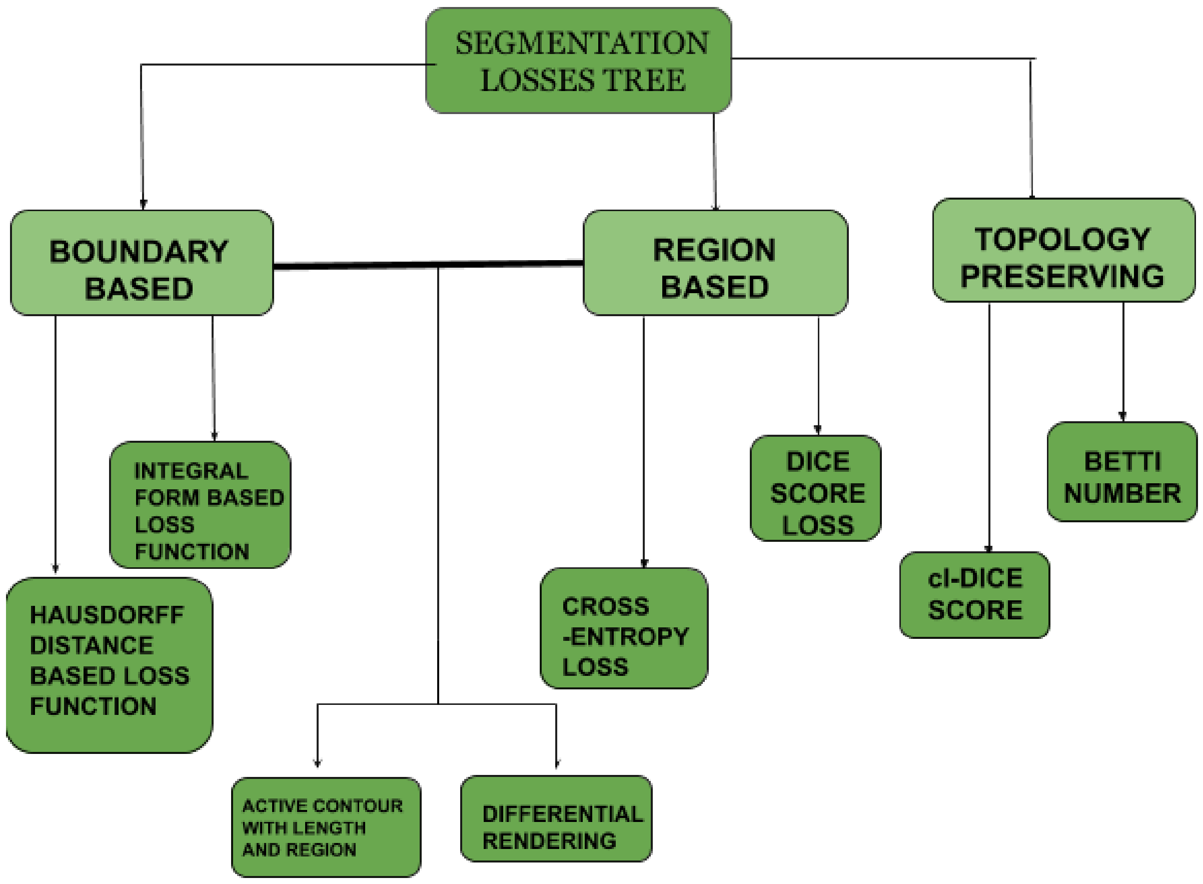
    - Unbalanced dataset can be of two types, frequency based or size based
        - Frequency based unbalanced, when an object instance is more pronounced or dominating compared to other instances, we have a frequency imbalance.
            - We pass a mini-batch to our network, because it is computationally efficient than passing individual pixels, and then we take the average of all the errors in the minibatch and the error backpropagates to update our gradients.
                - In case of frequency imbalance, we have our weight updates biased to high frequency class. The network learns that it is safer to predict the high frequency class because it will give higher accuracy.
            - Work-arounds won't help us like,
                - Assigning weights to classes inversely proportional to frequency of the class, may lead to noise amplification. Medical images which are merely CT scans are noisy and weights would exacerbate the problem.
                - Down-sampling high frequency class may lead to loss of information

        - Size based unbalanced dataset, when an object instance is disproportionately smaller than other instances, we have size imbalance. Even if they have same frequency, the results can be skewed to bigger size instances.

- Per-pixel accuracy
    - Passing a batch of pixel to network is computationally efficient and it helps the network to learn about relative positions of different objects which conserves localization. But as we take average of errors from individual pixels, so some pixel level error is tolerable thus compromising per-pixel accuracy
    - But if we have pixel level changes, then pixel level error is intolerable and we need topology preserving models.

As we have seen in our introduction section, the importance of loss function. Loss function provides a platform (in deep learning terminology, an objective function) which helps the network to minimize its inaccurate predictions. Choosing a pertinent loss function, can help you in getting accurate results even with small number of parameters, thus decrease computational burden and improve performance. So, now we will discuss different segmentation loss function used in state-of-the-art models.

Segmentation Losses Tree diagram



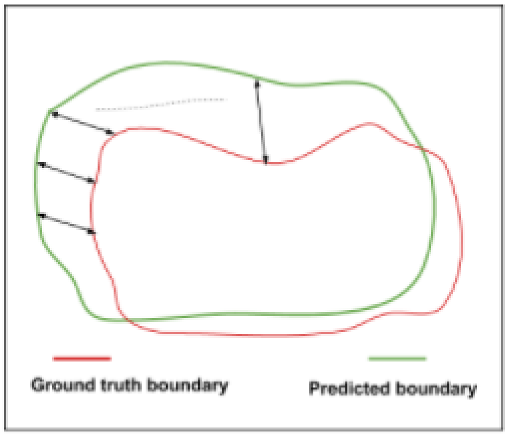
We will start with boundary based loss function

BOUNDARY BASED LOSS FUNCTIONS

HAUSDORFF DISTANCE BASED LOSS FUNCTION

- Hausdorff distance (HD) is the maximum among the minimum distances between two curves. It is the highest segmentation error. It is used for benchmarking segmentation results and it is a well-known evaluation metric.
- HD is sensitive to outliers and noise. Thus it is unstable while training, so we will discuss three HD "inspired" loss function where we mildly penalize the HD based loss function. The main idea is to estimate the HD in an indirect way and then try to minimize the error.
- **METHODS**
    - Estimation of HD based on distance transform

    - Distance transform is the vectorial representation of Euclidean distance between two curves. The Euclidean distance is defined in the non-overlapping regions between two closed curves. Here two curves mean, predicted segment curve and ground truth curve.
    - So the main intuition is to extract the binary maps (foreground is 1 and background is 0) of predicted and ground truth curves, then find out the non-overlapping regions between the curves and try to minimize the distance in this non-overlapping region
    - The double arrow lines in the diagram below, represents the minimum distance between two boundaries.



    - The final loss function is given by
        - $Loss_{DT}(p,q) = \sum (p-q)^2 \circ (d_p^\alpha + d_q^\alpha)$
            - where DT is distance transform based estimation of HD, p is the ground truth label map and q is the predicted label map. $(p-q)^2$ denotes the non-overlapping region. The loss function is calculated over the entire image domain
            - Calculating d_q is computationally intensive as it changes in each iteration, so we can ignore the d_q in the final loss function, this will help in faster convergence
- <u>Estimation of HD by morphological operation</u>
    - Here we use morphological erosion

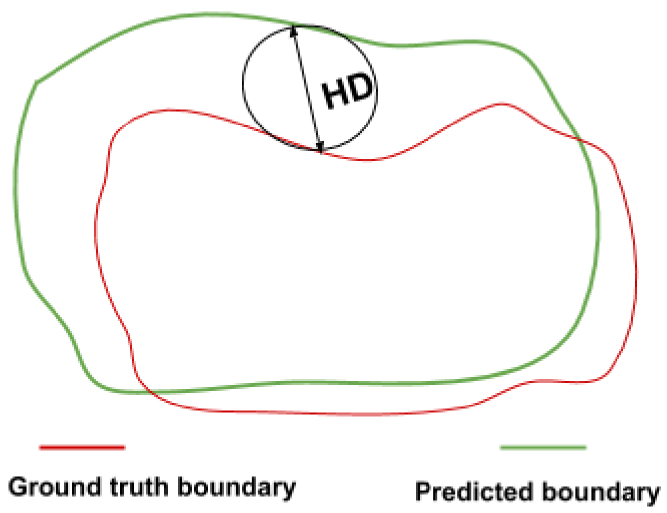
    - In morphological erosion, we take a structuring element centred at the middle and traverse it through the grayscale image. If the structuring lies entirely inside our foreground then only the centred pixel corresponding foreground pixel stays foreground (logical 1), else it is marked as background (logical 0).

  - So the main intuition is to find non-overlapping region (which is given by the symmetric difference between ground truth and predicted segment), apply successive morphological erosion by a circular structuring element in these region to erode away the non-overlapping region.

    - **So how does HD comes into play here?**

      - We talked about circular structuring element which is of radius r^*

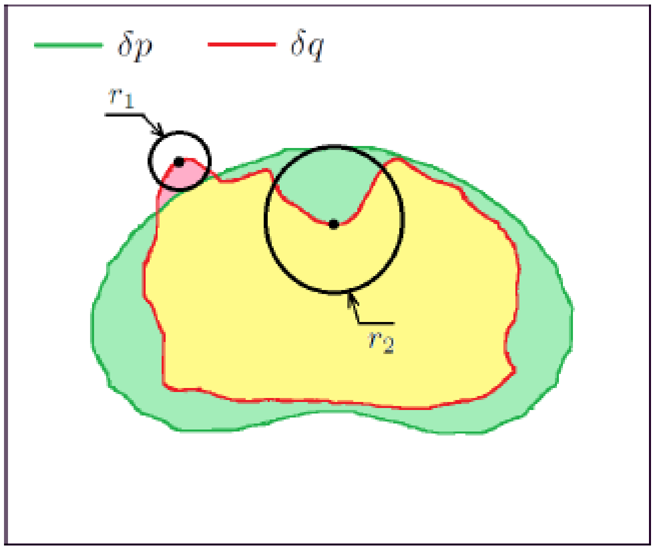
      - If $2r^*$ is more than HD then there is no possibility that the structuring element will lie inside the non-overlapping region, thus it will erode away the non-overlapping region.



      - So under the hood, the loss function is trying to estimate the HD, so that by morphological erosion operation, it can erode away the non-overlapping region.

- Estimation of HD by circular convolutional kernel

- Here we have a circular-shaped convolutional kernel B_r, where r is the radius of the kernel. We have a discrete set of radius R, we sequentially try out each radius.



Taken from the paper, Reducing the Hausdorff Distance in Medical Image segmentation in CNN

- In the figure, δp is the ground truth boundary and δq is the predicted boundary.
- As in the figure, among r_1 and r_2, r_2 is an accurate estimation of HD. So the loss function tries to estimate the radius of the circular convolutional kernel and then tries to minimize the loss.

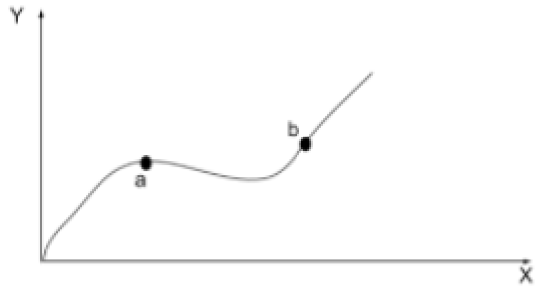
○ COMPARISON BETWEEN LOSS FUNCTIONS

- Here CV is HD loss function by circular convolution, ER is by morphological operation and DT is by distance transform and DSC is dice loss.
- According to the time required for training
  - $L_{DSC} < L_{ER} < L_{CV} < L_{DT}$
- According to performance
  - $L_{DSC} < L_{ER} < L_{CV} < L_{DT}$
- For 2D images calculating the distance transform is not a heavy load so it is chosen over other two as it gives more accurate results whereas for 3D

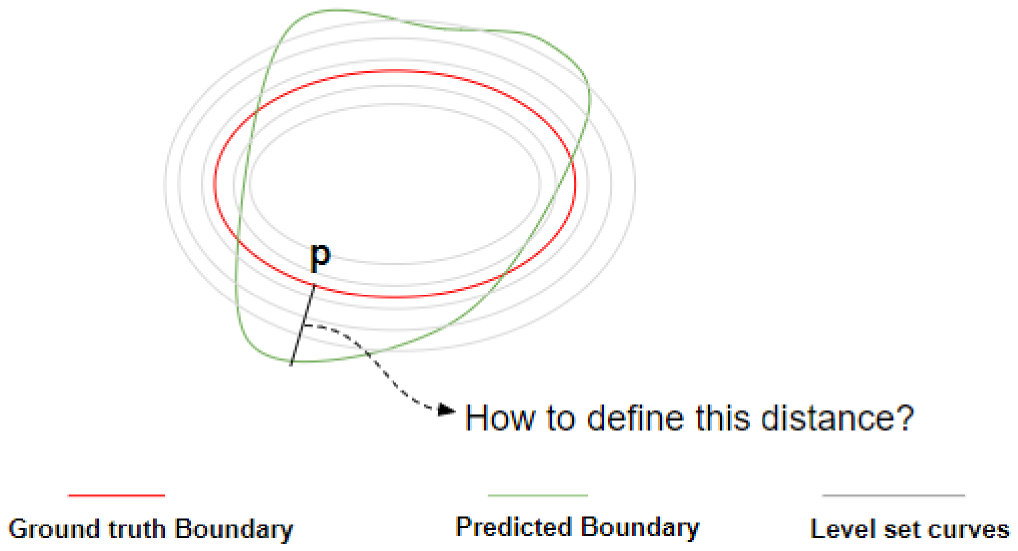
images it is difficult to compute the distance transform so CV or ER based loss function is chosen. CV based loss function gives better result if we limit the maximum kernel size (radius R).

INTEGRAL FORM BASED LOSS FUNCTION

- How do you find the distance between two points lying on a function $f(x)$?
    - We use integration here, suppose we are traversing from point a to point b on $f(x)$.



    - Suppose in the above function $f(x)$, we want to find the distance between a and b. Imagine, jumping dx distance every time along the function $f(x)$. If we sum all the $dx's$ from a to b, then we will get the the total distance. This is given by the definite integral
        - $\int_a^b f(x)dx$
    - Utilizing this theory, we will try to define an integral form of distance based loss function between ground truth boundary and predicted boundary. But here we have one caveat, a and b in the above example belong to the same function, but how to find distance between two points which belong to different function.
    - So here comes the idea of level sets
        - Level sets are the family of curves which have the same constant value in the domain of $\mathbb{R}$. So $x^2 + y^2 = k$ represents a family of circles with the same underlying function, $x^2 + y^2$ and we have a varying parameter k.
        - So here, we generate the level sets of the ground truth boundary and find the distance between two boundaries.



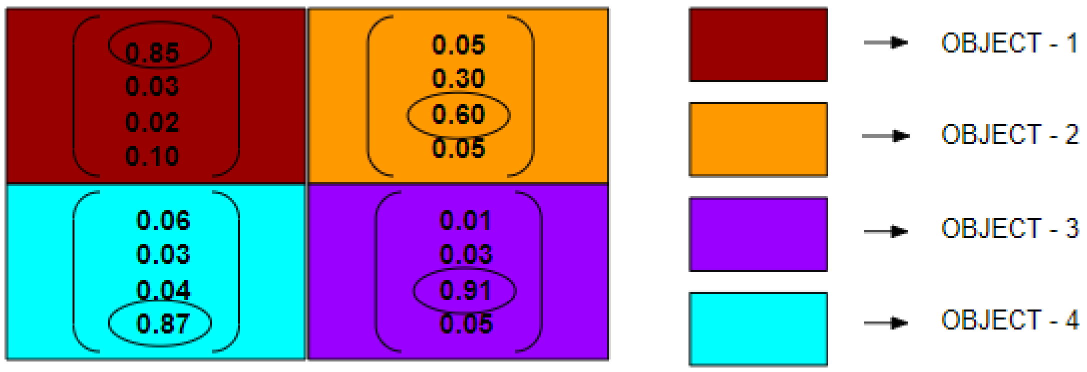
- The general intuition is, we take a point p (a point defined by the ground truth boundary function) on the ground truth and along the normal find a point on predicted boundary, $y_{\delta s}(p)$. So the MSE distance function is given by
  - $||y_{\delta s}(p) - p||^2$, from here we can either use gradients by differentiating the loss function or we can follow the integral form with a simple tweak
    - $\int_p^{y_{\delta s}(p)} 2D_G(q)dq$ which is equal to the $||y_{\delta s}(p) - p||^2$
    - What is q here?
      - We have generated the family of curves corresponding to ground truth, imagine "jumping" through these curves to get to the point on predicted segment. The level set curves will guide us in "jumping".
      - q is the independent variable in the level sets (like x is independent variable in $f(x)$)
- So the distance is given by
  - $\frac{1}{2} * Dist(dG, dS) = \int_\Omega \phi_G(q)s(q)dq - \int_\Omega \phi_G(q)g(q)dq$
    - Here

        - G is the ground truth boundary, $g(q)$ are the ground truth probabilities (it is grayscale image so it is logical 1 for foreground and logical 0 for background)
        - S is the predicted boundary, $s(q)$ are predicted segmentation probabilities
        - $\phi_G(q)$ are the level set curves
        - Ω is the entire image domain
        - So we are weighing the distance with the predicted segmentation probabilities, so that we can decrease the distance error and the likelihood error.
    - The main takeaway is, it can be used for SIZE-BASED UNBALANACED DATASET, as it is concerned with boundary of the object of interest.

REGION BASED LOSS FUNCTION

CROSS-ENTROPY LOSS

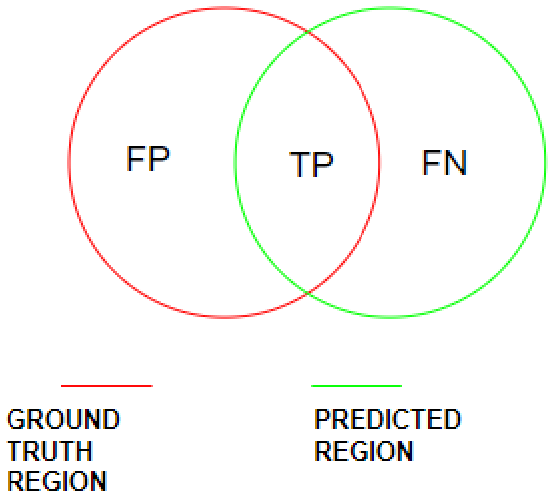
- Cross-entropy loss is the sequential combination of log-SoftMax and NLL (negative log-likelihood loss).
- Log-SoftMax operation is applied on the values of the output layer and then negative likelihood loss is computed to one-hot encoded labels of each pixel
    - The one-hot encoded value of each pixel in ground truth is of dimension (#object_instances,1), where the argument corresponding to the object class has the value 1, else 0



- It does not perform well in both frequency and size based unbalanced segmentation and the reasons are discussed in the section challenges to segmentation.

DICE SCORE LOSS

- Dice score loss is the IoU (intersection over union) of two regions. The two regions here are, ground truth label and predicted segment region.



- In the above diagram, FP means False positive, TP means True positive and FN means False Negative. So the dice score loss function is given by

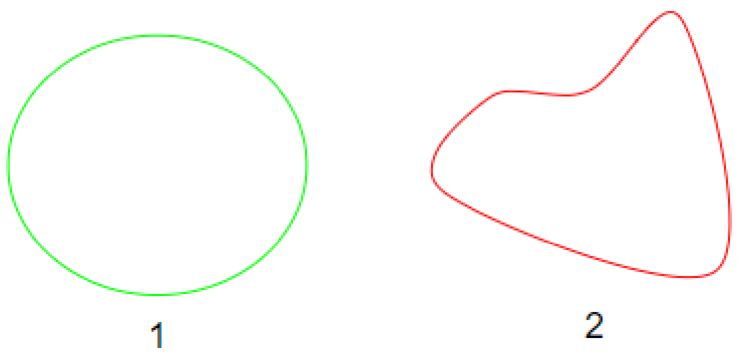
$$1 - \frac{2 * TP}{2 * TP + FP + FN}$$

- It is a foreground-region-centric loss function, so it performs well in frequency and size based unbalanced dataset. It is concerned about the foreground spatial region, also it penalizes both FP and FN, so it performs better than cross-entropy loss in unbalanced dataset.
- It can also be written as $\frac{2*pt}{p+t}$, where p is the predicted segmentation probability and t is the ground truth probability. We can see a problem here, the differential form of the function is
    - $\frac{2t^2}{(p+t)^2}$, so if p is very small then the gradients may shoot up due to the squared term in the denominator. So it leads to unstable training

COMBINED REGION - BOUNDARY LOSS FUNCTION

ACTIVE CONTOUR WITHOUT EDGES (LENGTH AND REGION)

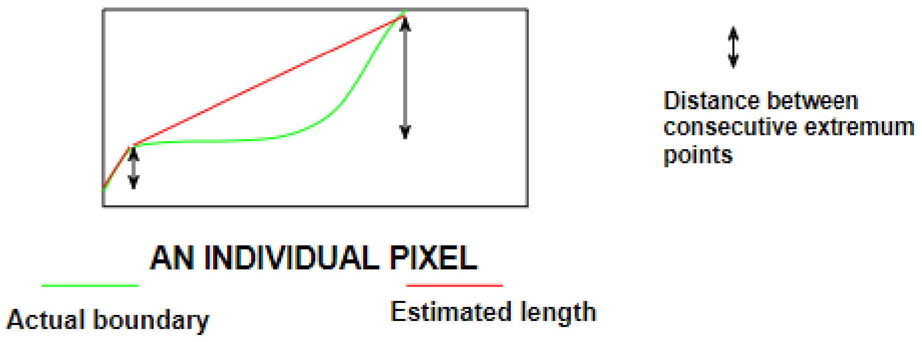
- Active contour without edges are different from active contour or curve evolution models
    - Curve evolution is an energy minimization method, where we have internal (shape of the curve) and external energy (image gradients) differential equations
    - Internal energy has first-order and second-order derivative terms, where the first order takes care of the stretchiness of the curve and the second-order takes care of the bendiness of the curve.
        - Stretchiness describes whether the points on the curve are uniformly placed or not. First-order derivative terms have $f(x + \delta x) - f(x)$ in the numerator, if the points are uniformly distributed then the numerator would be less, hence the first-order derivative would be less
        - Similarly, if the second-order derivative is less, then the curve has fewer twists and turns (bendiness).



Curve 1 has lower internal energy than curve 2

- External energy has image edges or gradients, so it helps the curve finally evolve to fit the boundary of object.
- The problem here is, in the process of minimization of energy of the curve, it may be stuck in local minima and give sub-optimal results.

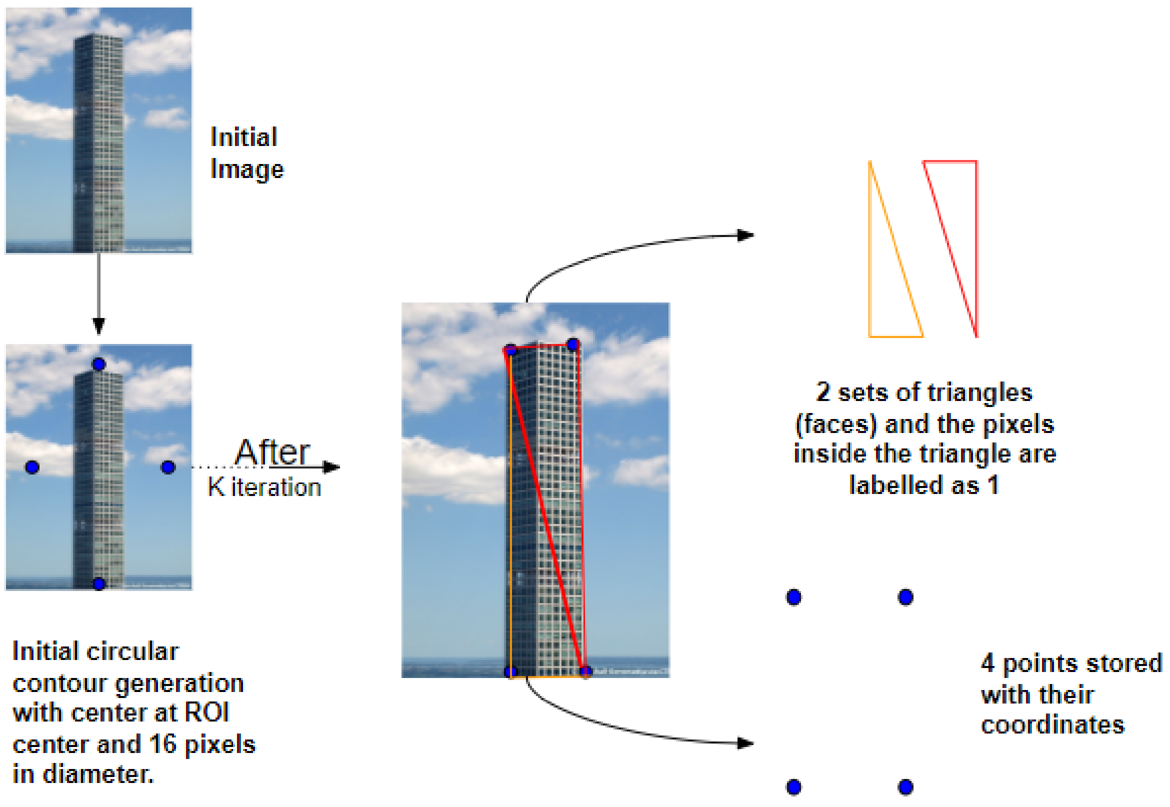
- So here we choose the length and regional area of our object of interest. As we are taking the length and area of the image, the model will be robust to noise.
  - Also the model is **dataset-specific**, the authors trained it on heart images so it is very accurate at segmenting heart images
- The methodology is
  - For length, the estimated length of boundary can be given as hypotenuse length of the horizontal and vertical projection. It is like moving along the curve and finding the length of y-projection between two consecutive extremum points



- For region, the conventional method assigning the pixel either to foreground or to the background method is used. The foreground and background of ground truth is labelled as logical 1 and 0 respectively.
- The final loss function is combination of length and regional loss function.

DIFFERENTIABLE RENDERING

- In active contour models we have a curve initialization, but curve are not so good at fitting straight edges like skyscrapers. This method applies well in images where we have straight edges.
- Every shape (2D or 3D) can be made of triangles, like a rectangle can be made of two right angled triangles. So the methodology is as follows


Initial Image
After K iteration
Initial circular contour generation with center at ROI center and 16 pixels in diameter.
2 sets of triangles (faces) and the pixels inside the triangle are labelled as 1
4 points stored with their coordinates

- The methodology is
  - We initialise a circular contour centred at the ROI's centre and of diameter 16 pixels and number of points are chosen so that we get proper boundary

edge meanwhile conserving computation. It requires no user-intervention.

- Then it is passed through an encoder and decoder layer, which outputs a displacement field J. J, helps in updating the initialised points towards the boundary of ROI
    - It gives the displacement vector along x and y direction of each point.
- In each iteration the points are displaced according to the displacement field to cover the boundary of the region
    - $p_j^t = p_j^{t-1} + J(p_j^{t-1})$, here p_j means the vertices of polygon and t^{th} iteration
    - J guides the points to the object boundary.
- After K-iteration, with the points we follow Delaunay triangulation to generate triangular faces
    - Delaunay triangulation is forming set of triangles from discrete points in a plane so that no point is present inside the circumcircle of a triangle.
    - The pixels inside the triangle (or the faces) are labelled as 1 (foreground) and others are labelled as 0. This is how you get predicted segment binary map.

- The loss function is MSE distance loss between predicted contour and ground truth contour. This loss function is combined to balloon term, which helps the initial contour to expand or "balloon" if it is initialised inside the object, and a curvature reduction term, which takes the average between two points to reduce curvature.

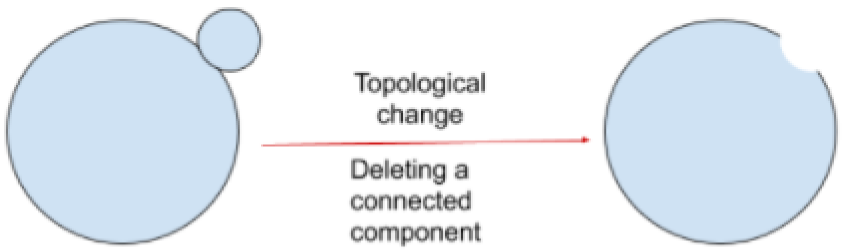
TOPOLOGY PRESERVING LOSS FUNCTIONS

- We expressed our challenges for per-pixel accuracy in challenges to segmentation. Topology preserving loss function can help ameliorate this problem.

cl-DICE SCORE (Centre-Line in mask)

- Blood vessels are very thin and tubular structures, have pixel level changes. So to conserve pixel-level accuracy we use a modification of Dice score, called as centre-line in mask dice coefficient. Another application is in road-networks detection

- The main intuition if you can ensure that your predicted segment does not have any ghosts or misses compared to ground truth, you can ensure topological correctness
    - Another key point is, here we are talking about combinatorial topology. The topology that is concerned with vertices, edges and faces.
        - Here Euler number was used as evaluation metric, which is equal to $V - E + F$, where V, E, and F are the number of vertices, edges, and faces of an object
    - If we can ensure there are no misses in the foreground after the topological change (after each iteration), then we can conserve topology.



- We have predicted (V_P) and ground truth (V_L) binary masks and then we generate skeletons of the two masks, named S_P and S_L.
    - Skeletonization is an iterative min-max pooling layer so that we can include our skeletonization layer in our CNN architecture.
    - Now we will go into the mathematical formulation
        - $T_{prec}(S_P, V_L) = \frac{|S_P \cap V_L|}{|S_P|}$,and it tells what fraction of S_P lies within V_L
            - Precision takes FP or ghosts into account, the places where the model thinks the segmented object to be present but it is not in the ground truth. So T_{prec} penalizes FP
        - $T_{sens}(S_L, V_P) = \frac{|S_L \cap V_P|}{|S_L|}$, it tells what fraction of S_L lies within V_P

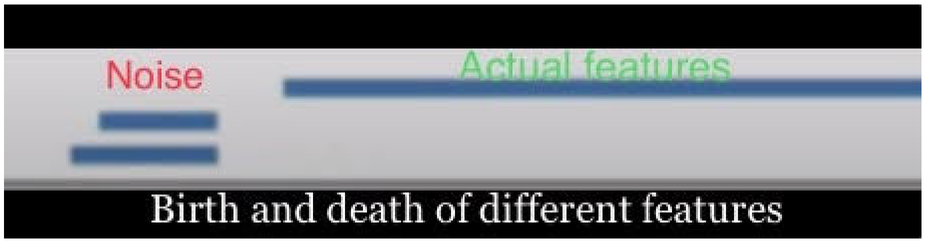
    - Sensitivity or recall takes FN or misses into account, the places where the model didn't predict anything, but we have some part of object according to the ground truth. So T_{sens} penalizes FN
  - **The IoU score between T_{prec} and T_{sens} is called as cl-Dice (centre-line) score** and the associated loss function is $1 - clDicescore$. This loss function is combined dice score function as final loss function expression.

TOPOLOGY PRESERVING BY USING BETTI NUMBER

- In segmentation, our main idea is to generate predicted segment similarly shaped to the ground truth. Geometrical equivalence which means to ensure we have same length and angle is a computational overload given that out predicted segment changes in each iteration. So we look for other ways of equivalence.
- Betti number is used to distinguish between two topological structures, like number of connected components ($b0$), 2D holes or connecting handles ($b1$), 3D holes or cavities ($b2$) and so on. Betti number helps in establishing topological equivalence.
  - Two structures are topologically equivalent, if they can be deformed into one another by bending, twisting, stretching and shrinking.
  - Like a square and circle have same topology, they both divide the space into two parts (inner and outer) and a square can be deformed into a circle and vice versa. They both have $b0 = 1$ and $b1 = 1$, so they belong to same homology class.
  - Topological equivalence is less computationally demanding compared to geometric equivalence.
  - But directly comparing betti number is an discrete operation, so we need a continuous process which can be end-to-end trainable.

- Our main intuition here is, if we vary the threshold of the image then some features will die and some features will be born
  - Features means connected components and handles.

    - The features that have longer survival time or we can say the features that "persists" for longer time are true features else they are noise. This is known as persistence homology



    - We compare this birth and death time of the features in predicted and ground truth
- METHOD
    - We vary the threshold of the predicted and ground truth from 0.0 to 1.0
    - The features that stay for longer duration are plotted as persistent dots on persistence diagram where we have x-axis as (1-birth_time) and y-axis as (1-death_time). This is done for both ground truth and predicted segment
    - Then we take MSE loss between the persistence dots of ground truth and predicted segment. Also we take BCE loss in the final loss function. The topological loss is more accurate on pixels compared to BCE loss.
        - The Betti number derived loss function is computationally expensive and performs well on critical pixels (the pixels which are hard to segment). So at initial part of training, we use BCE loss function which helps in regional classification of foreground and at later stage of training, we use Betti number derived loss function.
        - It is like saving your expensive and players who can perform better in critical games, for later part of the tournament.

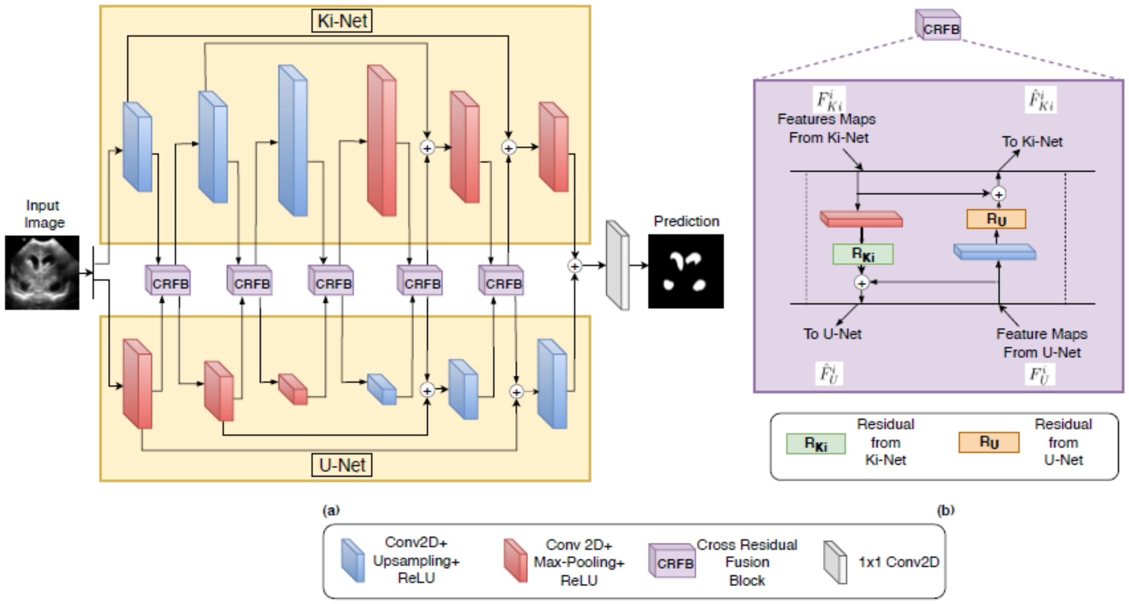
ARCHITECTURAL CHANGES TO DETECT SMALLER REGIONS

- So far we have talked about carefully choosing loss function to fit our purpose like to preserve topology or for unbalanced datasets. But CNN architectures is another component in the deep learning framework, which can be modified to fit our purpose. So we will discuss about KiU-Net,

KiU-Net

- In the standard CNN architectures like U-Net we have encoder and decoder layer. Encoder layer or top-down model, has bunch of convolutional layer followed by max-pooling layers and the decoder or down-top model has up-sampling layers to make up for the decrease in dimension due to max-pooling layers.
- Max-pooling layers don't have any parameters to learn, they reduce the dimension of feature map so that the network only concentrates on highlighting or dominating features.
- Max-pooling layers leads to increase in receptive field size
    - Receptive fields can be thought of as the area that the kernel concentrates on. So receptive field is the size of the input which produces one node in the feature map.
    - As our dimension of input image decreases, the subsequent kernels in our network implicitly concentrates on a larger area on the input image.
    - Suppose you are training on MNIST dataset where you have 28x28 input image and kernel size of 3x3. After the first max-pooling layer the dimension reduces to 14x14.
    - So now a 3x3 kernel would concentrate on 9 pixels on 14x14 feature map, implicitly concentrating on 18 pixels on the input image (28x28). Hence, there is increase receptive field size
    - Increasing receptive field size leads to missing smaller regions
- Hence we go for some architectural changes to detect smaller region. As the basic intuition, if max-pooling increases our receptive field size then we need up-sampling layers in the encoder layer which will constrict the receptive field size.
    - And to compensate for this increase in dimension, we will have max-pooling layers in the decoder layer. This is the Ki-Net architecture.

- **So KiU-Net is actually Ki-Net + U-Net**
    - They output feature map of each layer in both networks are combined by CRFB (Cross-Residual Fusion block).
    - CRFB up-samples from U-Net layer and then concatenates with Ki-Net. Similarly, it downsamples by max-pooling Ki-Net layer and then concatenates with U-Net


Ki-Net
U-Net
Input Image
CRFB
Prediction
F^i_{Ki} Features Maps From Ki-Net
$\hat{F}^i_{Ki}$ To Ki-Net
R_U
R_{Ki}
To U-Net $\hat{F}^i_U$
Feature Maps From U-Net F^i_U
R_{Ki} Residual from Ki-Net
R_U Residual from U-Net
(a)
(b)
Conv2D+ Upsampling+ ReLU
Conv 2D+ Max-Pooling+ ReLU
Cross Residual Fusion Block
1x1 Conv2D

Taken from the paper KiU-Net : Towards Accurate segmentation of bio-medical images using over-complete representations.

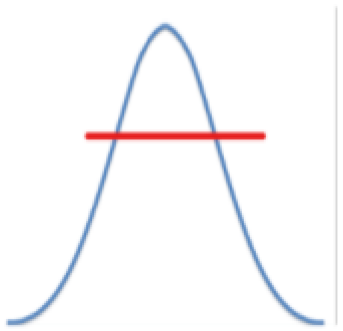
- KiU-Net requires less number of parameters, as it does not to be deep to detect smaller and larger featues. Hence it has faster convergence with superior results.
- Ki-Net learns smaller features and U-Net learns larger features and they share their knowledge with each other through CRFB layers. IT IS LIKE COMBINING THE BEST OF BOTH WORLDS.

SOME DATA ANNOTATION METHODS

- We discussed under challenges in segmentation, that data annotation is the bottleneck in segmentation problems. It requires intensive man-hours to annotate boundaries accurately. Noisy annotation can lead to sub-optimal results
- So you maybe in two situation
  - You have labelled data but it is noisy and inaccurate. Annotators go for thicker annotation boundaries so that they don't need to worry about precise boundaries.
  - You haven't started labelling

STEAL (Semantically Thinned Edge Alignment Learning)

- If you are in situation - 1 then you can opt for STEAL. It is an add-on in an end-to-end detector model which helps to reduce the thickness of the boundaries.The problems with thick boundaries are
  - It can lead to overlapping boundaries with other objects
  - Network cannot generate accurate boundaries, if it is trained on thick boundaries
- STEAL has 3 components in loss
  - NMS (Non-Max suppression) loss. You can see below that we have a thick edge and we move in the normal direction to the edge (denoted by red line). As we move along the red line the intensity of the edge increase
    - So we keep track of the intensity and clip it, if it exceeds our threshold. It thins our boundary

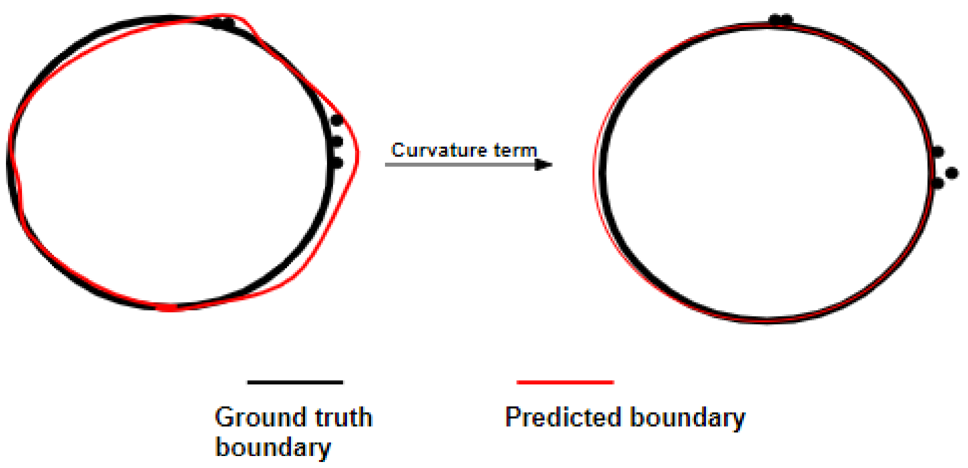


- BCE loss, is the standard cross-entropy loss to detect segmentation probabilities of pixels
- Direction loss, it is the similarity loss between normal of ground-truth boundary pixel and normal of predicted boundary.
- STEAL helps our model to learn from noisy annotations, as it sort-of does the pre-processing of the noisy annotations to get accurate ground truth boundaries. Thus you can spend less man-hours in annotation and let the network handle it.

DELSE (DEEP EXTREME LEVEL-SET EVOLUTION)

- If you are in situtation-2, then you can opt for DELSE. DELSE uses curve evolution to find boundaries (we discussed curve evolution in Active contour without edges)
- Here the annotators place 4-points around the object of interest, from these 4 points a curve is initialized. There are 3 DELSE components which guide the curve to the object boundary.
    - Motion term
        - It takes care of the velocity vector of each pixel in predicted boundary, we define a unit velocity vector which keeps track of the change in gradient of the level set curve along x and y direction
    - Curvature terms

- It helps the curve to regularize, by regularize we mean that the final curve is displaced slightly towards the true object boundary, so that it does not overfit to noise around the object as we can see in the below diagram. But the regularization needs to be modulated so that the network fits the straight edges properly.



▪ Regularization term

- It is a stability loss function that tries to minimize irregularities and numerical errors in the final result

CONCLUSION

In the early stages of development in CNN's, people tried to increase the number of parameters and built deep architectures to get superior results. This demands a lot of computational power. But carefully devising a loss function, can help us to get superior results with fewer parameters. Also, hand-engineering loss functions require expertise from different fields like topology etc. So it can bring people from different fields under one umbrella, working for advancement in Artificial intelligence