Semantic Edge Detection

Akshi Sharma
DIBRIS
Data science and Engineering
Artificial Intelligence
University of Genoa

***Abstract*—Semantic edge detection is a challenging multi-label problem. The aim is to be able to detect visually predominant edges and also recognize their categories. Boundary and edge cues are highly beneficial in improving a variety of vision tasks such as semantic segmentation, object recognition, stereo, and object proposal generation[2]. Semantic edge detection has far reaching applications in multiple concepts of computational vision. This process of simultaneous extraction of edges as well as their respective categories is complex which makes it difficult to be precise. The issues range from an edge being shared by multiple categories which can lead to ambiguous or multiple labels[2] to having flaws in annotations that lead to noisy labels[1]. Several state-of-the-art methods have been developed in an effort to overcome the shortcomings and to find techniques that can help in effectively improving the results. The aim of this paper is to understand Semantic edge detection and the methodologies being used to implement and subsequently improve it.**

I. Introduction

Most of the shape information of an image can be captured by highlighting the edges. Edge detection can be done by identifying points in a digital image with discontinuities, such as sharp changes in the image brightness, which is basically what defines edges or boundaries. But classical edge detection is a binary classification problem where each pixel of the image can either belong to the edge class or non-edge class. Semantic edge detection is a deeper dive into detecting what is the category of each edge class pixel. The basic instinct is that all pixels that fall in the edge class are not the same. So semantic edge detection involves two steps. One to find the edges in the image and second to figure out from all the objects in the image the category of the edge pixel. It can be seen as a dual task to image segmentation which identifies object regions[1].

II. About the problem

Though it sounds simple there are some rational constraints that need to be encountered in order to make semantic edge detection more effective and precise. The objects in the images may share boundaries which makes it harder to define a specific category for the edge pixel[2]. Also the annotations can contain a lot of label noise as it takes longer to define precise boundaries. This in turn causes the predicted edges to be very thick as the labels defined in the training data are misaligned[1]. Most of the images will have a huge difference in the number of edge pixels and Non-edge pixels which creates an imbalance in positive and negative samples. This too leads to thicker edges in the prediction due to false positives. Solutions to these issues have been suggested in CASENet[2] and STEAL[1]. The STEAL approach was to create an extension to existing detectors by adding a new layer and loss in order to get better results. CASENet[2] being the state-of-the-art prior to STEAL[1] was used as the backbone method.

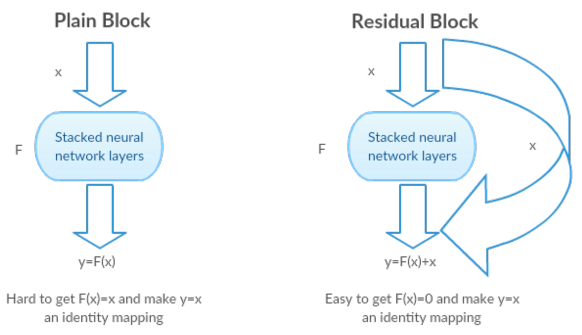
Plain Block
x
F
Stacked neural network layers
y=F(x)
Hard to get F(x)=x and make y=x an identity mapping
Residual Block
x
F
Stacked neural network layers
x
y=F(x)+x
Easy to get F(x)=0 and make y=x an identity mapping

Fig. 1. Residual block Mapping

III. Approaches

The aforementioned problems were tackled by using concepts of two techniques described in different papers. Since STEAL[1] uses CASENet[2] as the basis for their methodology the architecture of other related networks need introduction.

A. RESNET

ResNet[5], short for Residual Network, provided the capabilities to tackle vanishing/ exploding gradient issue and create Deep Neural Networks. The core idea of ResNet is to introduce “identity shortcut connection” or ”residual connection” that skips one or more layers. The authors of [5] hypothesize that fitting a residual mapping is easier as compared to underlying mapping. In other words it is simpler to come up with a solution like F(x) =0 rather than F(x)=x as it is easier for the Residual block to learn the Identity function. ”Fig. 1”. Function F(x) is what the authors called Residual function. This also allows each layer to see more than just its previous layer’s observations and get access to extra information.

B. CASENet

The authors of CASENet[2],Category Aware Semantic Edge Detection, focus on the issue of identifying multiple classes of

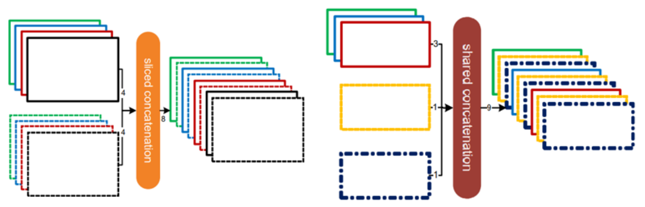

Fig. 2. Sliced and Shared Concatenation module details

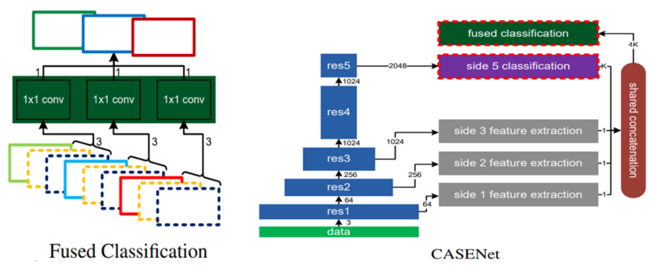

Fig. 3. Casenet architecture and Fused classification module details

objects in edge detection. They bring forth the rationale that an edge pixel could be shared by multiple semantic categories in a real world scenario. For given 'K' defined semantic categories, each edge pixel could belong to any of the categories. To identify these categories the network essentially produces 'K' separate edge maps where each map indicates the probability of the edge to fall in a certain category[2].

a) CASENet Architecture: CASENet uses the fully convolutional network framework by adopting ResNet-101 and modifying it. The authors removed the original average pooling and fully connected layer but kept the bottom convolution blocks. The main idea followed by the authors is that early on in the network the layers do not contain high level features and therefore any form of classification task is not very beneficial and does little in terms of improving the efficiency of the network. They adopt the nested architecture similar to DSN but propose the idea of using side feature extractor modules instead of side classification modules in the initial layers of the network. This helps in better preserving the low-level edge information. This information is helpful in augmenting top classifications, suppressing non-edge pixels and providing detailed edge localization and structure information[2]. They used the side classification module only at the top layer of the network. The side feature extraction module gives as output a single channel feature map. Whereas the side classification module gives activation map with 'K' channels as output. They also replace the sliced concatenation with shared concatenation process. In sliced concatenation, same index channels from each side are fused together(mathematically Equation 1).

$$A_f = A^1_{(1)}, ..., A^1_{(5)}, A^2_{(1)}, ..., A^2_{(5)}, ..., A^K_{(5)} \tag{1}$$

On the other hand shared concatenation separately concatenates with each of the K top activations(mathematically Equation 2). Bottom features in F from 3 sides are $F = F^{(1)}, F^{(2)}, F^{(3)}$

$$A_f = F, A^1_{(5)}, F, A^2_{(5)}, F, A^3_{(5)}, ..., F, A^K_{(5)} \tag{2}$$

The approach behind shared concatenation is that for each feature map in K maps, the first three stage output from feature extraction modules are queued in, so the final feature map channel is 4xK. The resulting concatenated activation map is fed into the fused classification layer with K-grouped convolution. This means every map does group convolution,four maps, one from advanced features(classification module), the remaining three from low-level characteristics(feature extraction module). It produces a K-channel activation map.As stated by the authors[2] CASENet can be thought of as a joint edge detection and classification network by letting lower level features participating and augmenting higher level semantic classification through a skip-layer architecture.

C. SEAL

Simultaneous Edge Alignment and Learning[4] is a framework that deals with misaligned labels and noisy annotations. They formulated the problem with a probabilistic model. The method involved assuming the ground truth edge labels as latent variables which can be jointly learned during training. The optimization of latent edge labels is transformed into a bipartite graph min-cost assignment problem, and present an end-to-end learning framework towards model training. They used Biased Gaussian kernel and Markov prior to make sure the edge re-alignment does not go astray due to background clutter.The method was computationally expensive and also does not work in cases where the difference between ground truth is due to topology.

D. Level Set Methods

Level set methods are used to solve topology change problems during the evolution of curves. Most algorithms cannot perform well in situations where the surface or curves are evolving like splits or merges. Level set is a numerical solution for processing topological changes of contours.[7] The basic idea is to represent the surfaces as the zero level set of a higher dimensional hyper-surface, where zero level set implies that the surface has zero height at that point. Basically closed curves are treated as an instance of a continuous surface of a three-dimensional space.Level set method, due to its stability and irrelevancy with topology, displays a great advantage in solve the problems of corner point producing, curve breaking and combing, etc.[7]

E. STEAL

Semantically Thinned Edge Alignment Learning[1] was created with the aim to tackle semantic boundary prediction and to also solve the issue of noisy annotations. Annotated object boundaries are often inaccurate and the edges detected

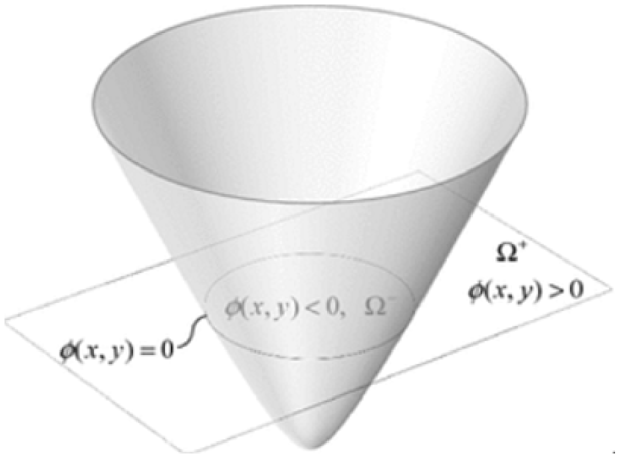

Fig. 4. Level set method

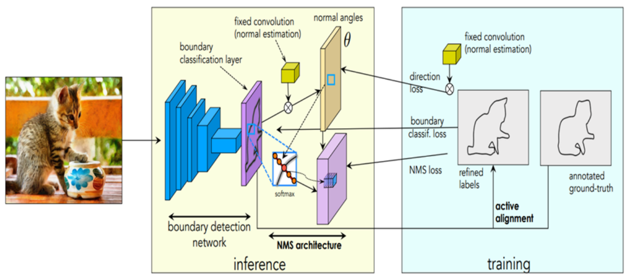

Fig. 5. Steal architecture

when learning with such data have thickness and are not very precise. The authors propose Boundary Thinning Layer to generate thin and accurate edges by introducing a new layer and loss that can be added to the top of any end-to-end edge detector. They also introduce a framework,that they refer to as Active alignment, to improve the accuracy of coarse annotation boundaries by incorporating classical boundary detection methods.Active alignment involves learning to align the noisy human labeled edges with the real boundaries during training, all the while learning the object edges as well. The intuition is that by using real boundary signals for training the boundary network will be able to learn and produce more accurate predictions.

a) STEAL Architecture: STEAL can work on any backbone architecture and essentially adds on to the capabilities of the existing network. STEAL architecture consists of two parts: Semantic edge detection and Active alignment.

Semantic edge detection is to predict boundary maps for K object classes given an input image x through maximizing the likelihood of $P(y_k|x;\theta)$,where y_k^m indicates whether pixel 'm' belongs to class 'k' Each pixel can therefore belong to multiple classes to represent the edge that borders multiple objects.

Binary Cross Entropy loss is used for standard boundary detectors. The difference between the edge pixels and non-edge pixels can be lot depending upon the image. To manage this difference weighted BCE loss is minimized

$$\mathcal{L}_{BCE}(\theta) = -\sum_k logP(y_k|x;\theta) \tag{3}$$

Thinning Layer is added on top of the boundary detection network where each pixel that falls in edge category is normalized in the direction of its normal $\vec{d}_p$ The boundary thinning layer computes the edge normals, and samples 5 locations along the normal at each boundary pixel[Fig5]. **NMS Loss** is calculated for each edge pixel along this normal.

$$\mathcal{L}_{nms}(\theta) = -\sum_k \sum_p logh_k(p|x,\theta) \tag{4}$$

Direction Loss. The normal direction of ground truth pixel $\vec{d}_p$ and the computed normal $\vec{e}_p$ should be similar[Fig5].

$$\mathcal{L}_{dir}(\theta) = \sum_k \sum_p ||cos^{-1} < \vec{d}_p, \vec{e}_p(\theta) > || \tag{5}$$

Full augmented Loss is the weighted combination of the three losses. The weights or hyper-parameters α defines the contribution of each loss. It also helps evaluate the performance if either of the losses is completely ignored.

$$\mathcal{L} = \alpha_1 \mathcal{L}_{BCE} + \alpha_2 \mathcal{L}_{nms} + \alpha_3 \mathcal{L}_{dir} \tag{6}$$

Active alignment is to find a more accurate version $\hat{y}$ of ground-truth label y, where $\hat{y} = \hat{y_1}, \hat{y_2}, ..., \hat{y_K}$

A **level set formulation** is followed to ensure that the inferred "true" boundaries remain connected, and are generally well behaved. $\hat{y}$ is defined at level set zero.

$$\hat{y_k} = \Gamma : \phi(\Gamma, t) = 0 \forall t \tag{7}$$

The aim of active alignment is to not only find a better annotation but to also ensure the edge detection model works fine as well. So the resultant loss function that needs to be minimized is

$$\underset{\hat{y},\theta}{min} \mathcal{L}(\hat{y}, \theta) = -\sum_k logP(y_k, \hat{y_k}|x;\theta) \tag{8}$$

$$\underset{\hat{y},\theta}{min} \mathcal{L}(\hat{y}, \theta) = -\sum_k (logP(y_k|\hat{y_k}) + logP(\hat{y_k}|x;\theta)) \tag{9}$$

Eq.(9) contains the re-alignment function and the edge detection function. It can be hence written as an optimizing objective function

$$\underset{\hat{y},\theta}{min} \mathcal{L}(\hat{y}, \theta) = \underset{\theta}{min} \underset{\hat{y}}{min} \mathcal{L}(\hat{y}, \theta) \tag{10}$$

The re-alignment of boundaries(first log value of Eq.(9)) is defined using an energy form which turns it into a dynamic contour model. This allows efficient tracking of the boundary or curves using the level set method.
STEAL architecture follows the following steps:

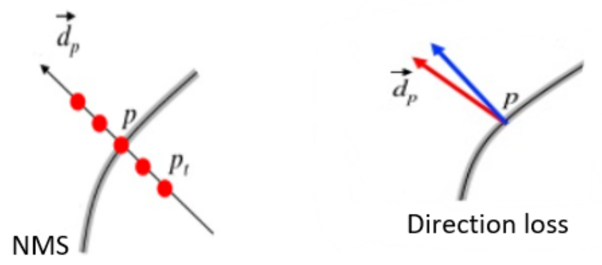

Fig. 6. Normalization along direction of normal, these values are used to calculate NMS; Direction Loss computation at pixel p

- Since STEAL is applied on a network the standard binary cross entropy loss needs to be computed for that boundary detection network (Eq.(3)).
- It assumes that the edge pixel classes are independent which may give rise to more predictions around the object boundary leading to thicker boundaries.
- To manage this boundary thinning layer is added upon boundary classification predictions which allows us to control the direction of other predictions and guide them into each boundary pixel's normal(Fig(5)).
- The responses of the boundary predictions are normalized.Inspired by edge based non maximum suppression NMS loss function is created to achieve the highest response in the direction of normal(Eq.(4)).
- NMS loss is applied only to the pixels which actually fall on the boundary or the edge.
- NMS loss introduces the normal direction and therefore the need to monitor correct direction introduces the Direction loss. It measures whether the normal direction of the predicted pixel is similar to the ground truth pixel(Eq.(5)).
- The combination of these losses forms the full augmented loss. Each loss has hyper-parameters defining its importance in the final loss calculation(Eq.(6)).
- Active alignment and learning involves optimizing another loss function which depends on θ which is the parameter for the edge detection network and $\hat{y}$ which is the improved version of ground truth label(Eq.(10)).
- Minimizing the objective function is performed with an iterative two step optimization process - keeping $\hat{y}$ fixed and optimizing θ then by keeping θ fixed while optimizing $\hat{y}$.
- In other words use the results of the boundary detection model to obtain new $\hat{y}$ and then with this new improved boundary move the boundary detection model parameter θ.
- The output of the edge detection model is unreliable at the beginning of training, which is reasonable, therefore Active Alignment is introduced after the learning of the detection model starts to converge (initially ∞).

Conclusion

Semantic edge detection problem is vastly researched and a varied set of approaches are available for not only implementing it (like CASENet) but also for improving it(like STEAL). The methodologies discussed in this paper have brought out several effective solutions to the problems faced in this field.I came across many papers during my research on semantic edge detection and that gives a glimpse into it's ever growing importance in the field of computer vision, which in itself encompasses multitudes of concepts and applications. The research into the methodologies pertaining to semantic edge detection are expected to only increase given its popularity.

References

[1] Devil is in the Edges: Learning Semantic Boundaries from Noisy Annotations David Acuna1,2,3 Amlan Kar2,3 Sanja Fidler1,2,3 1NVIDIA 2University of Toronto 3Vector Institute davidj, amlan@cs.toronto.edu, sfidler@nvidia.com

[2] CASENet: Deep Category-Aware Semantic Edge Detection Yu, Z.; Feng, C.; Liu, M.-Y.; Ramalingam, S. TR2017-100 July 2017

[3] Semantic Edge Detection with Diverse Deep Supervision Yun Liu1 · Ming-Ming Cheng1 · Deng-Ping Fan1 · Le Zhang2 · Jia-Wang Bian3 · Dacheng Tao4

[4] Z. Yu, W. Liu, Y. Zou, C. Feng, S. Ramalingam, B. Vijaya Kumar, and J. Kautz. Simultaneous edge alignment and earning. In ECCV, 2018.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, 2016.

[6] Xin Jiang, Renjie Zhang, Shengdong Nie, Image Segmentation Based on Level Set Method,Physics Procedia,Volume 33,2012,