Pattern Recognition 133 (2023) 109011

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

ScribbleNet: Efficient interactive annotation of urban city scenes for semantic segmentation

Bhavani Sambaturu[a,*], Ashutosh Gupta[b], C.V. Jawahar[a], Chetan Arora[b]

[a] International Institute of Information Technology, Hyderabad, India
[b] Indian Institute of Technology, Delhi, India

ARTICLE INFO

Article history:
Received 3 June 2021
Revised 15 March 2022
Accepted 27 August 2022
Available online 29 August 2022

2020 MSC:
05-15
34-49
Annotation
Interactive segmentation
Human-in-the-Loop

ABSTRACT

Annotation is a crucial first step in the semantic segmentation of urban images that facilitates the development of autonomous navigation systems. However, annotating complex urban images is time-consuming and challenging. It requires significant human effort making it expensive and error-prone. To reduce human effort during annotation, multiple images need to be annotated in a short time-span. In this paper, we introduce ScribbleNet, an interactive image segmentation algorithm to address this issue. Our approach provides users with a pre-segmented image that iteratively improves the segmentation using scribble as an annotation input. This method is based on conditional inference and exploits the learnt correlations in a deep neural network (DNN). ScribbleNet can: (1) work with urban city scenes captured in *unseen* environments, (2) annotate new classes not present in the training set, and (3) correct several labels at once. We compare this method with other interactive segmentation approaches on multiple datasets such as CityScapes, BDD, Mapillary Vistas, KITTI, and IDD. ScribbleNet reduces the annotation time of an image by up to 14.7 × over manual annotation and up to 5.4× over the current approaches. The algorithm is integrated into the publicly available LabelMe image annotation tool and will be released as an open-source software.

1. Introduction

Modern deep learning techniques are powerful but also extremely data-hungry. Large annotated datasets are necessary to train models using supervised learning techniques of deep neural networks (DNN). The benchmarks need to include all possible scenarios for effective evaluation of various techniques.

Manual annotation of any kind is tedious; labeling for semantic image segmentation is arguably the most time-consuming. Keeping in mind the criticality of the use case scenario, large volumes of visual data under diverse conditions have been collected globally. The human effort to accurately annotate the collected data is tremendous in time, effort and expenditure [1]. This paper aims to reduce the effort and cost involved in the annotation of such datasets.

Several automated semantic segmentation methods have been proposed to segment images into various classes [2–4]. However, due to the limited availability of annotated data in volume and diversity, their performance is restricted. This shortage of data results in sub-optimal training of semantic segmentation approaches which work in a limited set of images, and fail to generalize to new, unseen conditions [1].

Interactive segmentation approaches attempt to fill the gap by using a human in the loop to facilitate accurate annotation. Various interactive segmentation techniques such as graph-cut-based methods [5–7], random walks [8], and edge-based methods [9,10] have been proposed. In recent years, weak supervision [11] and other deep learning methods [12,13] for object segmentation and medical imaging have garnered interest. However, they suffer from drawbacks as outlined below. Typically, the aforementioned techniques are formulated as two label problems, *object versus background segmentation*. Further, these techniques are generally used for sparse scenarios, where the images contain only three to four objects and only allow correction of single label annotations at a time. The annotation of one object at a time slows down the overall annotation time of an image. However, urban city scenes typically contain several objects belonging to diverse classes captured in an uncontrolled environment.

The current approaches are typically pre-trained on a specific dataset using multiple images and annotation pairs. Training with small amounts of such data may overfit a model to suit the style of a particular set of annotators. On the other hand, while annotating new datasets, one often encounters new, unseen visual

* Corresponding author.
E-mail address: bhavani.sambaturu@research.iiit.ac.in (B. Sambaturu).

https://doi.org/10.1016/j.patcog.2022.109011
0031-3203/

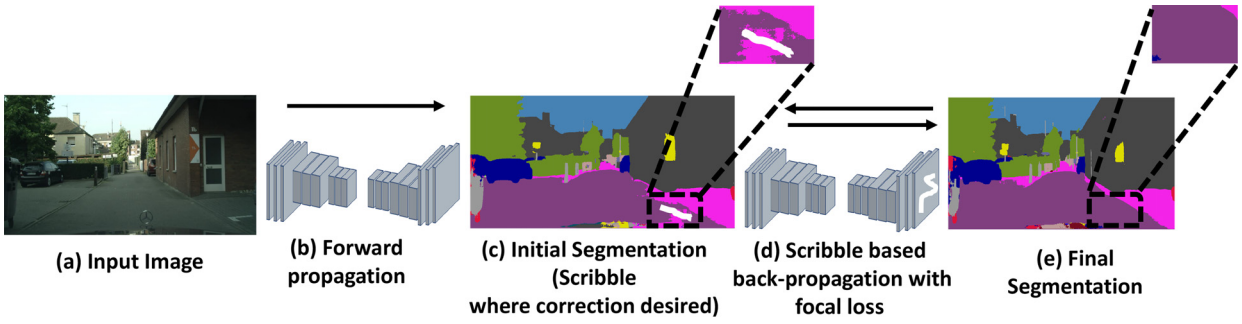

Fig. 1. ScribbleNet: The initial segmentation of an image is obtained using the pre-trained ERFNet model. The user provides scribbles in areas where correction is desired till the user is satisfied with the obtained segmentation.

appearances and backgrounds. The pre-trained approaches do not often generalize to these contexts, as a limited use-case scenario restricts their usage. Further, the new contexts may often contain new classes unseen while training. The current techniques can not handle such situations. Hence there is a need for a semi-automated annotation method that uses scenes captured in varying weather conditions with unknown objects.

This paper proposes ScribbleNet, an efficient annotation method for semantic segmentation of urban city scenes. It is based on latent feature perturbation at the test time using the user-provided scribble as depicted in Fig. 1. Latent feature perturbation involves modification of the image representation to align it with user provided scribbles. It also enables better generalization to conditions hitherto unseen in the training set [14]. This is important to facilitate annotation of images captured in novel conditions unseen during the training.

We focus on annotation for semantic segmentation tasks in the context of autonomous driving or driving assistance systems. In many of these tasks, one often needs to annotate large sequential data from consecutive frames of a video. Since we want latent feature perturbation to align the current image with the user-provided corrections, the perturbations discovered while correcting a particular frame remain relevant for the adjacent frames. The experiments using our framework prove that this method saves time while annotating video data.

It may be noted that in our method, the scribble is provided only at the test-time. This allows our method to work with any existing pre-trained deep learning technique. In contrast to the method proposed by Liew et al. [15], we do not require any additional refinement network to correct the segmentation since the existing network updates itself using the test image. It eliminates the need for scribbles during training and any additional datasets containing the user's input. This significantly reduces any preconditions to use our framework, and enhances the applicability. This also leaves the door open for our framework to potentially use any other models developed in future for semantic segmentation.

To the best of our knowledge, ours is the first paper to propose an annotation method of urban city scenes that annotates unseen classes, i.e., contexts not encountered in the training set. We greatly reduce the annotation effort up to 14.7× with respect to full human annotation and upto 5.4× over other interactive segmentation methods. We summarise the key advantages below:

- Correct several labels at the same time using the interactive framework.
- Annotates images captured in a different setting and weather conditions not present in the training set.
- Annotates new classes not present in the training set.
- Seamlessly annotates sequential video data to save time by up to 5.4× over other competitive approaches.

In the subsequent sections, we provide details of our method and experimentation to highlight the various advantages of our approach on several publicly available datasets [16–20]. We also compare our method against full human annotation and several existing state-of-the-art interactive segmentation approaches [5,8,12,13,21–25]. We also briefly describe the annotation tool developed on top of the LabelMe image annotation tool [26].

2. Related work

The generation of annotated images through interactive segmentation is a well-studied problem in computer vision. Contour-based [10] and bounding box methods [5] are among the popular methods. Scribble-based methods include a variety of algorithms, such as random walks [8], graph cut [6,7], geodesics [22,27], a combination of graph cut and geodesics [28]. In the techniques mentioned above, the object of interest is the foreground, and the rest is the background. These methods only allow corrections to a single label, leading to a longer annotation time for images with multiple objects. Recently several interactive segmentation techniques utilizing a DNN architecture [15,29] have been proposed. In this paper, we also focus on an interactive segmentation framework with a DNN backbone. We first describe the classical methods and then proceed to elaborate on the recent techniques using DNNs. The details of ScribbleNet have been given in Section. 3.

2.1. Classical approaches

The classical methods typically pose the problem as a binary segmentation problem with the object of interest segmented against its background. GrabCut [5] and Lazy Snapping [6] have formulated it as a graph-cut problem and use the min-cut/max-flow approach to obtain the segmentation. But, they only consider the appearance-based similarity between pixels and do not consider the spatial relationship. Another set of methods have been proposed to overcome this drawback; they first segment the image into superpixels and utilize user-provided cues to merge them [30,31]. However, these techniques are constrained by the initial scale of superpixel segmentation and could wrongly merge small objects into a superpixel. Random walk methods [8] obtain the segmentation by utilizing the seeded points provided by the user after mapping the image intensities to the edge weights. However, it depends on the user to provide the initial seeds for the objects. In the study by Benato et al. [32], the autoencoder utilises the user-provided points to obtain the features projected to a 2-D space using t-SNE and later projected to the other pixels. However this method also heavily depends on the position of the points provided by the user.

2.2. Deep learning-based methods

Several deep learning-based methods are proposed for interactive segmentation that exploit the DNN correlations and user-provided corrections. A combination of two convolutional networks has been proposed by Li et al. [12], where the first neural network generates a potential set of segmentations based on a user's input. The second network chooses the best possible segmentation from a set of plausible segmentations selected by the first network. Euclidean distance maps based on the user inputs are concatenated to the image [33] during the training and test time to segment the object and the background. But this technique is tightly coupled to the user inputs at the training time.

A local refinement network is combined with an existing deep learning network [15] by using images appended with Euclidean distance map to a user input. A Gaussian centered at an object's extreme points is computed to segment an image. It acts as an additional channel of the image [21]. A similar principle is used by Zhang et al. [23] but varies by providing the one point for the foreground object and two points at the two opposite extremes of the object to represent the background. Point-based inputs indicating the foreground and the background objects segment the image in IntSeg [12], BRS [29], f-BRS [13], Zheng et al. [25] and Lin et al. [34] interactively. The user-provided scribbles and a combination of minimizing the entropy and a random walk on the neural representation are used by [24] to impose consistency of predictions due to variation in scribbles. Wang et al. [11] use bouding boxes as weak cues for the domain adaptation of a segmentation model from the synthetic to the real domain.

The methods described above can interactively segment either a single object or depend on the user to provide the initial seeds as locations for segmentation. Some approaches require user-provided cues to point to objects similar in appearance. Due to these factors, the overall speed of annotation slows down. Additionally, the models are typically trained on images captured in good lighting conditions and contain very few objects. Our approach is robust while working with images captured in varying light and weather conditions. It can also handle images containing a large number of objects.

3. ScribbleNet - proposed approach

This section presents an overview of the proposed architecture followed by a theoretical foundation for the latent feature perturbation using the scribbles provided by the annotator at test time.

3.1. Motivation

ScribbleNet performs annotation by employing interactive segmentation. Our approach incorporates scribble cues at test time into an existing semantic segmentation architecture with pre-trained weights. Our method differs from the existing deep-learning-based interactive segmentation approaches which suffer from the following drawbacks: (a) propose their architecture specifically for interactive segmentation due to which they may not generalize to images with varying lighting and weather conditions, and unseen classes they were not trained for [15], and (b) append the user inputs to the image in both during training and inference that may not generalize to the differing annotator styles [12,13,21]. We now provide the details of our method.

3.2. Proposed architecture

ScribbleNet carries out efficient annotation using deep interactive segmentation by incorporating scribble cues to correct the segmentation. Scribbles encode two aspects in the proposed framework: (a) a subset of the pixels in the area we would like to correct, and (b) the class label of a segment which we would like to change. Note that in case (b), the segmentation area is correct, but the predicted label for the segment may be incorrect. Hence, scribbles cues enable us to correct both the segmentation and the class label while giving localized cues to refine the segmentation. We now describe the process by which the annotation of an image is generated by the iterative segmentation correction using user-proved scribbles.

3.3. Segmentation correction using scribble inputs

The user has an input image, X, for which an annotation A needs to be generated. The segmentation network $\mathcal{M}$ is parameterized by a set of weights W, such that $\mathcal{M}(X, W) = O$, where O is the segmentation output. A user examines the segmentation O, and provides scribbles $s_t(i, j, c_{ij})$ provided at iteration t to correct the segmentation where (i, j) are the pixel coordinates of the scribbles, c_{ij} denotes the desired class label.

To generate annotation for an image X, we perform multiple forward-backward iterations (denoted by t) over $\mathcal{M}$, which updates the weights W such that the segmentation O is of sufficient quality to be considered as annotation A. We refer to the process of updating the weights W using the scribbles as just-in-time training. We refer to the updated weights obtained at the end of iteration t as W_t. W_0 denote the weights of the pre-trained segmentation network. O_t is the segmentation output at iteration t.

Each iteration t, involves the following steps:

1. Obtain the segmentation output O_t:
$$O_t = \mathcal{M}(W_{t-1}, X) \tag{1}$$
2. Elicit additional user inputs in terms of a scribble $s_t(i, j, c_{ij})$ to correct the segmentation O_t.
3. Compute the just-in-time training loss $\mathcal{L}_{jit}$ as:
$$\mathcal{L}_{jit} = \sum_{(i,j,c_{ij}) \in s_t} \mathcal{L}(c_{ij}, p_{ij}) \tag{2}$$
where, p_{ij} denotes the predicted class label. Here, $\mathcal{L}$ denotes the loss computed for a scribble pixel coordinate (i, j) using the desired class label c_{ij} and the predicted class label p_{ij}. Note that any standard loss function similar to the one used for training semantic segmentation can be used here. However, we use the focal loss [35] for reasons described in the next section.
4. Back-propagate the loss computed above using stochastic gradient descent and update the model weights. We denote the updated model as W_t.
In principle, we could update the whole network. But as an efficiency/accuracy trade-off, we keep the lower layers fixed and only update the last few higher layers. The exact number of layers to update is determined by ablation studies.

Intuitively, just-in-time training loss helps to change the network weights to align a network's prediction with the user-provided scribbles. As shown in the next section, we look upon this as the perturbation that helps find the latent vector producing the labeling closest to the one provided by the user scribbles. Fig. 2 gives the graphical illustration of the steps. We perform multiple iterations of the steps described above until the output labeling O_t is deemed satisfactory by the annotator and can be considered as the annotation A.

3.3.1. Bayesian framework for latent feature perturbation

We provide a theoretical foundation for the proposed framework in terms of latent feature perturbation based on the scribble input at an iteration t. The theoretical justification is motivated by

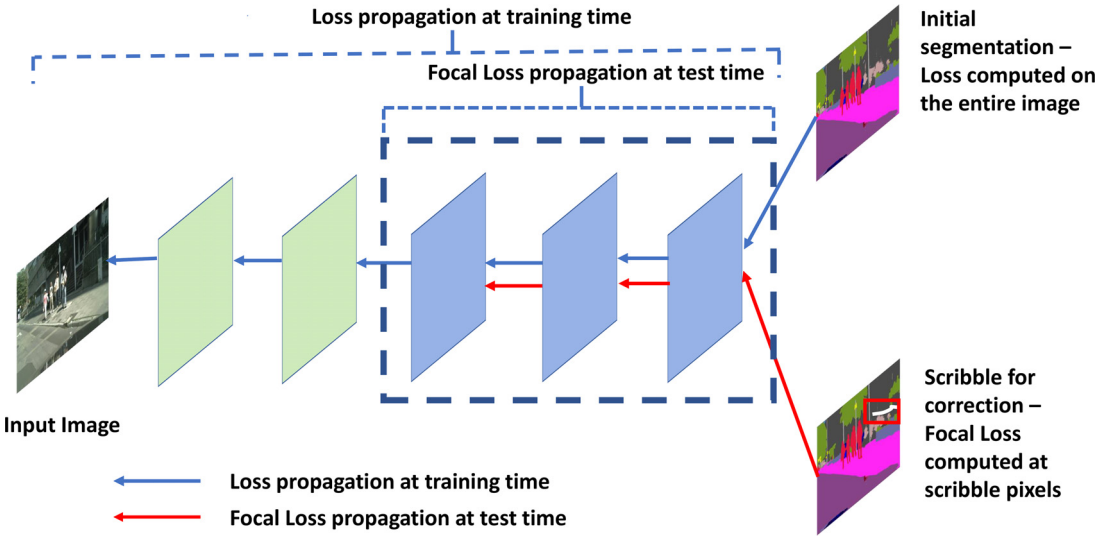

Fig. 2. Loss computation: Computation and propagation of loss at training and test time is shown here. At the training time, the loss is computed over the entire image. Focal loss is computed only from the scribble pixels during the test time. At test time, the back-propagation is performed only up to a few intermediate layers.

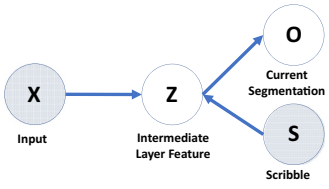

Fig. 3. Bayesian Network: The representation of input (X), intermediate layer feature (Z), Scribble (S), and current segmentation (O) given by the segmentation network is depicted here. **Inputs** - X and S (filled circles), **Outputs** - O and Z (unfilled circles). The back-propagation is performed using the loss computed at the scribble pixels alone (S). This is in turn employed to do latent feature perturbation only until a few layers of the network with the intermediate layer feature depicted by Z.

the arguments given in [36]. Consider the Bayesian network shown in Fig. 3, which models the dependencies between input (X), Current Segmentation (O), Intermediate Layer Feature (Z), and Scribble(s) provided by the user (S). For the sake of compactness, we have denoted O_t as O and s_t as s here. The observed variables are X and S. x and y are the input and the ground truth segmentation pixels at a pixel location (i, j) in the image. We denote the instance of the intermediate layer feature representation under consideration as z. In general for a segmentation task, we would like to find the best possible segmentation output o such that $\mathbb{P}(y|x)$ is maximized. This can be written as:

$$\arg\max_o \mathbb{P}(y|x) = \arg\max_o \sum_{z\in Z} \mathbb{P}(y|z)\mathbb{P}(z|x) \tag{3}$$

The latent feature representation z is perturbed to generate the segmentation output o that matches the ground truth segmentation y. We assume a fixed, but unknown probability over the latent variable z: $\mathbb{P}(z|x)$ (e.g., Gaussian), and assume that a DNN outputs the distribution mode. In our setting, we have additional information at the annotation time in the form of scribbles provided by the user to correct the segmentation containing the desired segmentation class label at a given location in an image. Hence, both the input pixel x as well as additional information encoded in the scribble s can be used to perturb z such that the segmentation is corrected. Thus, in our setting, the best possible segmentation output o is such that $\mathbb{P}(y|x, s)$ is maximized:

$$\arg\max_o \mathbb{P}(y|x,s) = \arg\max_o \sum_{z\in Z} \mathbb{P}(y|z)\mathbb{P}(z|x,s) \tag{4}$$

$$\approx \arg\max_o \mathbb{P}(y|z_{\max})\mathbb{P}(z_{\max}|x,s) \tag{5}$$

The value of $z_{\max}$ denotes the value of z maximizing $\mathbb{P}(y|x, s)$. Hence, we need to find the optimal value of z (feature representation) given the observed information about the scribbles and the input x to correct the segmentation. One possibility is to compute the $\partial\mathcal{L}_{jit}(o, s)/\partial z$ which can be used to perturb the value of z opposite to the computed gradient direction with the learning rate η. Post updation, the new value of z produces an output o which is closer to the scribble s. However, there is no valid mapping from x to z. We instead propose to perturb the values of W as per the gradient direction obtained from $\partial\mathcal{L}_{jit}(o, s)/\partial W$, since the weights of the network can be modified to obtain the desired segmentation output. The weights are updated in an iterative fashion using the scribble inputs till we arrive at the value of z which gives the output o matching the scribble input s. Since the weights are typically tied across space, the nearby pixels generally also get corrected along with the scribble pixels.

3.3.2. Optimization using back-propagation with focal loss

We now describe our method to back-propagate the loss using the scribble pixels to modify the weights during annotation time. After performing a forward pass through the network, the user provides additional information to the model in the form of scribbles. We divide the N layered network into two parts, i.e., m initial layers which are not updated to generate an annotation, and $N - m$ layers, which are updated, with the corresponding weights, denoted as W_{xm} and $W_{(N-m)o}$. The loss $\mathcal{L}_{jit}(o, s)$ is then back-propagated through the network. A regularization term is added to ensure that the new weights stay close to the original trained weights of the network. Since the weights W_{mx} in the first part of the network are fixed, the change in the proposed loss can only be effected through a change in the weights $W_{(N-m)o}$. The gradient of the total loss $\mathcal{L}_{\text{tot}}$ in an iteration t can be written as:

$$\nabla_{W_{(N-m)o}}\mathcal{L}_{\text{tot}}(.) = \nabla_z\mathcal{L}_{jit}\nabla_{W_{(N-m)o}}(z) + \beta\|W_{t(N-m)o} - W_{0(N-m)o}\| \tag{6}$$

Here, β is the regularization parameter, W_0 are the weights of the pre-trained network and W_t are the weights of the network at iteration t. We perform the segmentation correction using the scribble inputs provided by the user. Once the user is finally satisfied with the segmentation, the segmentation output is considered as the annotation A for a given image X.

We use Stochastic Gradient Descent (SGD) to minimize the proposed objective function (6) with respect to W, which is similar to the regular gradient descent employed for training deep neural networks. Hence, different loss functions used in the back-propagation ranging from cross-entropy to focal loss [35] can be used here as well. We use focal loss in our experiments. The choice is motivated by the observation that cross-entropy loss weighs all the scribble pixels equally, which may lead to difficulty in assigning a common learning rate that can switch the labels of all the pixels for which the scribble input is provided. Consider the case when a user would like to assign a pixel at a given location (i, j), a class label c. If the probability of label c outputted by the network originally is low, then it will require a larger number of iterations, and a higher learning rate, before the network can increase the probability of c significantly to make it the maximum among all the labels. On the other hand, if the probability of c were sufficiently high, a lower learning rate would have been sufficient for the switch. The focal loss allows us to down-weigh the easily switchable scribble pixels in the back-propagation, lowering their gradient values, and hence keep the same learning for all the scribbled pixels. The equation for focal loss is:

$$\text{focal-loss}(p_f) = -(1 - p_f)^{\gamma} \log(p_f) \tag{7}$$

where for a two-class case (foreground vs background),

$$p_f = \begin{cases} p_a & \text{if } y = 1 \\ 1 - p_a & \text{otherwise,} \end{cases} \tag{8}$$

where p_a is the probability of the class predicted by the labeling class 1. The use of focal loss allows to improves the chances that all the scribble pixels are corrected in the final annotation. The improvement in the output observed by using focal loss is inline with the other reported applications of focal loss. It has been utilized in object recognition to give more importance to difficult examples and down-weigh easily classified samples such that the difficult examples which are less in number also contribute to the loss value [35].

3.4. Weight persistence (WP) for video annotation

We update the weights W through back-propagation as a way to find the best labeling output consistent with user-provided scribbles. In images, this fixes errors in the spatial neighborhood with the scribble labels and the correlations learnt by the network. This idea can be easily extended to videos to fix errors in the temporal neighborhood. Consider an example of semantic segmentation of a frame b from a video. Suppose that our algorithm (incorrectly) labels a segment as a road in Step 1. In Step 2, the user provides scribbles for frame b and inputs the segment label as “sidewalk ”. In Step 3, we compute new weights W using our algorithm to label the segment as a sidewalk correctly. In the next frame $(b+1)$, retaining these weights will allow the network to correctly predict the segment as a “sidewalk” in Step 1 itself, thus eliminating the need for user-provided scribbles. We refer to the continuing use of weights updated while annotating frame b, in frame $(b+1)$ as weight persistence. We show that weight persistence helps further improvement in the user time for video annotation.

4. Dataset and evaluation

The main objective of our experiments is to demonstrate the reduction in annotation time for semantic segmentation. We apply the ScribbleNet approach to various urban city scene datasets. We compare ScribbleNet against full human annotation and other interactive segmentation methods. The experiments conducted display our approach’s domain-shift handling ability, i.e., to interactively segment images captured in conditions not present in the images in the training set. We demonstrate the capability of our method to interactively segment new classes not present in the training set.

For all the experiments described above, we have used the ERFNet model [2] trained on the CityScapes dataset [16] for 150 epochs, which generates the initial segmentation of the images. Though other architectures could have been used, we use ERFNet since it exhibits good performance for the semantic segmentation of urban city scenes [2]. But, for the sake of completion, we also test with a few other architectures such as HR-Net-OCR [37], HR-Net [38], Shelf-Net [3], LiteSeg [4] and CCNet [39] as the backbone in our ablation studies (see Section 6). The testing is done using a set of images selected from the CityScapes [16] belonging to the validation set, KITTI [19], Mapillary Vistas [17], BDD 100K [18], and IDD [20] datasets. We use qualitative metrics based on the similarity of images to ensure that the selected images are unique, as described in Section 5. We now provide the details of the datasets, evaluation metrics, and the baseline approaches.

4.1. Datasets

1. **CityScapes** [16]: CityScapes consists of a large and diverse set of stereo video sequences recorded from different cities. It consists of 30 classes which are most commonly found on the streets. It has 2975 training and 500 validation images with publicly available annotations, as well as 1525 test images.
2. **KITTI** [19]: KITTI consists of 200 annotated training images and 200 testing images. The KITTI dataset consists of images for semantic segmentation, 2D and 3D object detection, object tracking, road/lane detection, scene flow, depth evaluation, optical flow, and semantic instance-level segmentation.
3. **Mapillary Vistas (MV)** [17]: Mapillary Vistas consists of 25,000 high-resolution images annotated with 66 classes. It was captured in North America and South America, Europe, Africa, and Asia. It consists of images captured using different imaging devices like mobile phones, tablets, and action cameras in daylight, rain, snow, fog, haze, dusk, and night.
4. **BDD 100K** [18]: The dataset consists of images captured around the United States of America. It is the largest publicly available dataset with semantic segmentation, object detection, lane detection, driveable area, and semantic instance segmentation. It has been captured in daylight, rain, snow, fog, haze, dawn, dusk, and night.
5. **IDD** [20]: The IDD dataset was created for road scene understanding in unstructured environments. It consists of 10,000 images, finely annotated with 34 classes collected from 182 drive sequences on Indian roads.

4.2. Baseline techniques

We compare our approach with various well-known interactive segmentation methods such as GrabCut [5], Random Walk (RW) [8], Geodesic Star Convexity (GSC) [22], DEXTR [21], IntSeg, [12], f-BRS [13], Zhang et al. [23], Pan et al. [24] and Zheng et al. [25]. Since the deep-learning based approaches (IntSeg, DEXTR, f-BRS, Zhang et al. [23], Pan et al. [24], Zheng et al. [25]) were trained using object segmentation datasets, we retrained them using the training set of the CityScapes dataset. We used a pre-trained model of the ERFNet model trained on the CityScapes to maintain uniformity in comparison.

4.3. Evaluation metrics

To validate our approach, the annotation was done by employing two annotators who were well experienced in annotating urban city scene images. We evaluate our approach by employing

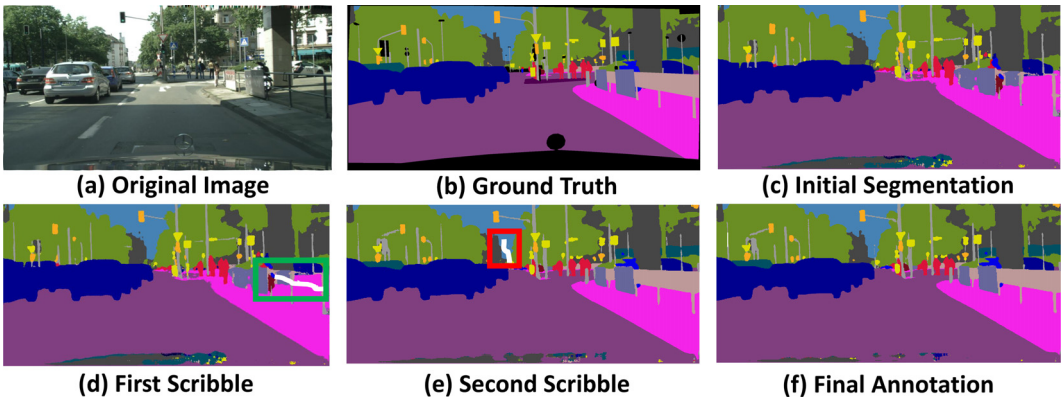

Fig. 4. Segmentation Correction with Scribbles: For an image (a) with the ground truth (b), the initial segmentation (c) is obtained using a pre-trained ERFNet model. The user provides scribbles as seen in images (d) and (e), and the final annotation (f) is obtained. The scribbles are marked in white and rectangular boxes have been drawn around the scribbles in red and green to highlight them.

two metrics to measure the quality and the total time taken by the annotator, similar to the evaluation done in [1]. For quality, we use the Intersection Over Union (IOU) metric, by computing the IOU over all the types of objects in the scene and averaging it. To capture the annotator's time, we asked the annotators to continuously provide scribbles till an IOU of 0.9 was obtained. Since a user provides scribble input in each iteration of our framework, the number of iterations used is reported as the number of user interactions for comparison.

5. Results and discussion

To show the benefits of ScribbleNet, we performed various experiments to highlight multiple settings that reduce annotator effort. The images selected were from several publicly available databases [16–20]. We also wanted to ensure that the subset of images picked from each dataset were different from each other using a quantitative metric. The metric used is based on the similarity of images in terms of the presence/absence of various classes. Both qualitative and quantitative results have been presented by comparing both popular classical methods [5,8,22] and recent deep-learning-based techniques [12,13,21,23–25]. All our experiments were performed using the NVIDIA 1080 Tx GPU. In all the figures, the scribbles are shown in white and highlighted by boxes for better visual clarity. However, the scribbles are provided with the color corresponding to the desired semantic class in the image location for our experiments.

Fig. 4 shows how the annotator provides scribbles to correct the segmentation and obtain the desired annotation iteratively. As the iterations progress, we observe that network output starts to match the scribble pixels' labels. Our method can exploit the correlations already learnt by the pre-trained network. For example, when a user provides a scribble to switch some pixels to say 'car', the back-propagation accordingly boosts the weights responsible for predicting 'car'. This key aspect of our technique improves the annotation time.

5.1. User time advantage

One of our approach's major strengths is that the annotator can save time compared to full human annotation and other interactive segmentation methods. To validate and highlight this capability, we selected five images each from the CityScapes [16], KITTI [19], Mapillary Vistas [17], BDD 100K [18], and the IDD [20] datasets. We also wanted to ensure that the images selected for annotation contain different objects which best reflect the performance of our approach on datasets with wide variability. Hence, we used the ground truth images provided with the datasets, and constructed a binary vector for each image with each bin representing the presence/absence of a semantic class such as a car, pedestrian and so on. We compute the percentage similarity between the binary vectors corresponding to the two images A and B with N classes as follows:

$$Similarity = 100 \times \frac{\sum_{i=0}^{N} A[i] \wedge B[i]}{N},$$

where $A[i]$ denotes ith element of vector A. Higher value of *similarity* indicates images with similar types of objects. We select images with lower values of *similarity* to ensure that they contain different objects, and ensure validation of our algorithm on widely varied images. Hence, we choose the image pairs with the five smallest values of similarity for each dataset. We then annotate each of the selected images in the following ways: (1) Full human annotation using the CityScapes. Here we used the annotation tool provided in the GitHub repository of the ERFNet algorithm [2]. (2) Interactive segmentation using ScribbleNet. (3) Interactive segmentation using baseline methods.

For each method, we recorded the user time to draw scribbles, the GPU time, and the network time. The user time is the time required for the user to provide the user inputs (scribbles, points, boxes). The GPU time is the time for the GPU to take the user inputs and correct the segmentation. The network time is the time to transfer the user inputs over the network to the GPU for correction. Since the user time and the GPU time are important factors that ensures faster annotation of images and reduce the annotator effort, we have presented these values in most experiments. These factors are compared against those of other interactive segmentation methods.

Table 1
User time saved (UTS) and total annotation time saved (TTS) saved over full human annotation.

Method	UTS	TTS
ScribbleNet	14.7x	15.4x
Zheng et al [25]	12.1x	11.3x
Pan et al [24]	11.6x	14.2x
Zhang et al [23]	10.5x	13.4x
f-BRS [13]	9.5x	10.4x
IntSeg [12]	8.5x	9.6x
DEXTR [21]	7.5x	9.4x
GrabCut [5]	5.0x	4.2x
GSC [22]	4.5x	5.6x
RW [8]	5.5x	6.7x

Table 2
Compares the ratio of user interactions for different datasets to the CityScapes dataset using the same method. BDD* represents the weather- degraded images from the BDD dataset. Grab Cut [5], Geodesic Star Convexity [22], Random Walks [8], DEXTR [21], IntSeg [12], f-BRS [13], Zhang et al. [23], Pan et al. [24], Zheng et al. [25].

Dataset	ScribbleNet	[5]	[22]	[8]	[12]	[21]	[13]	[23]	[24]	[25]
BDD	4.1	15.1	14.3	15.1	10.4	11.5	10.3	9.1	8.5	7.3
Mapillary Vistas	5.3	16.2	16.4	16.2	9.3	10.7	11.5	10.6	9.1	8.2
KITTI	6.2	15.3	15.3	14.4	10.2	9.9	10.2	10.1	9.8	8.5
IDD	4.5	14.4	14.3	13.5	8.1	10.8	8.7	8.5	7.2	7.1
BDD*	8.5	20.3	25.4	24.6	26.4	21.1	22.6	20.5	21.2	20.6

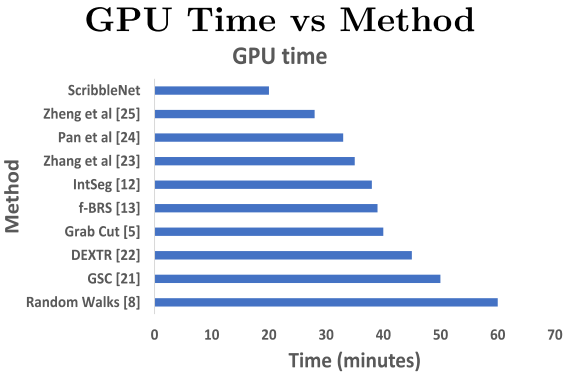

Fig. 5. Machine/GPU time for interactive segmentation by various techniques.

Table 1 shows the annotation time saved by each interactive segmentation approach for full human annotation. Fig. 5 shows a comparison of the average GPU time required by all the methods. We note that we save a significant amount of user time and GPU time compared to other approaches for a diverse set of images. The improvement in the user and GPU time achieved by our method can be attributed to the improved initial IOU of the segmentation by the pre-trained ERFNet model. Hence, we have studied the effect of the initial IOU of the segmentation on the annotation time in Section 6.

5.2. Model behavior in an unseen scenario: domain shift

We aim to evaluate our approach on images captured in conditions not present in the training set. Towards this, we test our interactive segmentation approach on images selected from the BDD 100k [18], Mapillary Vistas [17], KITTI [19], and the IDD [20] datasets. Since the effect of the unseen scenario is also seen as lower IOU of the initial segmentation, we select the images for these experiments on the following criterion: (1) Select the images with an approximate IOU of 50% when segmented with the pre-trained ERFNet model. (2) From the images with an IOU of approximately 50%, compute the percentage similarity among the binary vectors as described in the previous section, and select the image pairs with the lowest values from each dataset. Fig. 6 shows the number of user interactions and the GPU time required for each dataset by our approach. We see that our approach requires a far lesser number of user interactions and GPU time for the CityScapes dataset [16], which is expected since the model was pre-trained on the CityScapes dataset.

To understand the degradation due to unseen scenarios, we compute the annotation time using a particular approach on images captured in good weather conditions from the CityScapes dataset. We then compare it against images selected from BDD 100k [18], Mapillary Vistas [17], KITTI [19] and IDD [20] datasets. We do the same for our approach and other baseline approaches.

Table 2 shows the ratio of the average number of user interactions required for various datasets to the average number of interactions by the same method for the CityScapes dataset [16]. We note that other interactive segmentation methods require far more user interactions than our approach when tested on a new dataset on which it was not trained. This demonstrates that our method can better adapt to the domain shift.

5.3. Model performance on weather degraded images

Our next objective was to compare the performance of our approach on images captured in poor weather conditions. This may also be treated as a special case of domain shift. Like the previous experiments, we select images captured in poor weather conditions from the BDD 100k [18]. Fig. 7 shows the qualitative output. Even after many user interaction iterations, the techniques stopped improving beyond a point for most other methods. Hence, we had to stop the iterations when the output stopped changing. The last row in Table 2 shows the ratio of the number of average user interactions required for the weather degraded images in the BDD 100k dataset to the images in the CityScapes dataset for our approach and other interactive segmentation methods. Note that the increase in the number of interactions is far lesser for our method. Fig. 8 shows the user time and the GPU time required for the images captured in poor weather conditions. Again, we see that the time for our method is far lesser than the other methods.

5.4. Suitability of our approach for irregular contoured objects

The interactive segmentation approaches proposed so far have been primarily designed for the object segmentation tasks which are dominated by objects with box like shapes. On the contrary, urban city scenes comprise several entities such as vegetation, sky, and so on, which do not have box-like shapes and have more irregular contours. Our scribble-based approach is better suited for such scenes. The first step towards this is to define a quantitative metric that determines the *Boxiness* of an object as follows:

$$Boxiness = \frac{\text{Area of the bounding box around the object}}{\text{Number of pixels occupied by the object}} \quad (9)$$

We created the Boxiness metric to highlight the advantages of our approach for urban city scene images. The current interactive segmentation methods (DEXTR,f-BRS etc) were primarily created for natural images with a limited number of objects with a fairly well-defined structure [13,21]. However, a typical urban city scene image consists of objects with both a well-defined structure such as cars, bicycles as well as objects such as sky and greenery which do not have a well-defined structure. We wanted to show that our method is more effective than other approaches for city scene images. Hence, we defined the boxiness metric to differentiate images dominated by unstructured objects like sky and greenery from images with more structured objects such as cars, buildings and so on. We empirically determined that images with Boxiness greater than 0.5 were images with structured objects and the ones with Boxiness less than 0.2 were images with unstructured objects.

To validate our approach, we selected five images with an average boxiness greater than 0.5 and five images with an average box-

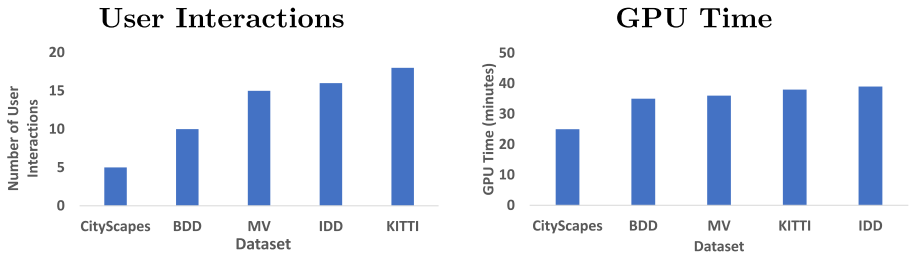

Fig. 6. Unseen Scenario by Training Set: Performance comparison of ScribbleNet applied to images from CityScapes dataset with its performance on images captured in scenarios not present in CityScapes dataset (MV - Mapillary Vistas).

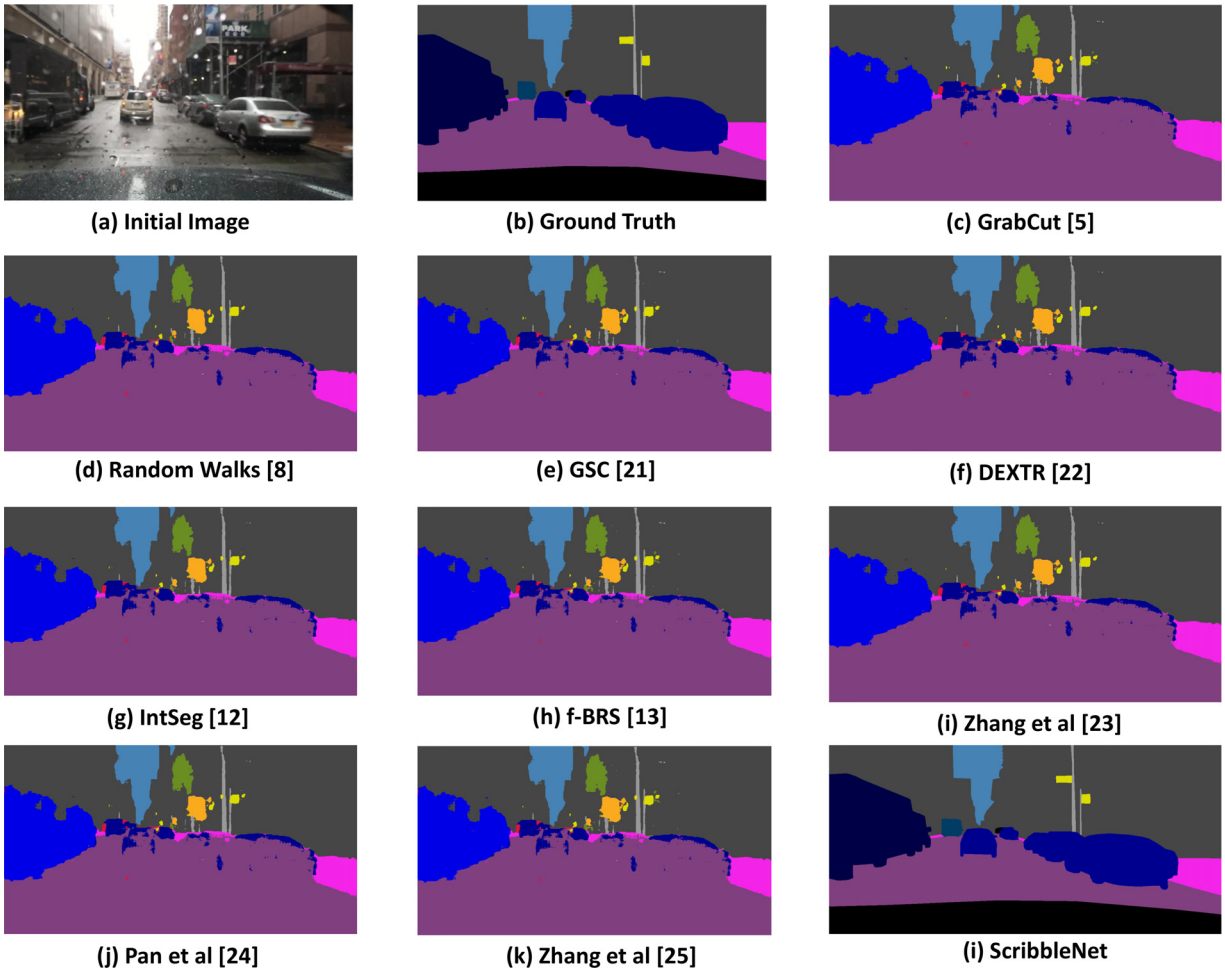

Fig. 7. Bad Weather: Results of annotated images captured in bad weather by various techniques.

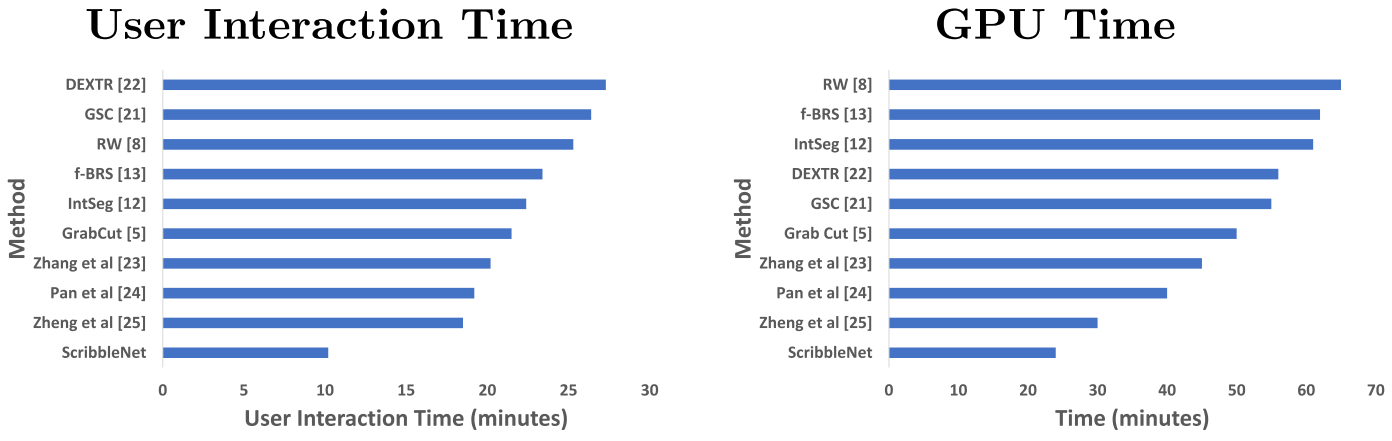

Fig. 8. Poor Weather: GPU and user time comparison for various approaches on weather degraded images from BDD (GSC - Geodesic Star Convexity, RW - Random Walks).

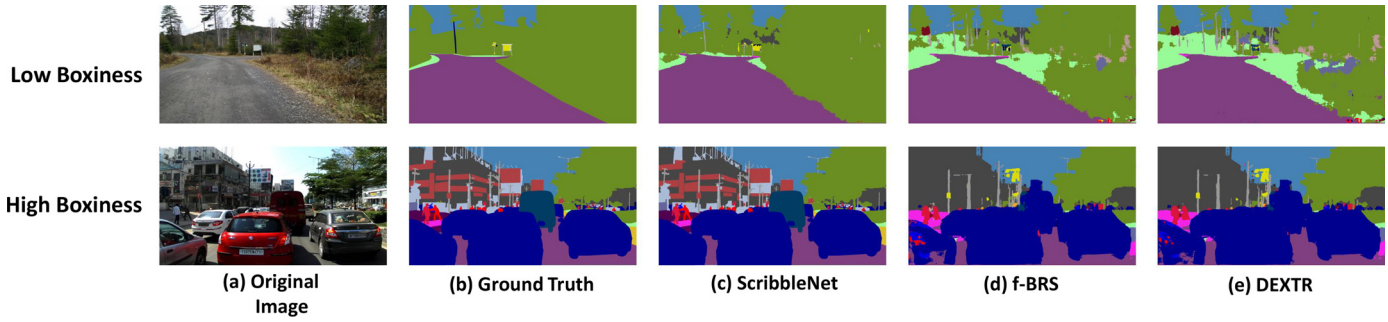

Fig. 9. Suitability of ScribbleNet for irregular contoured objects: The annotation results on images with a high boxiness (bottom row) and a low boxiness (top row) are shown.

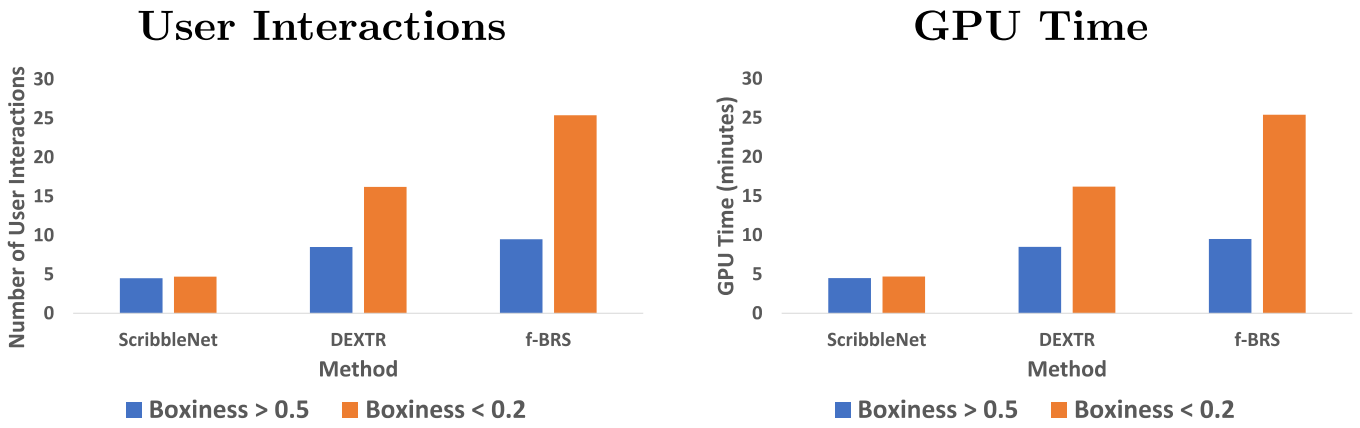

Fig. 10. Boxiness: Comparison of user and GPU time on images with different levels of irregular shaped objects.

iness less than 0.2. We then compared the performance of ScribbleNet on these images, against DEXTR [21] and f-BRS [13], which work with user inputs given in the form of boxes and points, respectively. In Fig. 9, our method can handle images better irrespective of the contour types, whereas other methods are more suited for images with more regular contours. Fig. 10 shows the number of user interactions and the GPU time required for each method. We see that ScribbleNet requires both a lesser number of user interactions and GPU time for both types of images. However, we see that for images with an average boxiness of less than 0.2, i.e. images that have irregular contours, the other methods require a much larger amount of GPU and user time. The difference in GPU time and user time for images with a boxiness greater than 0.5, i.e., images with regular contours are less significant when comparing our method with other approaches. This clearly demonstrates the suitability of our approach for images with both regular and irregular contoured objects, which is the case with urban city scenes.

5.5. Ability to learn unseen classes

Our technique's significant strength is its ability to learn new classes unseen while training the segmentation network. We select the IDD dataset [20] for this experiment since it has 13 new classes not present in the CityScapes dataset [16]. To segment the unseen classes, we replaced the last layer of the pre-trained EFRNet network with a layer of size having the older classes plus the newer classes. For the older classes, we retained the older weights. For the newer classes, we initialized with random values. Similar to the earlier experiments, we ensure that the selected images are different from each other in the following manner: (1) Rank the images in terms of the number of unknown classes present and select the top 25 images containing the most number of unknown classes. (2) Use the similarity metric described earlier to select the 5 most dissimilar images for experiments. We compute the Intersection Over Union (IOU) by interactively segmenting the images using our technique and other methods. For each method, we interactively segment the image until it could not be improved any further. Fig. 11 shows the qualitative and quantitative comparison. Our method shows a superior performance in terms of segmentation accuracy.

5.6. Video annotation

The proposed approach, ScribbleNet can be used to segment images captured in a sequential manner i.e., frames of a video. The consecutive frames in a video share information with each other. Hence an important question that arises in this context is "Can the learned weights from the first frame in the sequence be used to effectively segment the subsequent frames with far lesser annotation time than for images captured independent of each other?" Specifically, given two images A and B, which are two consecutive frames, is it possible to use the final set of weights after the interactive segmentation of frame A to segment B with fewer scribbles than if frame B was segmented independently.

We employ our framework seamlessly to annotate subsequent frames with either the pre-trained weights from the ERFNet model or re-trained weights from the previous frames. Hence, we aimed to verify which of the two approaches was more efficient to annotate video frames. We took four consecutive frames from the CityScapes [16], KITTI [19], and the BDD 100k [18] datasets. We then proceed to interactively segment the first frame from each dataset by providing scribbles. For the next frames, we perform two types of experiments (a) Use the re-trained weights from the previous frame as the starting set of weights for the current frame (b) Use the original set of weights from the pre-trained ERFNet model for all the frames. Fig. 12 shows the annotation of sequen-

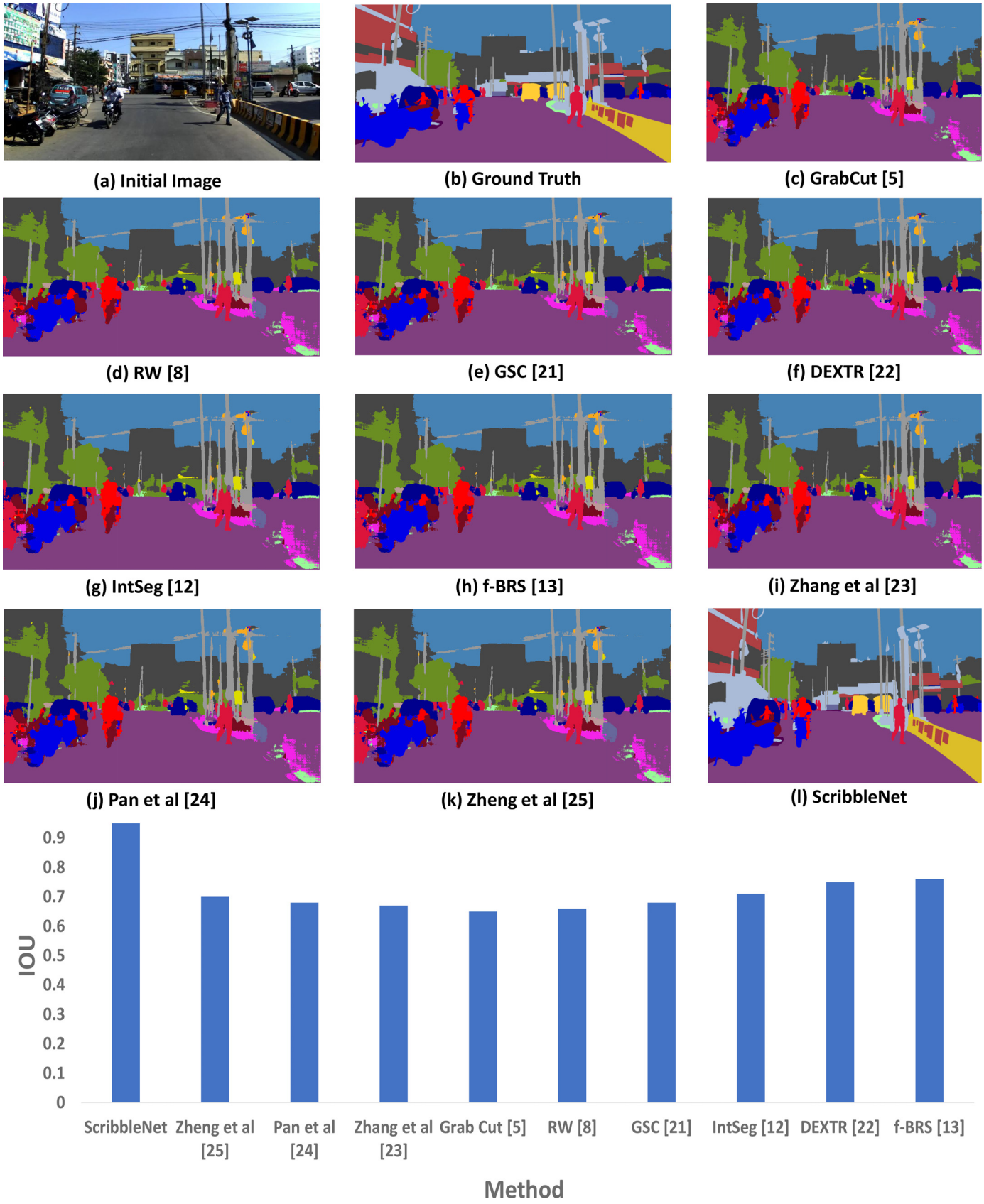

Fig. 11. Unseen Classes: Comparative results when the image to be annotated contains unseen classes. The image from the IDD dataset contains objects such as autorickshaws and billboards not present in Cityscapes dataset. Note that each method is pre-trained on the images from the training set of the Cityscapes dataset. The histogram contains the IoU obtained with each method (GSC - Geodesic Star Convexity, RW - Random Walks).

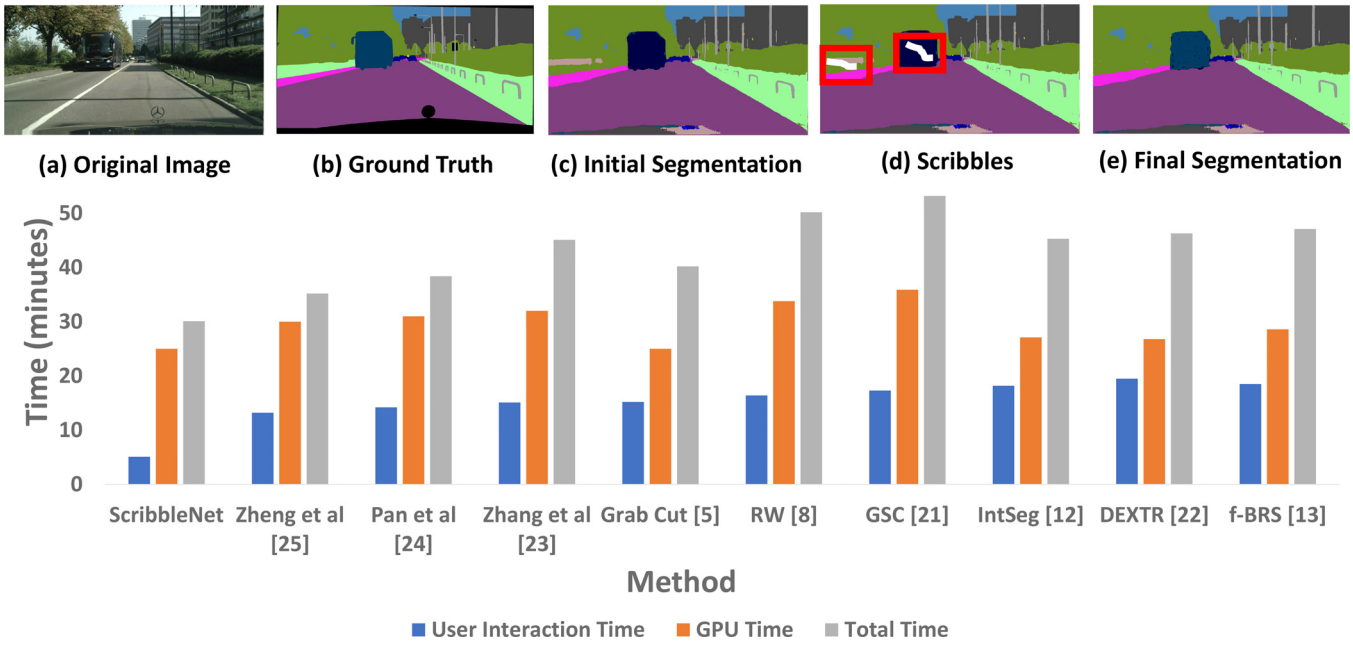

Fig. 12. Video annotation: The annotation of four consecutive video frames with and without weight persistency is shown here. It may be seen that the user needs to provide more scribbles when weight persistency is not used. (**Please zoom to see the scribbles clearly**).

tial data with our approach. Many of the errors that we need to fix through annotations for the first frame are auto-corrected in the subsequent frames, saving time in re-correcting them. Quantitatively also, the user time, and GPU time (which depends on the number of user iteration) reduces significantly when weight persistence is used for sequential data (c.f. Fig. 13). This further consolidates the utility of our method.

5.7. Multiple label correction

The next experiment's objective is to show the correction of multiple areas simultaneously. For this, we first manually selected images from the validation set of the CityScapes dataset where it is possible to correct multiple labels. Then we selected five images using the procedure described in Section 5.1 to ensure that the images were different and then correct the segmentation to obtain an IOU of 0.9. As seen in Fig. 14, ScribbleNet requires the least amount of time compared to the other methods.

6. Ablation study

We have conducted various ablation studies to understand the effect of various hyper-parameters on the annotation efficiency obtained by our method:

1. The ScribbleNet framework can be built around any semantic segmentation network. Hence, our first ablation study was to select the appropriate semantic segmentation network developed for the segmentation of urban city scenes. Thus, we selected six semantic segmentation networks from the CityScapes dataset leaderboard i.e., HR-Net OCR [37], HR-Net [38], ERFNet [2], ShelfNet [3], LiteSeg [4], and CCNet [39] for which the source code and the pre-trained segmentation models for the CityScapes dataset were available. The interactive segmentation built using the semantic segmentation network and scribbles provides segmentation correction to obtain an IOU of 0.9. We compared the user time, GPU time, and the total time for the interactive segmentation. We found that ERFNet took the least time among all the segmentation networks as seen in Fig. 15. Hence, we selected ERFNet as our experiments' base semantic segmentation network.
2. Our second ablation study is the effect of the number of back-propagation iterations performed after each user interaction. We selected 500 images from the CityScapes dataset [16] for this study. Fig. 16 shows the number of back-propagation iterations versus the IOU obtained, from which we see that we obtain the optimum IOU with 50 back-propagation iterations.
3. Once we determine the right number of iterations, the next step is to determine the layer upto which we need to back-propagate for optimum performance (number of layers to freeze weights). We observe optimum performance by back-propagating until layer seven, as seen in Fig. 16.
4. Our method relies on pre-learned correlations using a DNN to improve the segmentation output given by a user or annotator. Hence, if there are no or very few such learnt correlations, fully manual annotation is likely to be faster. We have tried to relate it with the IOU achieved by a DNN in the first forward propagation when no user input has been received yet. We selected images from various datasets with the initial IOU ranging from 0.2 to 0.85. We compared the user time for interactive segmentation with our approach against complete human annotation. We see from Fig. 16 that the user time increases as the initial IOU decreases. For an image with an IOU lesser than 0.3, the interaction time overshoots the full human annotation time. This is one of the limitations of our approach, which can be overcome by fine-tuning a pre-trained DNN on a small subset of images from the target domain annotated in conventional fully manual style.

We also conducted the following experiments: (a) Segmentation Deterioration for area out of scribbles when the segments were similar in appearance but belonged to different classes (b) Convergence of ScribbleNet for images containing only seen classes vs images containing both seen and unseen classes, and (c) Impact of Order of Interactive Segmentation Methods on the segmentation correction. Owing to space constraints, the details of these experiments have been provided in the supplementary material after the references.

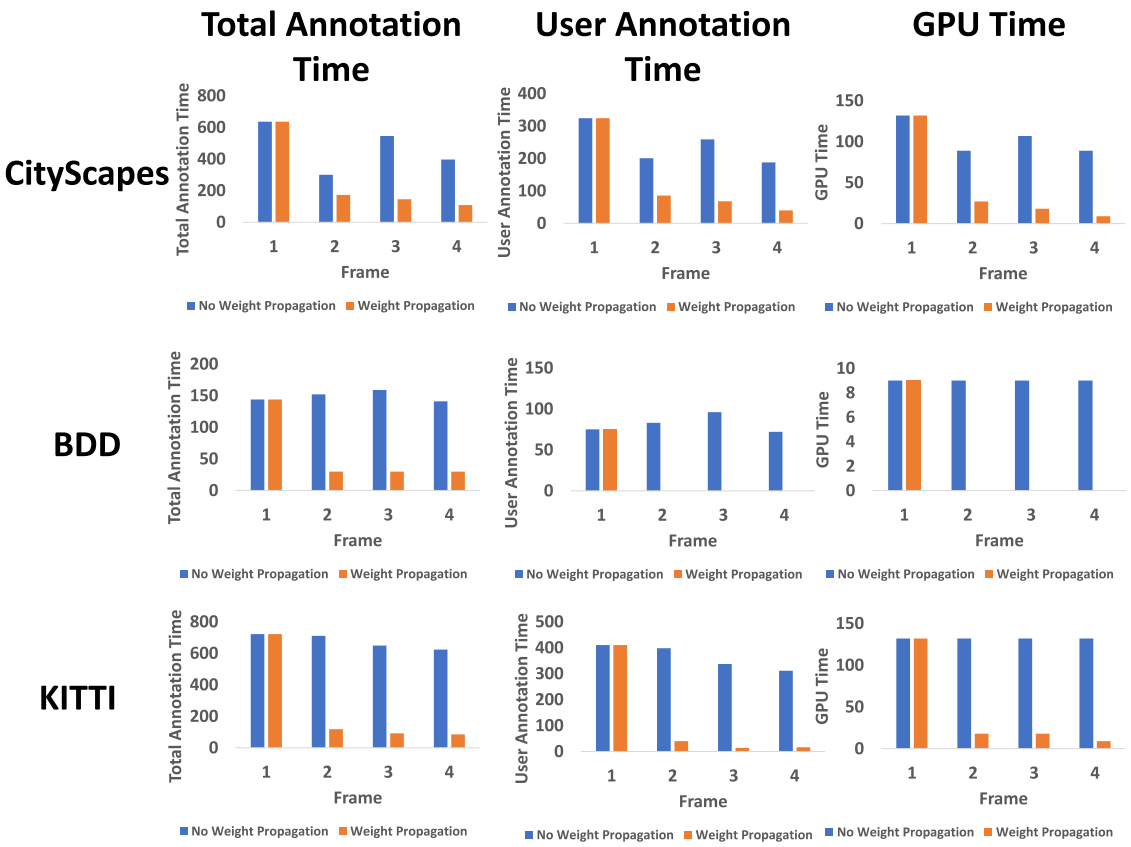

Fig. 13. Video Time Comparison: Annotation time (minutes) comparison for streaming video data from all datasets with and without weight persistency. Note that the time with persistent weights for BDD was almost zero, and so the bars are not visible. **Column 1:** Total Annotation Time. **Column 2:** User Annotation Time. **Column 3:** GPU Time.

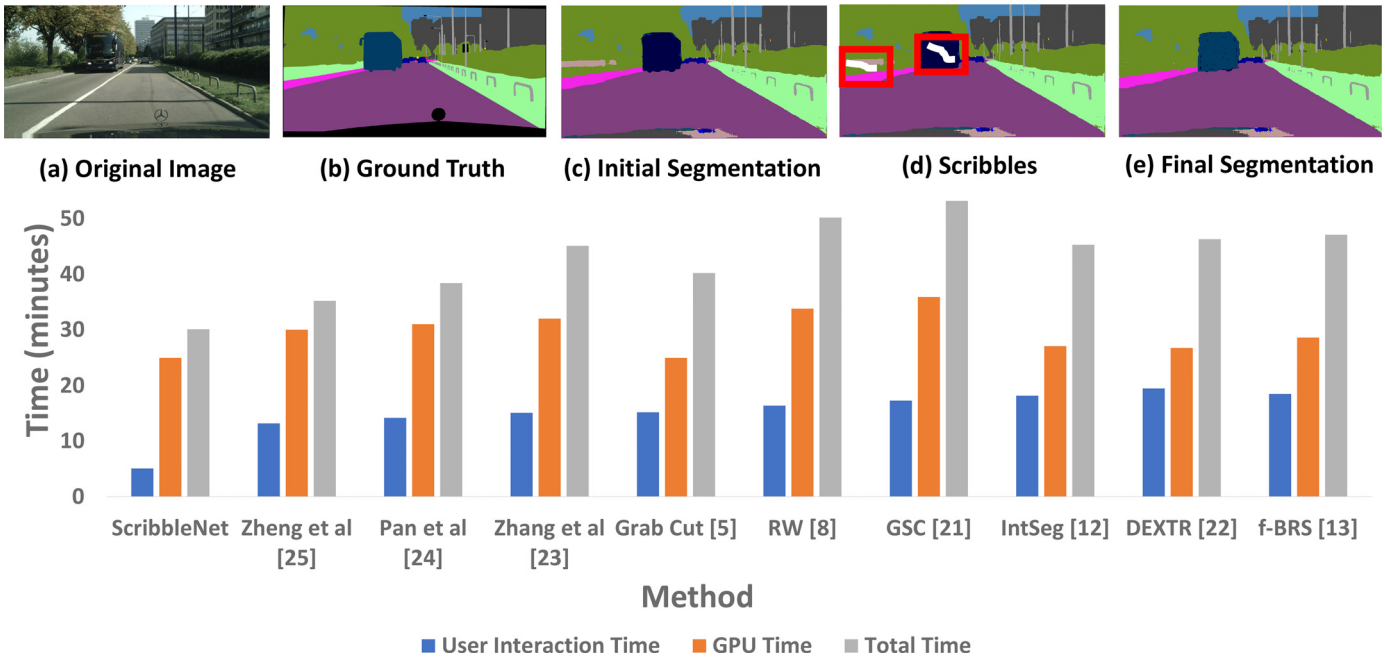

Fig. 14. Multiple Label Correction: Comparative results when multiple labels are corrected at the same time. (Please zoom for better clarity). The top figure shows the qualitative results when multiple scribbles are provided for correcting multiple classes simultaneously. The histogram at the bottom shows the time comparison (user time, GPU time and total time) by different approaches. It is apparent that since ScribbleNet has the ability to correct multiple labels at the same time, the overall time required for interactive segmentation is lesser than that of other methods.

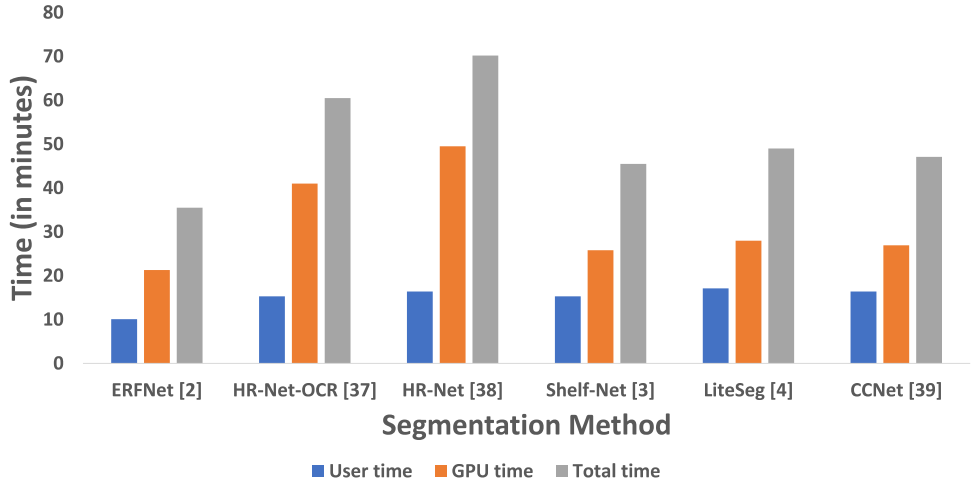

Fig. 15. Segmentation Method Comparison: Timing Comparison for Semantic Segmentation Methods.

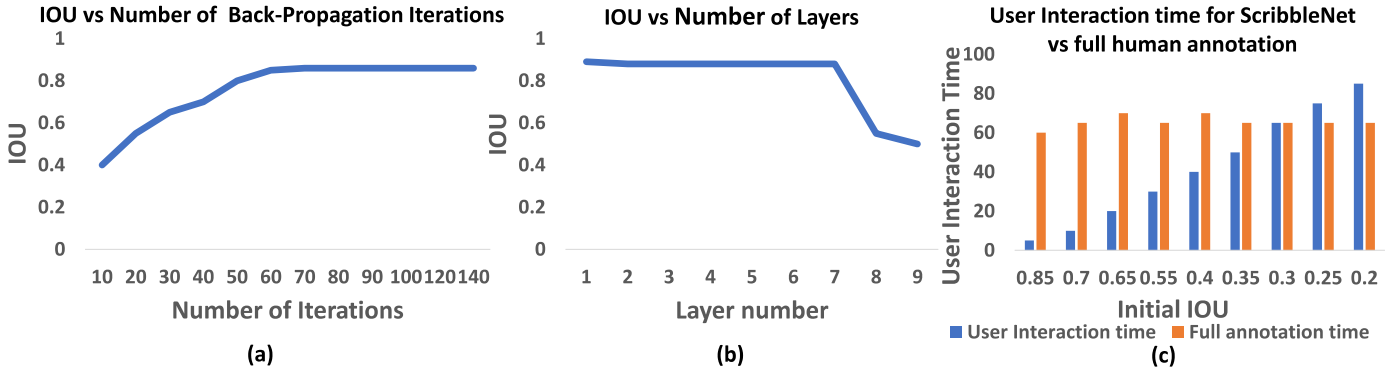

Fig. 16. Ablation Study: (a) IOU vs number of iterations up to which back-propagation should be done. We see that 50 iterations are sufficient to obtain the optimum IOU. (b) IOU vs the layer up to which back-propagation should be done. It is seen that we need to train up to layer 7 for an optimum IOU. (c) User interaction time for our approach and full human annotation time versus the initial IOU of the segmentation by the pre-trained segmentation network. As the IOU of the initial segmentation is reduced, a greater amount of user interaction time is required to reach an optimum IOU.

Fig. 17. Annotation Tool: Screenshot of the ScribbleNet Annotation Tool.

7. Annotation tool

We have created an annotation tool to facilitate the annotation of images built on top of the LabelMe image annotation tool [26], as shown in Fig. 17. When the annotator invokes the tool, he loads an image to be annotated. The image is first segmented using the ERFNet segmentation method. The annotator then examines the segmentation result to see if it looks satisfactory. Scribbles are then provided in areas where the annotator desires an improvement in annotation. The segmentation is then corrected based on the user-provided inputs. This is continued till the annotator is completely satisfied with the annotation.

8. Conclusions

In this paper, we have proposed a method to facilitate rapid annotation for semantic segmentation of urban city scenes. Our system utilises the scribble cues provided by the annotator to obtain the annotation of the image. We achieve a speed improvement of 14.7× compared to full human annotation and also perform considerably better than the other interactive segmentation methods. We have demonstrated the capability to annotate unseen classes not seen in the training set and annotating several classes at once which is crucial for any semi-automated annotation algorithm. Our method's success bodes well for its application in other related fields such as medical and industrial imaging. We observed that for images containing similar looking regions, but belonging to different classes, ScribbleNet sometimes got confused between the segmentation classes and corrected the segment of one class as belonging to the other class. The very same limitation is present in other interactive segmentation methods and is not exclusive to ScribbleNet. In our experiment with video annotation, since the weights of the network are updated after every frame, this can potentially lead to catastrophic forgetting if a frame similar to the one seen in the distant past reappears in the video. In the future, we would like to improve our method to overcome both these limitations. We will release the complete source code of our framework upon acceptance.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2022.109011

References

[1] D. Acuna, H. Ling, A. Kar, S. Fidler, Efficient interactive annotation of segmentation datasets with polygon-RNN++, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 859–868.

[2] E. Romera, J.M. Alvarez, L.M. Bergasa, R. Arroyo, ERFNet: efficient residual factorized convnet for real-time semantic segmentation, IEEE trans. Intell. Transp. Syst. 19 (1) (2017) 263–272.

[3] J. Zhuang, J. Yang, L. Gu, N. Dvornek, Shelfnet for fast semantic segmentation, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2019. 0–0

[4] T. Emara, H.E. Abd El Munim, H.M. Abbas, LiteSeg: a novel lightweight convnet for semantic segmentation, in: Proceedings of the Digital Image Computing: Techniques and Applications, IEEE, 2019, pp. 1–7.

[5] C. Rother, V. Kolmogorov, A. Blake, "GrabCut" - Interactive foreground extraction using iterated graph cuts, TOG 23 (3) (2004) 309–314.

[6] Y. Li, J. Sun, C. Tang, H.-Y. Shum, Lazy snapping, ToG 23 (3) (2004) 303–308.

[7] V. Vezhnevets, V. Konouchine, Growcut: interactive multi-label ND image segmentation by cellular automata, in: Proceedings of the International Conference on Computer Graphics and Vision, volume 1, 2005, pp. 150–156.

[8] L. Grady, Random walks for image segmentation, IEEE PAMI 28 (11) (2006) 1768–1783.

[9] M. Kass, A. Witkin, D. Terzopoulos, Snakes: active contour models, Int. J. Comput. Vis. 1 (4) (1988) 321–331.

[10] E.N. Mortensen, W.A. Barrett, Intelligent scissors for image composition, in: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, 1995, pp. 191–198.

[11] Q. Wang, J. Gao, X. Li, Weakly supervised adversarial domain adaptation for semantic segmentation in urban scenes, IEEE Trans. Image Process. 28 (9) (2019) 4376–4386.

[12] Z. Li, Q. Chen, V. Koltun, Interactive image segmentation with latent diversity, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 577–585.

[13] K. Sofiiuk, I. Petrov, O. Barinova, A. Konushin, f-BRS: rethinking backpropagating refinement for interactive segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 8623–8632.

[14] H.B. Lee, T. Nam, E. Yang, S.J. Hwang, Meta dropout: learning to perturb latent features for generalization, in: Proceedings of the International Conference on Learning Representations, 2019.

[15] J. Liew, Y. Wei, W. Xiong, S. Ong, J. Feng, Regional interactive image segmentation networks, in: Proceedings of the IEEE International Conference on Computer ision, 2017, pp. 2746–2754.

[16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset for semantic urban scene understanding, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3213–3223.

[17] G. Neuhold, T. Ollmann, S. Rota Bulo, P. Kontschieder, The mapillary vistas dataset for semantic understanding of street scenes, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 4990–4999.

[18] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, T. Darrell, BDD100K: a diverse driving dataset for heterogeneous multitask learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 2636–2645.

[19] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, Vision meets robotics: the KITTI dataset, Int. J. Robot. Res. 32 (11) (2013) 1231–1237.

[20] G. Varma, A. Subramanian, A. Namboodiri, M. Chandraker, C.V. Jawahar, IDD: a dataset for exploring problems of autonomous navigation in unconstrained environments, in: Proceedings of the IEEE Winter Conference on Applications of Computer Vision, IEEE, 2019, pp. 1743–1751.

[21] K. Maninis, S. Caelles, J. Pont-Tuset, L. Van Gool, Deep extreme cut: from extreme points to object segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 616–625.

[22] V. Gulshan, C. Rother, A. Criminisi, A. Blake, A. Zisserman, Geodesic star convexity for interactive image segmentation, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 3129–3136.

[23] S. Zhang, J.H. Liew, Y. Wei, S. Wei, Y. Zhao, Interactive object segmentation with inside-outside guidance, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 12234–12244.

[24] Z. Pan, P. Jiang, C. Tu, Scribble-supervised semantic segmentation by uncertainty reduction on neural representation and self-supervision on neural eigenspace, in: Proceedings of the IEEE International Conference on Computer Vision, volume 35, 2021, pp. 7416–7425.

[25] E. Zheng, Q. Yu, R. Li, P. Shi, A. Haake, A continual learning framework for uncertainty-aware interactive image segmentation, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, 2021, pp. 6030–6038.

[26] B.C. Russell, A. Torralba, K.P. Murphy, W.T. Freeman, LabelMe: a database and web-based tool for image annotation, Int. J. Comput. Vis. 77 (1–3) (2008) 157–173.

[27] A. Criminisi, T. Sharp, A. Blake, Geos: geodesic image segmentation, in: Proceedings of the European Conference on Computer Vision, 2008, pp. 99–112.

[28] B.L. Price, B. Morse, S. Cohen, Geodesic graph cut for interactive image segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 3161–3168.

[29] W. Jang, C. Kim, Interactive image segmentation via backpropagating refinement scheme, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 5297–5306.

[30] C. Panagiotakis, H. Papadakis, E. Grinias, N. Komodakis, P. Fragopoulou, G. Tziritas, Interactive image segmentation based on synthetic graph coordinates, Pattern Recognit. 46 (11) (2013) 2940–2952.

[31] T. Wang, Z. Ji, Q. Sun, Q. Chen, Q. Ge, J. Yang, Diffusive likelihood for interactive image segmentation, Pattern Recognit. 79 (2018) 440–451.

[32] B.C. Benato, J.F. Gomes, A.C. Telea, A.X. Falcão, Semi-automatic data annotation guided by feature space projection, Pattern Recognit. 109 (2021) 107612.

[33] N. Xu, B. Price, S. Cohen, J. Yang, T.S. Huang, Deep interactive object selection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 373–381.

[34] Z. Lin, Z. Zhang, L.-Z. Chen, M.-M. Cheng, S.-P. Lu, Interactive image segmentation with first click attention, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 13339–13348.

[35] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.

[36] D. Khandelwal, S. Agarwal, P. Singla, C. Arora, A novel technique for evidence based conditional inference in deep networks via latent feature perturbation, arXiv preprint arXiv:1811.09796 (2018).

[37] A. Tao, K. Sapra, B. Catanzaro, Hierarchical multi-scale attention for semantic segmentation, arXiv preprint arXiv:2005.10821 (2020).

[38] K. Sun, Z. Yang, J. Borui, C. Tianheng, X. Bin, L. Dong, M. Yadong, W. Xinggang, L. Wenyu, W. Jingdong, High-resolution representations for labeling pixels and regions, arXiv preprint arXiv:1904.04514 (2019).

[39] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, W. Liu, Ccnet: Criss-Cross attention for semantic segmentation, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 603–612.

Bhavani Sambaturu received her MSc from École Polytechnique Fédérale de Lausanne. She is currently pursuing her PhD at International Institute of Information Technology, Hyderabad. Her research interests include computer vision and machine learning.

Ashutosh Gupta received his B.E. from Indraprashtha Institute of Information Technology Delhi. He is currently pursuing his MSc at Indian Institute of Technology, Delhi. His research interests include computer vision and machine learning.

C.V. Jawahar is currently a Professor at International Institute of Information Technology, Hyderabad. His research interests lie in the areas of computer vision, machine learning and multimedia systems.

Chetan Arora is currently an Associate Professor at Indian Institute of Technology, Delhi. His research interests include trustworthy AI, cancer detection, mobility, egocentric vision and social impact.