Rethinking Interactive Image Segmentation: Feature Space Annotation

Jordão Bragantini, Alexandre X Falcão, Laurent Najman

Rethinking Interactive Image Segmentation: Feature Space Annotation

Jordão Bragantini[a,b,*], Alexandre X. Falcão[a], Laurent Najman[b]

[a] *University of Campinas, Laboratory of Image Data Science, Brazil*
[b] *Université Gustave Eiffel, LIGM, Equipe A3SI, ESIEE, France*

Abstract

Despite the progress of interactive image segmentation methods, high-quality pixel-level annotation is still time-consuming and laborious — a bottleneck for several deep learning applications. We take a step back to propose interactive and simultaneous segment annotation from multiple images guided by feature space projection. This strategy is in stark contrast to existing interactive segmentation methodologies, which perform annotation in the image domain. We show that feature space annotation achieves competitive results with state-of-the-art methods in foreground segmentation datasets: iCoSeg, DAVIS, and Rooftop. Moreover, in the semantic segmentation context, it achieves 91.5% accuracy in the Cityscapes dataset, being 74.75 times faster than the original annotation procedure. Further, our contribution sheds light on a novel direction for interactive image annotation that can be integrated with existing methodologies. The supplementary material presents video demonstrations. Code available at `https://github.com/LIDS-UNICAMP/rethinking-interactive-image-segmentation`.

Keywords: interactive image segmentation, data annotation, interactive machine learning, feature space annotation

1. Introduction

Convolutional Neural Networks (CNNs) can achieve excellent results on image classification, image segmentation, pose detection, and other images/video-related tasks [1, 2, 3, 4, 5], at the cost of an enormous amount of high-quality annotated data and processing power. Thus, interactive image segmentation with reduced user effort is of primary interest to create such datasets for the training of CNNs. Concerning image segmentation tasks, the annotations are pixel-wise labels, usually defined by interactive image segmentation methods [6, 7, 8, 9, 10, 11] or by specifying polygons in the object boundaries [12].

*Corresponding author
Email addresses: `jordao.bragantini@gmail.com` (Jordão Bragantini), `afalcao@ic.unicamp.br` (Alexandre X. Falcão), `laurent.najman@esiee.fr` (Laurent Najman)

Work done during an internship at LIGM.

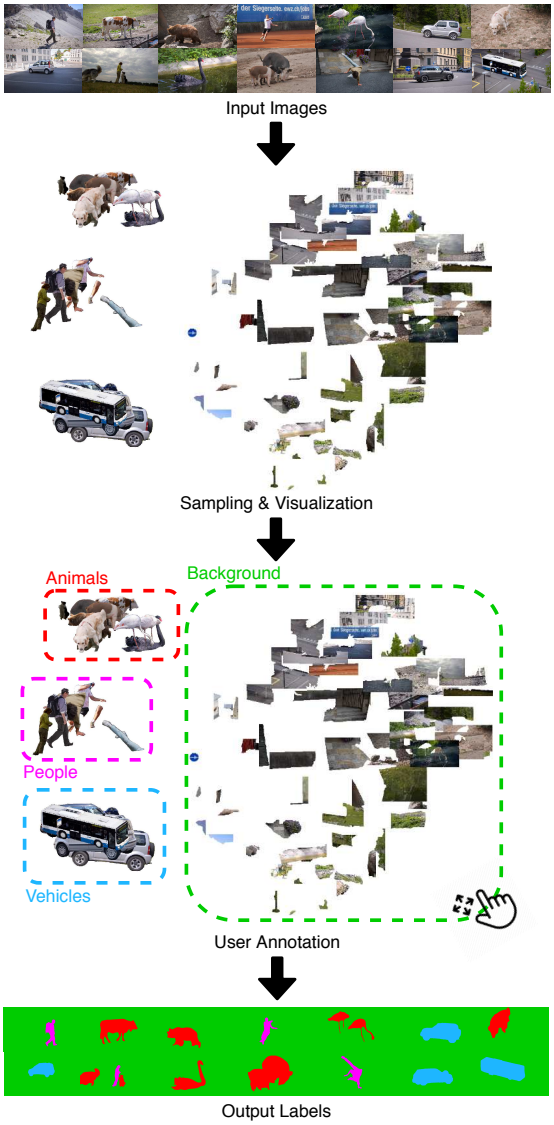


Figure 1: Our approach to interactive image segmentation: candidate segments are sampled from the dataset and presented in groups of similar examples to the user, who annotates multiple segments in a single interaction.

Recent interactive image segmentation methods based on deep learning can significantly reduce user effort by performing object delineation from a few clicks, sometimes in a single user interaction [13, 14, 15]. However, such deep neural networks do not take user input as hard constraints and so cannot provide enough user control. Novel methods can circumvent this issue by refining the neural network's weights

while enforcing the correct results on the annotated pixels [16, 17, 18]. Their results are remarkable for foreground segmentation. Still, in complex cases or objects unseen during training, the segmentation may be unsatisfactory even by extensive user effort.

The big picture in today's image annotation tasks is that thousands of images with multiple objects require user interaction. While they might not share the same visual appearance, their semantics are most likely related. Hence, thousands of clicks to obtain thousands of segments with similar contexts do not sound as appealing as before.

This work presents a scheme for interactive large-scale image annotation that allows user labeling of many similar segments at once. It starts by defining segments from multiple images and computing their features with a neural network pre-trained from another domain. User annotation is based on feature space projection, Figure 1. As it progresses, the similarities between segments are updated with metric learning, increasing the discrimination among classes, and further reducing the labeling burden.

Contribution: To our knowledge, this is the first interactive image segmentation methodology that does not receive user input on the image domain. Hence, our goal is not to beat the state-of-art of interactive image segmentation but to demonstrate that other forms of human-machine interaction, notably feature space interaction, can benefit the interactive image segmentation paradigm and can be combined with existing methods to perform more efficient annotation.

2. Related Works

In this work, we address the problem of assigning a label (*i.e*. class) to every pixel in a collection of images, denoted as image annotation in the remaining of the paper. This is related to the foreground (*i.e*. region) segmentation microtasks, where the region of interest has no specific class assigned to it, and the delineation of the object is of primary interest — in standard image annotation procedures, this is the step preceding label assignment [12].

Current deep interactive segmentation methods, from click [19, 13, 20, 18, 16, 17, 14], bounding-box [20, 21, 15] to polygon-based approaches [22, 23, 24] address this microtask of segmenting and then labeling each region individually to generate annotated data.

A minority of methods segment multiple objects jointly; to our knowledge, in deep learning, this has been employed only once [25]; in classical methods, a hand-full could do this efficiently [26, 27, 10].

Fluid annotation [28] proposes a unified human-machine interface to perform the complete image annotation; the user annotation process starts from the predictions of an existing model, requiring user interaction only where the model lacks accuracy, further reducing the annotation effort. The user decides which action it will perform at any moment without employing active learning (AL). Hence, the assumption is that the user will take actions that will decrease the annotation budget the most.

This approach falls in the Visual Interactive Labeling (VIAL) [29] framework, where the user interface should empower the users, allowing them to decide the optimal move to perform the task efficiently. Extensive experiments [30] have shown that this paradigm is as competitive as AL and obtains superior performance when starting with a small amount of annotated data.

Inside the VIAL paradigm, feature space projection has been employed for user guidance in semi-supervised label propagation [31, 32] and for object detection in remote-sensing [33]. However, its use for image segmentation has not been explored yet.

Other relevant works and their relationship with our methodology are further discussed in the next section.

3. Proposed Method

Our methodology allows the user to annotate multiple classes jointly and requires lower effort when the data are redundant by allowing multiple images to be annotated at once. Multiple image annotation is done by extracting candidate regions (*i.e*. segments) and presenting them simultaneously to the user. The candidate segments are displayed closer together according to their visual similarity [34] for the label assignment of multiple segments at once. Hence, user interaction in the image domain is only employed when necessary — not to assign labels but to fix incorrect segments. Since, this action is the one that requires the most user effort and has been the target of several weakly-supervised methodologies [35, 36, 37, 38] that try to avoid it altogether. Therefore, this approach is based on the following pillars:

- The segment annotation problem should be evaluated as a single task [28]. While dividing the problem into microtasks is useful to facilitate the user and machine interaction, they should not be treated independently since the final goal is the complete image annotation.
- The human is the protagonist in the process, as described in the VIAL process [29], deciding which action minimizes user effort for image annotation while the machine assists in well-defined microtasks.
- The annotation in the image domain is burdensome; thus, it should be avoided, but not neglected, since perfect segmentation is still an unrealistic assumption.
- The machine should assist the user initially, even when no annotated examples are present [30], and as the annotation progresses, labeling should get easier because more information is provided.

3.1. Overview

The proposed methodology is summarized in Figure 2 and each component is described in the subsequent sections.

The user interface is composed of two primary components, the Projection View and the Image View. Red contours in Figure 2 delineate which functionalities are present in these widgets. The Projection View is concerned with displaying the segments arranged in a canvas (Figure 1), enabling the user to interact with it: assigning labels to clusters, focusing on cluttered regions, and selecting samples for segmentation inspection and correction in the image domain.

Image View displays the image containing a segment selected in the canvas. It is highlighted to allow fast component recognition among the other segments' contours.

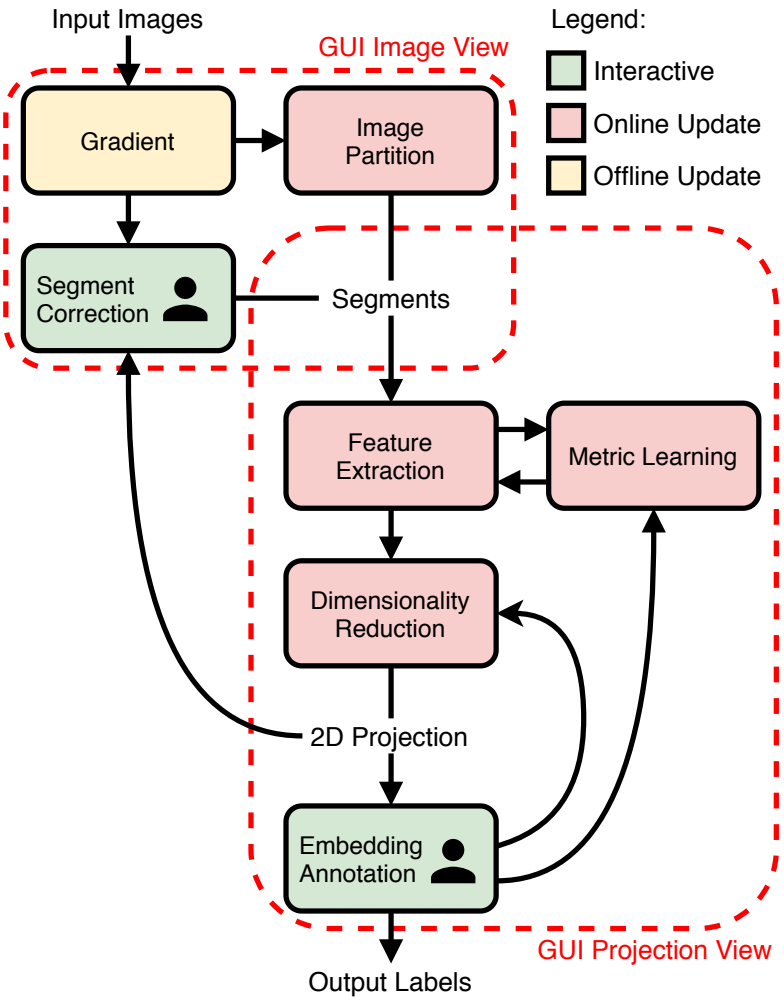


Figure 2: The proposed feature space annotation pipeline.

Samples already labeled are colored by class. This widget allows further user interaction to fix erroneous delineation, as further discussed in Section 3.7. Segment selection also works backwardly, when a region is selected in the Image View, its feature space neighborhood is focused on the projection canvas, accelerating the search for relevant clusters by providing a mapping from the image domain to the projection canvas.

The colored rectangles in Figure 2 represent data processing stages: yellow represents fixed operations that are not updated during user interaction, red elements are updated as the user annotation progresses, and the greens are the user interaction modules. Arrows show how the data flows in the pipeline.

The pipeline works as follows, starting from a collection of images, their gradients are computed to partition each image into segments, which will be the units processed

and annotated in the next stages.

Since we wish to cluster together similar segments, we must define a similarity criterion. Therefore, for each segment, we obtain their deep representation (*i.e*. features). Their Euclidean distances are used to express this information — they are more dissimilar as they are further apart in the feature space.

The next step concerns the notion of similarity between segments as presented and perceived by the user. We propose communicating this information to the user by displaying samples with similar examples in the same neighborhood. Hence, the segments' features are used to project them into the 2D plane while preserving, as best as possible, their relative feature space distances.

The user labeling process is executed in the 2D canvas by defining a bounding-box and assigning the selected label to the segments inside it. As the labeling progresses, their deep representation is updated using metric learning, improving class separability, enhancing the 2D embedding, thus, reducing the annotation effort. We refer to [29] for a review in visual interactive labeling and [34] for interactive dimensionality reduction systems.

This pipeline relies only upon the assumption that it is possible to find meaningful candidate segments from a set of images and extract discriminant features from them to cluster together similar segments. Even though these problems are not solved yet, existing methods can satisfy these requirements, as they are validated in our study of parts (Section 4.3).

3.2. Gradient and Image Partition

Gradient computation and watershed-based image partition are operations of the first step of the pipeline, obtaining initial candidate segments. In the ideal scenario, the desired regions are represented by a single connected component, requiring no further user interaction besides labeling.

However, obtaining meaningful regions is a challenging and unsolved problem. Desired segments vary from application to application. On some occasions, users wish to segment humans and vehicles in a scene, while in the same image, other users may desire to segment the clothes and billboards. Thus, the proposed approach has to be class agnostic and enables the user to obtain different segment categories without effort.

The usual approach computes several solutions (*e.g*. Multiscale Combinatorial Grouping (MCG) [39]) and employs a selection policy to obtain the desired segments. However, it does not guarantee disjoint regions, and it generates thousands of candidates, further complicating the annotation process.

Given that segments should be disjoint, and they are also task dependent, we chose to employ hierarchical segmentation techniques for this stage. Most of the relevant literature for this problem aims at obtaining the best gradient (edge saliency) to compute segmentation.

In Convolutional Oriented Boundaries [40], a CNN predicts multiple boundaries in multiple scales and orientations and are combined into an ultrametric contour map to perform the hierarchical segmentation – Refer to [41, 42] to review the duality between contours and hierarchies.

Recently, various CNN architectures for edge estimation were developed [43, 44, 45, 46, 47], where the multi-scale representation is fused into a single output image.

In such methods, boundaries that appear in finer scales present lower values than the clearest ones.

While segmentation can be computed from the gradient information directly, performing it in learned features has shown to be competitive. Isaacs *et al*. [48] propose a novel approach to use deep features to improve class agnostic segmentation, learning a new mapping for the pixels values, without considering any class information and further separating their representation apart.

Other tasks also employ edge estimation to enhance their performance, notably PoolNet [49] switches between saliency object prediction and edge estimation in the training loop with the same architecture to obtain saliency with greater boundary adherence. We noticed that this approach produces less irrelevant boundaries for image segmentation; thus, we chose this method for our experimental setup.

We opted to employ the flexible hierarchical watershed framework [50, 51] for delineating candidate segments on the gradient image estimated by PoolNet [49] architecture. The hierarchical watershed allows manipulation of the region merging criterion, granting the ability to rapidly update the segments' delineation. Besides, hierarchical segmentation lets the user update segments without much effort (*e.g.*, obtaining a more refined segmentation by reducing the threshold, but as a trade-off, the number of components increases). Further, a watershed algorithm can also interactively correct the delineated segments (Section 3.7).

Starting from a group of N images without annotations, $\{I_1, I_2, \ldots, I_N\}$, their gradient images are computed. For each gradient G_i, $i \in [1, N]$, its watershed hierarchy is built and disjoint segments $\{S_{i,1}, \ldots, S_{i,n_i}\}$ are obtained by thresholding the hierarchy. The required parameters (threshold and hierarchy criterion) are robust and easy to be defined by visual inspection on a few images. More details about that are presented in Section 3.7.

3.3. Feature Extraction

Before presenting the regions arranged by similarity, a feature space representation where dissonant samples are separated must be computed. For that, we refer to CNN architectures for image classification tasks, without their fully connected layers [52] used for image classification.

Each segment is treated individually; we crop a rectangle around the segment in the original image, considering an additional border to not impair the network's receptive field. In this rectangle, pixels that do not belong to the segment are zeroed out. Otherwise, segments belonging to the same image would present similar representations. The segment images are then resized to 224×224 and forwarded through the network, which outputs a high-dimensional representation, $\phi_{i,j}$. In this instance $\phi_{i,j} \in \mathbb{R}^{2048}$, for each $S_{i,j}$. We noticed that processing the segment images without resizing them did not produce significant benefits and restricted the use of large batches' efficient inference.

Since our focus is on image annotation, where labeled data might not be readily available, feature extraction starts without fine-tuning. It is only optimized as the labeling progresses. Any CNN architecture can be employed, but performance is crucial. We use the High-Resolution Network (HRNet) [5] architecture, pre-trained on ImageNet without the fully connected layers. It is publicly available with multiple depths,

and its performance is superior to other established works for image classification, such as ResNet [1]. During the development of this work, other architectures were proposed [53, 54], significantly improving the classification performance while using comparable computing resources. They were not employed in our experimental setup, but it might improve our results.

3.4. Dimensionality Reduction

Dimensionality reduction aims at reducing a feature space from a higher to a lower dimension with similar characteristics. Some methods enforce the global structure (*e.g.* PCA); other approaches, such as non-linear methods, focus on local consistency, penalizing neighborhood disagreement between the higher and lower dimensional spaces.

In some applications, the reduction aims to preserve the original features' characteristics; in our case, we wish to facilitate the annotation as much as possible. Hence, a reduction that groups similar segments and segregates dissonant examples is more beneficial than preserving the original information.

The t-SNE [55] algorithm is the most used technique for non-linear dimensionality reduction. It projects the data into a lower-dimensional space while minimizing the divergence between the higher- and lower-dimensional neighborhood distributions. However, we employed a more recent approach, known as UMAP [56], for the following reasons: the projection is computed faster, samples can be added without fitting the whole data, its parameters seems to provide more flexibility to choose the projection scattering — enforcing local or global coherence — and most importantly, it allows using labeled data to enforce consistency between samples of the same class while still allowing unlabeled data to be inserted.

Note that dimensionality reduction is critical to the whole pipeline because it arranges the data to be presented to the user, where most of the interaction will occur.

The 2D embedding can produce artifacts, displaying distinct segments clustered together due to the trade-off between global and local consistency even though they might be distant in the higher-dimensional feature space. Therefore, the user can select a subset of samples and interact with their local projection in a pop-up window, where the projection parameter is tuned to enforce local consistency. The locally preserving embedding (Figure 3) separates the selected cluttered segments (in pink) into groups of similar objects (tennis court, big households, small households, etc.), making it easier for label assignment.

On images, CNN's features obtain remarkable results consistent with the human notion of similarity between objects. Considering that an annotator evaluates images visually, sample projection is our preferred approach to inform the user about possible clusters, as explained in the next section. Other visualizations must be explored for other kinds of data, such as sound or text, where the user would have difficulty to visually exploit the notion of similarity [29, 34].

3.5. Embedding Annotation

Each segment is displayed on their 2-dimensional coordinate, as described in the previous section. To annotate a set of segments, the user selects a bounding-box around them in the canvas, assigning the designed label. Hence, each $S_{i,j}$ inside the defined

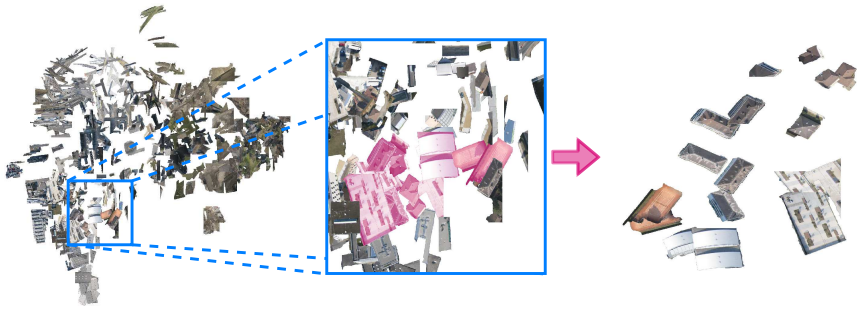


Figure 3: Local re-projection example: Global projection with a region highlighted in blue; A subset of segments is selected by the user, in pink; Their local embedding is computed for a simpler annotation.

box is assigned to a label $L_{i,j}$. Finally, to obtain annotated masks in the image domain, the label $L_{i,j}$ of a segment $S_{i,j}$ is mapped to its pixels in I_i, thus resulting in an image segmentation (pixel annotation).

Due to several reasons, such as spurious segments or over-segmented objects, a region could be indistinguishable. Hence, when a single segment is selected in the projection, its image is displayed in the Image View as presented earlier. This action also works backwardly, the user can navigate over the images, visualize the current segments, and upon selection, the segments are focused in the projection view. Thus, avoiding the effort of searching individual samples in the segment scattering.

Additional care is necessary when presenting a large number of images, mainly if each one contains several objects, because the number of segments displayed on the canvas may impair the user's ability to distinguish their respective classes for annotation. Therefore, only a subset of the data is shown to the user initially. Additional batches are provided as requested while the labeling and the embedding progress, reducing the annotation burden.

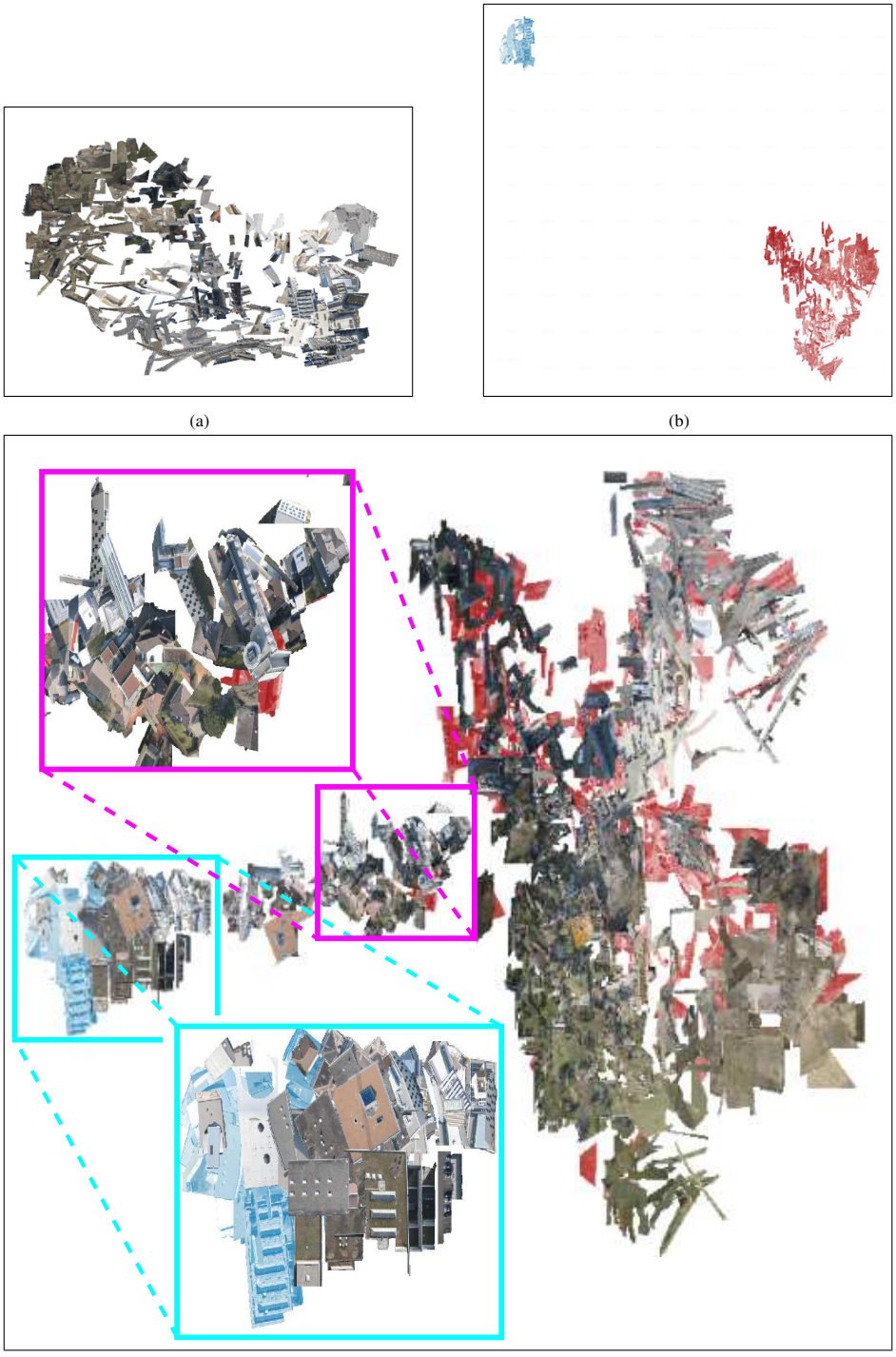
3.6. Metric Learning

The proposed pipeline is not specific to any objects' class and does not require pre-training, but as the annotation progresses, the available labels can be employed to reduce user effort — less effort is necessary when the clusters are homogeneous and not spread apart.

For that, we employ a *metric learning* algorithm. Figure 4 shows an example of how metric learning can make clusters of a same class more compact and better separate clusters from distinct classes in our application.

Initially, metric learning methods were concerned with finding a metric where some distance-based (or similarity) classification [57, 58] and clustering [59] would be optimal, in the sense that samples from the same class should be closer together than adversary examples. Given some regularity conditions, learning this new metric is equivalent to embedding the data into a new space.

The metric learning objective functions can be grossly divided into two main varieties, soft assignment and triplet-based techniques. The former, as proposed in NCA [58], maximizes a soft-neighborhood assignment computed through the soft-max function



(a) (b)

(c)

Figure 4: Example of metric learning in the Rooftop dataset: (a) Segment arrangement from an initial batch (10 images). (b) Displacement after labeling and employing metric learning, foreground (rooftops) in blue and background in red. (c) Projection with additional unlabeled data (plus 20 images). Most rooftops' segments are clustered together (the cyan box). The magenta box indicates clusters from mixed classes and spurious segments. They suggest where labels are required the most for a next iteration of labeling and metric learning. The remaining clusters can be easily annotated.

over the negative distance between the data points, penalizing label disagreement of immediate neighbors more than samples further apart. Triplet-based methods [57] select two examples from the same class and minimize their distance while pushing away a third one from a different class when it violates a threshold given the pair distance. Thus, avoiding unnecessary changes when a neighborhood belongs to a single class.

More recently, these methods began to focus mostly on improving embedding through neural networks rather than on the metric centered approach. Notably, in computer vision, it gained traction in image retrieval tasks with improvements in triplet mining methods [60, 61] and novel loss functions [62, 63].

In our pipeline, the original large-margin loss [57] was employed, using our previously mentioned feature extractor network, due to its excellent performance with only a single additional parameter. We follow here Musgrave *et al*. [64], which showed that some novel methods are prone to overfitting and require more laborious parameter tuning.

3.7. Segment Correction

Since segment delineation is not guaranteed to be perfect, component correction is crucial, especially for producing ground-truth data, where pixel-level accuracy is of uttermost importance. Hence, segments containing multiple objects (under segmentation) are corrected by positive and negative clicks, splitting the segment into two new regions according to the user's positive and negative cues.

Here we use a classical graph-based algorithm as we focus mainly on the feature space annotation; more modern CNN-based approaches can be employed on a real scenario.

Given an under-segmented region $S_{i,j}$, we define an undirected graph $\mathcal{G} = (V, E, w)$ where the vertices V are the pixels in $S_{i,j}$, the edges connect 4-neighbors, constrained to be inside $S_{i,j}$, and each edge $(p, q) \in E$ is weighted by $w(p, q) = (G_{i,j}(p) + G_{i,j}(q))/2$. A segment partition is obtained by the image foresting transform [27] algorithm for the labeled watershed operator given two sets of clicks C_{pos} and C_{neg} as defined by the user. This operation offers full control over segmentation, is fast, and improves segment delineation.

Further, the user can change the hierarchical criterion for watershed segmentation, preventing interactive segment correction in multiple images. For example, in an image overcrowded with irrelevant small objects, the user can bias the hierarchy to partition larger objects by defining the hierarchy ordering according to the objects' area in the image domain, filtering out the spurious segments.

Therefore, the Image View interface allows inspection of multiple hierarchical segmentation criteria and their result for given a threshold. The segments are recomputed upon user confirmation, maintaining the labels of unchanged segments. Novel segments go through the pipeline for feature extraction and projection into the canvas.

4. Experiments

This section starts by describing the datasets chosen for the experiments and the implementation details for our approach. Since our method partitions the images into

segments and solves the simultaneous annotation of multiple objects by interactive labeling of similar segments' clusters, we present a study of two ideal scenarios to evaluate each main step individually. In the first experiment, the images are partitioned into perfect segments constrained to each object's mask, the rest of the pipeline is executed as proposed, evaluating the feature space annotation efficiency. Next, we assess the image partition by assuming optimal labeling, thus, only investigating the initial segmentation performance. Hence, the study of parts evaluates the limitations of the initial unsupervised hierarchical segmentation techniques, description with metric learning, and projection. Subsequently, we compare with state-of-the-art methods and present the quantitative and qualitative results. The supplementary materials include videos of the UI usage and annotation experiments.

Note that our goal goes beyond showing that the proposed method can outperform others. We are pointing a research direction that exploits new ways of human-machine interaction for more effective data annotation.

Typically, a robot user executes the deep interactive segmentation experiments; in contrast, our study is conducted by a volunteer with experience in interactive image segmentation. Thus, we are taking into account the effort required to locate and identify objects of interest.

4.1. Datasets

We selected image datasets from video segmentation, co-segmentation, and semantic segmentation tasks, in which the objects of interest are to some extent related.

1. *CMU-Cornell iCoSeg* [65]: It contains 643 natural images divided into 38 groups. Within a group, the images have the same foreground and background but are seen from different point-of-views.
2. *DAVIS* [66]: It is a video segmentation dataset containing 50 different sequences. Following the same procedure as in [16], multiple objects in each frame were treated as a single one, and the same subset of 345 frames (10 % of the total) was employed.
3. *Rooftop* [67]: It is a remote sensing dataset with 63 images, and in total containing 390 instances of disjoint rooftop polygons.
4. *Cityscapes* [68]: It is a semantic segmentation dataset for autonomous driving research. It contains video frames from 27 cities divided into 2975 images for training, 500 for validation, and 1525 for testing. The dataset contains 30 classes (*e.g.* roads, cars, trucks, poles), we evaluated using only the 19 default classes.

4.2. Implementation details

We implemented a user interface in Qt for Python. To segment images into components, we used Higra [69, 70] and the image gradients generated with PoolNet [71]. We computed gradients over four scales, 0.5, 1, 1.5, and 2, and averaged their output to obtain a final gradient image. For the remaining operations, including the baselines, we used NumPy [72], the PyTorch Metric Learning package [73] and the available implementations in PyTorch [74].

Dataset	Avg. IoU	Median IoU	Time (s)
iCoSeg	95.07	99.96	5.32
DAVIS	98.54	99.97	7.82
Rooftop	95.10	99.99	3.96

Table 1: Average IoU, median IoU, average total processing time in seconds per image.

For segment description with metric learning, we used the publicly available *HRNet-W18-C-Small-v1* [75] configuration pre-trained on the ImageNet dataset. In the metric-learning stage, the Triplet-Loss margin is fixed at 0.05. At each call, the embedding is optimized through Stochastic Gradient Descent (SGD) with momentum of 0.8 and weight decay of 0.0005 over three epochs with 1000 triplets each. The learning rate starts at 0.1 and, at each epoch, it is divided by 10.

We used UMAP [56] with 15 neighbors for feature projection and a minimum distance of 0.01 in the main canvas. The zoom-in canvas used UMAP with five neighbors and 0.1 minimum distance; when labels were available, the semi-supervised trade-off parameter was fixed at 0.5, penalizing intra-class and global consistencies equally.

4.3. Study of Parts

Our approach depends on two main independent steps: the image partition into segments and the interactive labeling of those regions. The inaccuracy of one of them would significantly deteriorate the performance of the feature space annotation for image segmentation labeling. Therefore, we present a study of parts that considers two ideal scenarios: (a) interactive projection labeling of perfect segments and (b) image partition into segments followed by optimal labeling.

In (a), the user annotates segments from a perfect image partition inside and outside the objects' masks. Hence, every segment will always belong to a single class. Table 1 reports the results. The Intersection over Union (IoU) distribution is heavily right-skewed, as noticed from the differences between average IoU and median IoU, indicating that most segments were labeled correctly. Visual inspection revealed that user annotation errors occurred only in small components. Table 1 also reports the total time (in seconds) spent annotating (user) and processing (machine), starting from the initial segment projection presented to the user. It indicates that feature space projection annotation with metric learning is effective for image segmentation annotation.

In (b), we measure the IoU of the watershed hierarchical cut using a fixed parameter — the threshold of 1000 with the volume criterion. The segments were then labeled by majority vote among the true labels of their pixels. Table 2 shows the quality of segmentation, which imposes the upper bound to the quality of the overall projection labeling procedure if no segment correction was performed.

For reference, Click Carving [76] reports an average IoU of 84.31 in the iCoSeg, dataset when selecting the optimal segment (*i.e*. highest IoU among proposals) from a pool of approximately 2000 segment proposals per image, produced with MCG [39]. In contrast, we obtain an equivalent performance of 84.15 with a fixed segmentation with disjoints candidates only.

Figure 5 presents example of the candidate segments on the three datasets.

Dataset	Avg. IoU	Median IoU
iCoSeg	84.15	91.86
DAVIS	82.46	88.50
Rooftop	75.14	76.77

Table 2: Automatic segmentation results with their respective dataset.

Figure 5: Candidate segments from the study of parts. First row is from the iCoSeg dataset, second is from DAVIS and the last from Rooftop.

4.4. Quantitative analysis using baselines

Since existing baselines report scores only in a very limited scenario, we executed our own experiments according to the code availability; Them being, f-BRS-B [17] (CVPR2020) and FCANet [14] (CVPR2020), both with *Resnet101* backbone, with their publicly available weights [78, 79] trained on the SBD [80] and SBD plus PASCAL VOC [81] datasets, respectively. We are not comparing with IOG [15] (CVPR2020) because we could not reproduce the results (subpar performance) with their available code and weights, and [18] (ECCV2020) is not publicly available. The results are reported over the final segmentation mask, given a sequence of 3 and 5 clicks.

Table 3 report the average IoU and the total time spent in annotation. For click-based methods, the interaction time was estimated as 2.4s for the initial click and 0.9s for additional clicks [77].

Dataset	iCoSeg		DAVIS		Rooftop	
Method	IoU	Time (s)	IoU	Time (s)	IoU	Time (s)
f-BRS (3 clicks)	79.82	4.2	79.87	4.2	62.57	4.2
f-BRS (5 clicks)	82.14	6	82.44	6	74.53	6
FCANet (3 clicks)	<u>84.63</u>	4.2	82.44	4.2	65.99	4.2
FCANet (5 clicks)	**88.00**	6	**86.63**	6	**81.38**	6
Ours	84.29	5.96	<u>84.53</u>	8.74	<u>77.28</u>	7.02

Table 3: Average IoU and time over images, except for Rooftop, where time is computed over instances. For robot user experiments, with multiple budgets (3 and 5 clicks), time was estimated according to this study [77]. Our method obtains comparable accuracy, but it spends additional time annotating foreground and background. The Cityscapes experiment shows our results on a more realistic scenario where every pixel is labeled (not a microtask).

We achieve comparable accuracy results with state-of-the-art methods while employing less sophisticated segmentation procedures, qualitative results are presented in Figures 6- 8. Despite this, existing methods require less time to annotate these datasets; this is due to them being specialized in the foreground annotation microtask, while our approach wastes time annotating the background — this is exacerbated on the DAVIS dataset where a background object might be a category equal to the foreground.

The following experiment evaluates our performance on a semantic segmentation, where labeling the whole image is the final goal, not just the microtask of delineating a single object.

4.5. Qualitative analysis for semantic segmentation

To verify the proposed approach in a domain-specific scenario, we evaluate its performance on Cityscapes [68]. Since the true labels of the test set are not available, we took the same approach as [22], by testing on the validation set. Further, the annotation quality was evaluated on 98 randomly chosen images (about 20% of the validation set).

PoolNet was optimized based on the boundaries of the training set's true labels for five epochs using the Adam optimizer, a learning rate of $5e^{-5}$, weight decay of $5e^{-4}$, and a batch of size 8. Predictions were performed on a single scale. The fine-tuned gradient ignores irrelevant boundaries, reducing over segmentation.

The original article reports an agreement (*i.e*. accuracy) between annotators of 96%. We obtained an agreement of 91.5% with the true labels of the validation set (Figure 9), while spending less than 1.5% of their time — *i.e*. our experiment took 1 hour and 58 minutes to annotate the 98 images, while to produce the same amount of ground-truth data took approximately 6.1 days (average of 1.5 hour per image [68]) — about 74.75 times faster than the original procedure. These 98 images contain in total 6500 default classes' polygons (*i.e*., instances). Thus, with the estimate of 6 secs per instance, FCANet would take 10 hours and 50 minutes to label them.

Figure 6: The magenta boundaries delineate regions with foreground labels for the ground-truth data, our method, and the baselines using 5 clicks per image on the iCoSeg dataset. Figure (i) shows that FCANet has difficulties when segmenting multiples instances, as mentioned in their original article.

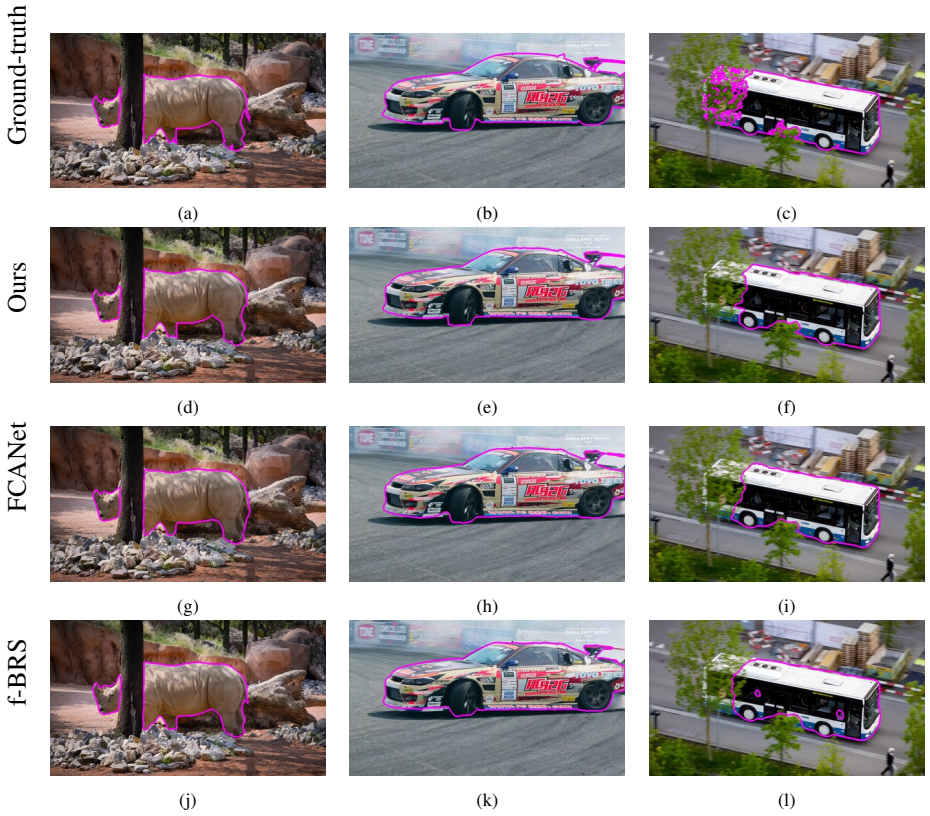


Figure 7: The magenta boundaries delineate regions with foreground labels for the ground-truth data, our method, and the baselines using 5 clicks per image on the DAVIS dataset.

Figure 8: The magenta boundaries delineate regions with foreground labels for the ground-truth data, our method, and the baselines using 5 clicks per instance on the Rooftop dataset.

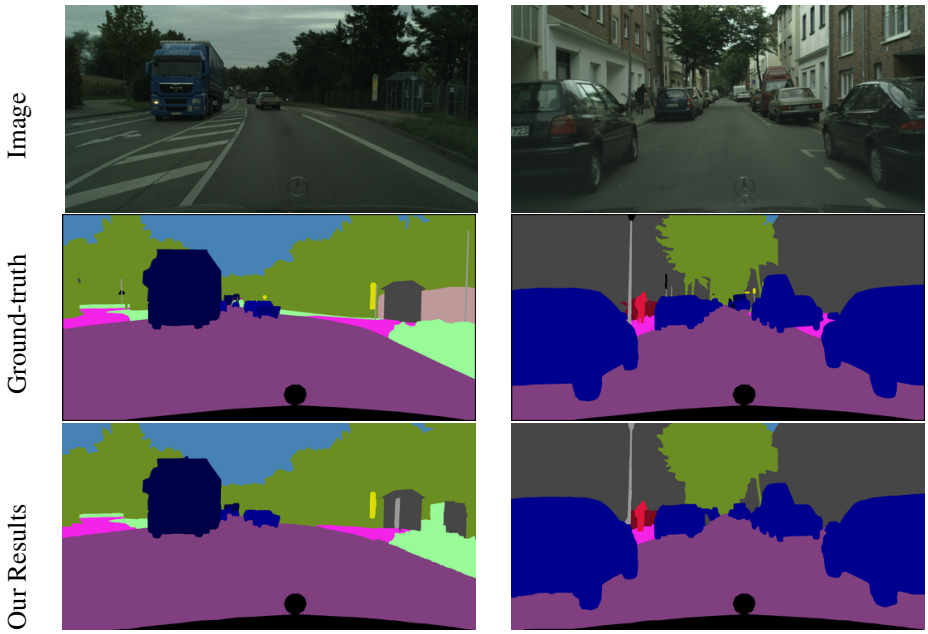


Figure 9: Cityscapes result, each column is a different image, row indicates which kind.

5. Discussion

We presented a novel interactive image segmentation paradigm for simultaneous annotation of segments from multiple images in their deep features' projection space. Despite employing less sophisticated segmentation methods, it achieves comparable performance with more modern approaches. Moreover, individual modules can be swapped to obtain optimal performance (*e.g.*, FCANet for segmentation correction, unifying the embedding and feature extraction with Parametric UMAP [82]).

We think that this work leads to several opportunities for combining the whole pipeline into a holistic segmentation procedure, where redundant samples are labeled at once, and annotation on the image domain occurs only when necessary. In a real scenario, we suggest letting a leading user interact with the projection to evaluate existing annotated data, model performance (segments clustering), and, when necessary, delegating segment correction to workers, as it is currently done in existing annotation procedures, diminishing the total images annotated on the image domain.

Acknowledgements

This work was supported by CAPES-COFECUB [#42067SF]; CNPq [#303808/2018-7]; and FAPESP research grants [#2014/12236-1, #2019/11349-0, and #2019/21734-9];

References

[1] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[2] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: IEEE International Conference on Computer Vision, 2017, pp. 2961–2969.

[3] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Communications of the ACM 60 (6) (2017) 84–90.

[4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, IEEE Transactions on Pattern Analysis and Machine Intelligence 40 (4) (2018) 834–848.

[5] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, et al., Deep high-resolution representation learning for visual recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence.

[6] 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.

[7] A. X. Falcão, J. K. Udupa, S. Samarasekera, S. Sharma, B. E. Hirsch, R. A. Lotufo, User-steered image segmentation paradigms: Live wire and live lane, Graphical models and image processing 60 (4) (1998) 233–260.

[8] Y. Boykov, V. Kolmogorov, An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (9) (2004) 1124–1137.

[9] C. Rother, V. Kolmogorov, A. Blake, Grabcut: Interactive foreground extraction using iterated graph cuts, in: ACM Trans. on Graphics, Vol. 23, 2004, pp. 309–314.

[10] L. Grady, Random walks for image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (11) (2006) 1768–1783.

[11] M. Tang, L. Gorelick, O. Veksler, Y. Boykov, Grabcut in one cut, in: IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1769–1776.

[12] B. C. Russell, A. Torralba, K. P. Murphy, W. T. Freeman, Labelme: a database and web-based tool for image annotation, International Journal of Computer Vision 77 (1-3) (2008) 157–173.

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

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

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

[16] W.-D. Jang, C.-S. Kim, Interactive image segmentation via backpropagating refinement scheme, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 5297–5306.

[17] K. Sofiiuk, I. Petrov, O. Barinova, A. Konushin, f-brs: Rethinking backpropagating refinement for interactive segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 8623–8632.

[18] T. Kontogianni, M. Gygli, J. Uijlings, V. Ferrari, Continuous adaptation for interactive object segmentation by learning from corrections, in: IEEE European Conference on Computer Vision, Springer, 2020, pp. 579–596.

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

[20] K.-K. Maninis, et al., Deep extreme cut: From extreme points to object segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[21] R. Benenson, S. Popov, V. Ferrari, Large-scale interactive object segmentation with human annotators, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 11700–11709.

[22] L. Castrejon, K. Kundu, R. Urtasun, S. Fidler, Annotating object instances with a polygon-rnn, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5230–5238.

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

[24] H. Ling, J. Gao, A. Kar, W. Chen, S. Fidler, Fast interactive object annotation with curve-gcn, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 5257–5266.

[25] E. Agustsson, J. R. Uijlings, V. Ferrari, Interactive full image segmentation by considering all regions jointly, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 11622–11631.

[26] C. Couprie, L. Grady, L. Najman, H. Talbot, Power watershed: A unifying graph-based optimization framework, IEEE Transactions on Pattern Analysis and Machine Intelligence 33 (7) (2011) 1384–1399.

[27] A. X. Falcão, J. Stolfi, R. A. de Lotufo, The image foresting transform: Theory, algorithms, and applications, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (1) (2004) 19–29.

[28] M. Andriluka, J. R. Uijlings, V. Ferrari, Fluid annotation: a human-machine collaboration interface for full image annotation, in: Proceedings of the 26th ACM international conference on Multimedia, 2018, pp. 1957–1966.

[29] J. Bernard, M. Zeppelzauer, M. Sedlmair, W. Aigner, Vial: a unified process for visual interactive labeling, The Visual Computer 34 (9) (2018) 1189–1207.

[30] J. Bernard, M. Hutter, M. Zeppelzauer, D. Fellner, M. Sedlmair, Comparing visual-interactive labeling with active learning: An experimental study, IEEE Transactions on Visualization and Computer Graphics 24 (1) (2017) 298–308.

[31] B. C. Benato, A. C. Telea, A. X. Falcão, Semi-supervised learning with interactive label propagation guided by feature space projections, in: 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), IEEE, 2018, pp. 392–399.

[32] B. C. Benato, J. F. Gomes, A. C. Telea, A. X. Falcão, Semi-automatic data annotation guided by feature space projection, Pattern Recognition 109 (2019) 107612.

[33] J. E. Vargas-Muñoz, P. Zhou, A. X. Falcão, D. Tuia, Interactive coconut tree annotation using feature space projections, in: IEEE International Geoscience and Remote Sensing Symposium, IEEE, 2019, pp. 5718–5721.

[34] D. Sacha, L. Zhang, M. Sedlmair, J. A. Lee, J. Peltonen, D. Weiskopf, S. C. North, D. A. Keim, Visual interaction with dimensionality reduction: A structured literature analysis, IEEE Transactions on Visualization and Computer Graphics 23 (1) (2016) 241–250.

[35] N. Araslanov, S. Roth, Single-stage semantic segmentation from image labels, in: IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 4253–4262.

[36] Y. Zhou, Y. Zhu, Q. Ye, Q. Qiu, J. Jiao, Weakly supervised instance segmentation using class peak response, in: IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3791–3800.

[37] Y. Zhu, Y. Zhou, H. Xu, Q. Ye, D. Doermann, J. Jiao, Learning instance activation maps for weakly supervised instance segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3116–3125.

[38] K.-J. Hsu, Y.-Y. Lin, Y.-Y. Chuang, Deepco3: Deep instance co-segmentation by co-peak search and co-saliency detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 8846–8855.

[39] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, J. Malik, Multiscale combinatorial grouping for image segmentation and object proposal generation, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (1) (2016) 128–140.

[40] K.-K. Maninis, J. Pont-Tuset, P. Arbeláez, L. Van Gool, Convolutional oriented boundaries: From image segmentation to high-level tasks, IEEE Transactions on Pattern Analysis and Machine Intelligence 40 (4) (2017) 819–833.

[41] L. Najman, M. Schmitt, Geodesic saliency of watershed contours and hierarchical segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (12) (1996) 1163–1173.

[42] L. Najman, On the equivalence between hierarchical segmentations and ultrametric watersheds, Journal of Mathematical Imaging and Vision 40 (3) (2011) 231–247.

[43] S. Xie, Z. Tu, Holistically-nested edge detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1395–1403.

[44] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, X. Bai, Richer convolutional features for edge detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3000–3009.

[45] R. Deng, C. Shen, S. Liu, H. Wang, X. Liu, Learning to predict crisp boundaries, in: IEEE European Conference on Computer Vision, 2018, pp. 562–578.

[46] J. He, S. Zhang, M. Yang, Y. Shan, T. Huang, Bi-directional cascade network for perceptual edge detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3828–3837.

[47] D. Linsley, J. Kim, A. Ashok, T. Serre, Recurrent neural circuits for contour detection, in: International Conference on Learning Representations, 2019.

[48] O. Isaacs, O. Shayer, M. Lindenbaum, Enhancing generic segmentation with learned region representations, in: IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 12946–12955.

[49] J.-J. Liu, Q. Hou, M.-M. Cheng, J. Feng, J. Jiang, A simple pooling-based design for real-time salient object detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3917–3926.

[50] J. Cousty, L. Najman, Y. Kenmochi, S. Guimarães, Hierarchical segmentations with graphs: quasi-flat zones, minimum spanning trees, and saliency maps, Journal of Mathematical Imaging and Vision 60 (4) (2018) 479–502.

[51] J. Cousty, L. Najman, Incremental algorithm for hierarchical minimum spanning forests and saliency of watershed cuts, in: Mathematical Morphology and Its Applications to Signal and Image Processing, Springer, 2011, pp. 272–283.

[52] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[53] M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: International Conference on Machine Learning, 2019, pp. 6105–6114.

[54] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, P. Dollár, Designing network design spaces, in: IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 10428–10436.

[55] L. v. d. Maaten, G. Hinton, Visualizing data using t-sne, Journal of Machine Learning Research 9 (Nov) (2008) 2579–2605.

[56] L. McInnes, J. Healy, J. Melville, Umap: Uniform manifold approximation and projection for dimension reduction, arXiv preprint arXiv:1802.03426.

[57] K. Q. Weinberger, L. K. Saul, Distance metric learning for large margin nearest neighbor classification., Journal of Machine Learning Research 10 (2).

[58] J. Goldberger, G. E. Hinton, S. Roweis, R. R. Salakhutdinov, Neighbourhood components analysis, Advances in Neural Information Processing Systems 17 (2004) 513–520.

[59] E. P. Xing, M. I. Jordan, S. J. Russell, A. Y. Ng, Distance metric learning with application to clustering with side-information, in: Advances in Neural Information Processing Systems, 2003, pp. 521–528.

[60] A. Hermans, L. Beyer, B. Leibe, In defense of the triplet loss for person re-identification, arXiv preprint arXiv:1703.07737.

[61] W. Zheng, Z. Chen, J. Lu, J. Zhou, Hardness-aware deep metric learning, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 72–81.

[62] C.-Y. Wu, R. Manmatha, A. J. Smola, P. Krahenbuhl, Sampling matters in deep embedding learning, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2840–2848.

[63] Q. Qian, L. Shang, B. Sun, J. Hu, H. Li, R. Jin, Softtriple loss: Deep metric learning without triplet sampling, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 6450–6458.

[64] K. Musgrave, S. Belongie, S.-N. Lim, A metric learning reality check, in: IEEE European Conference on Computer Vision, Springer, 2020.

[65] D. Batra, A. Kowdle, D. Parikh, J. Luo, T. Chen, Interactively co-segmentating topically related images with intelligent scribble guidance, International Journal of Computer Vision 93 (3) (2011) 273–292.

[66] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, A. Sorkine-Hornung, A benchmark dataset and evaluation methodology for video object segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 724–732.

[67] X. Sun, C. M. Christoudias, P. Fua, Free-shape polygonal object localization, in: IEEE European Conference on Computer Vision, Springer, 2014, pp. 317–332.

[68] 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: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3213–3223.

[69] B. Perret, G. Chierchia, J. Cousty, S. Guimarães, Y. Kenmochi, L. Najman, Higra: Hierarchical graph analysis, SoftwareX 10 (2019) 100335.

[70] "Higra: Hierarchical graph analysis" package., `https://higra.readthedocs.io/en/stable/`, last Accessed: 2020-11-06.

[71] Repository of "a simple pooling-based design for real-time salient object detection", `https://github.com/backseason/PoolNet`, last Accessed: 2020-11-06.

[72] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, et al., Array programming with numpy, Nature 585 (7825) (2020) 357–362.

[73] K. Musgrave, S. Belongie, S.-N. Lim, PyTorch metric learning (2020). arXiv:2008.09164.

[74] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Pytorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems, 2019, pp. 8026–8037.

[75] Repository of "high-resolution networks (HRNets for image classification", `https://github.com/HRNet/HRNet-Image-Classification`, last Accessed: 2020-11-06.

[76] S. D. Jain, K. Grauman, Click carving: Interactive object segmentation in images and videos with point clicks, International Journal of Computer Vision 127 (9) (2019) 1321–1344.

[77] A. Bearman, O. Russakovsky, V. Ferrari, L. Fei-Fei, What's the point: Semantic segmentation with point supervision, in: IEEE European Conference on Computer Vision, Springer, 2016, pp. 549–565.

[78] Repository of "f-BRS: Rethinking backpropagating refinement for interactive segmentation repository", `https://github.com/saic-vul/fbrs_interactive_segmentation`, last Accessed: 2020-11-06.

[79] Repository of "interactive image segmentation with first click attention", https://github.com/frazerlin/fcanet, last Accessed: 2021-03-10.

[80] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, J. Malik, Semantic contours from inverse detectors, in: IEEE International Conference on Computer Vision, IEEE, 2011, pp. 991–998.

[81] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, International Journal of Computer Vision 88 (2) (2010) 303–338.

[82] T. Sainburg, L. McInnes, T. Q. Gentner, Parametric umap: learning embeddings with deep neural networks for representation and semi-supervised learning, arXiv preprint arXiv:2009.12981.