Bergische Universität Wuppertal

Fakultät für Mathematik und Naturwissenschaften

Institute of Mathematical Modelling, Analysis and Computational Mathematics (IMACM)

Preprint BUW-IMACM 20/44

Pascal Colling, Lutz Roese-Koerner, Hanno Gottschalk, Matthias Rottmann

MetaBox+: A new Region Based Active Learning Method for Semantic Segmentation using Priority Maps

October 12, 2020

http://www.imacm.uni-wuppertal.de

MetaBox+: A new Region Based Active Learning Method for Semantic Segmentation using Priority Maps

Pascal Colling[1,2], **Lutz Roese-Koerner**[1], **Hanno Gottschalk**[2], **Matthias Rottmann**[2]

[1]**Aptiv, Wuppertal, Germany**
[2]**University of Wuppertal, School of Mathematics and Natural Sciences, IMACM & IZMD**
{pascal.colling,lutz.roese-koerner}@aptiv.com.de
{hanno.gottschalk,rottmann}@uni-wuppertal.de

Abstract. We present a novel region based active learning method for semantic image segmentation, called *MetaBox+*. For acquisition, we train a meta regression model to estimate the segment-wise Intersection over Union (IoU) of each predicted segment of unlabeled images. This can be understood as an estimation of segment-wise prediction quality. Queried regions are supposed to minimize to competing targets, i.e., low predicted IoU values / segmentation quality and low estimated annotation costs. For estimating the latter we propose a simple but practical method for annotation cost estimation. We compare our method to entropy based methods, where we consider the entropy as uncertainty of the prediction. The comparison and analysis of the results provide insights into annotation costs as well as robustness and variance of the methods. Numerical experiments conducted with two different networks on the Cityscapes dataset clearly demonstrate a reduction of annotation effort compared to random acquisition. Noteworthily, we achieve 95% of the mean Intersection over Union ($mIoU$), using MetaBox+ compared to when training with the full dataset, with only 10.47% / 32.01% annotation effort for the two networks, respectively.

Key words: active learning • semantic segmentation • cost effective • priority maps via meta regression • cost estimation

1 Introduction

In recent years, semantic segmentation, the pixel-wise classification of the semantic content of images, has become a standard method to solve problems in image and scene understanding [27, 40, 4, 37]. Examples of applications are autonomous driving and environment understanding [40, 4, 37], biomedical analyses [27] and further computer visions tasks. Deep convolutional neural networks (CNN) are commonly used in semantic segmentation. In order to maximize the accuracy of a CNN, a large amount of annotated and varying data is required, since with an increasing number of samples the accuracy increases only logarithmically [35]. For instance in the field of autonomous driving, fully and precisely annotated street scenes require an enormous (and tiring) annotation effort. Also biomedical applications, in general domains that require expert knowledge for annotation, suffer from high annotation costs. Hence, from multiple perspectives (annotation) cost reduction while maintaining model performance is highly desirable.

One possible approach is *active learning* (AL), which basically consists of alternatingly annotating data and training a model with the currently available annotations. The key component in this algorithm that can substantially leverage the learning process is the so called query or acquisition strategy. The ultimate goal is to label the data that leverages the model performance most while paying with as small labeling costs as possible. For an introduction to AL methods, see e.g. [32].

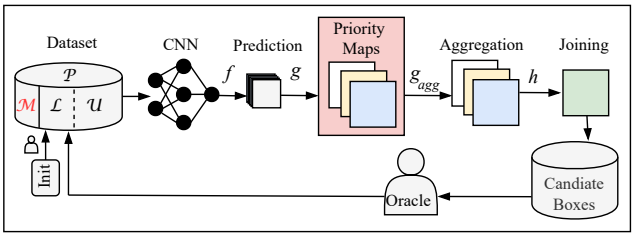

Figure 1. Illustration of the region based AL method. The red parts highlight the novel meta regression based ingredients: as query priority maps we use MetaSeg and the estimated number of clicks. Training of MetaSeg requires an additional small sample of data (fixed for the whole course of AL), indicated by $\mathcal{M} \subset \mathcal{P}$ (in red color).

First AL approaches (before the deep learning (DL) era) for semantic segmentation, for instance based on conditional random fields, go back to [36, 18, 23, 16]. At the heart of an AL method is the so-called *query strategy* that decides which data to present to the annotator / oracle next. In general, uncertainty sampling is one of the most common query strategies [10, 38, 1, 13, 29], besides that there also exist approaches on expected model change [36] and reinforcement learning based AL [2].

In recent years, approaches to deep AL for semantic segmentation have been introduced, primarily for two applications, i.e., biomedical image segmentation and semantic street scene segmentation, cf. [39, 12, 25, 22, 34, 17, 2, 21]. The approaches in [39, 12, 25, 22] are specifically designed for medical and biomedical applications, mostly focusing on foreground-background segmentation. Due to the underlying nature of the data, these approaches refer to annotation costs in terms of the number of labeled images. The methods presented in [34, 17, 2, 21] use region based proposals. All of them evaluate the model accuracy in terms of mean Intersection over Union ($mIoU$). The method in [34] is designed for multi-view semantic segmentation datasets, in which objects are observed from multiple

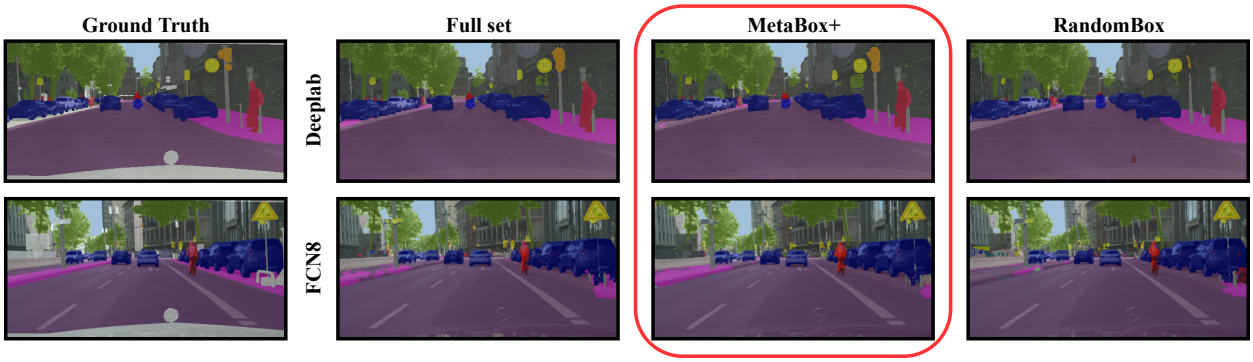

Figure 2. Segmentation results with our novel method MetaBox+ for two CNN models. With an annotation effort of only 10.47% for the Deeplab model *(top)* and 32.01% for the FCN8 model *(bottom)*, we achieve 95% full set $mIoU$. Additionally, the segmentation results are shown when training the networks with the full dataset (Full set) and with a random selection strategy (RandomBox) producing the same annotation effort as MetaBox+. Annotation effort is stated in terms of a click based metric ($cost_A$, Equation (10)).

viewpoints. The authors introduce two new uncertainty based metrics and aggregate them on superpixel (SP) level (SPs can be viewed as visually uniform clusters of pixels). They measure the costs by the number of labeled pixels. Furthermore they have shown that labeling on SP level can reduce the annotation time by 25%. In [17] a new uncertainty metric on SP level is defined, which includes the information of the Shannon Entropy [33], combined with information about the contours in the original image and a class-similarity metric to put emphasis on rare classes. In [17], the number of pixels labeled define the costs. The authors of [2] utilize the same cost metric. The latter work uses reinforcement learning to find the most informative regions, which are given in quadratic format of fixed size. This procedure aims at finding regions containing instances of rare classes. The method in [21] queries quadratic regions of fixed-size from images as well. They use a combination of uncertainty measure (Vote [7] and Shannon Entropy) and a clicks-per-polygon based cost estimation, which is regressed by a second DL model. The methods in [21, 2, 17] are focusing on multi-class semantic segmentation datasets like Cityscapes [6].

In this work, we introduce a query strategy, which is based on an estimation of the segmentation quality. We use a meta regression model, as introduced in [28] and extended in [30, 20, 3], to predict the segment-wise IoU and then aggregate this information over the quadratic candidate regions of images. Furthermore, we introduce a simple and practical method to estimate the annotation costs in terms of clicks. Through the combination of both, we target informative regions with low annotation costs. A sketch of our method is given in Figure 1. Based on the number of clicks required for annotation, we introduce a new cost metric. For numerical experiments we used the Cityscapes dataset [6] with two models, namely the FCN8 [19] and the Deeplabv3+ Xcpetion65 [4] (following in short only Deeplab).

2 Related Work

In this section, we compare our work to the works closest to ours. Therefore, we focus on the region based approaches [34, 17, 2, 21]. All of them evaluate the model accuracy in terms of $mIoU$. The approaches in [17, 34] use handcrafted uncertainty metrics aggregated on SP level and also query regions in form of SP. While [34] is specifically designed for multi-views datasets, our approach focuses on single-view multi-class segmentation of street scenes. For the query strategy presented in [17], the authors do not only use uncertainty metrics, but also information about the contours of the images as well as class similarities of the SP to identify rare classes. We also use different types of information to generate our proposals. The AL method in [2] is based on deep reinforcement learning and queries quadratic fixed size regions. We query the same region format but instead of reinforcement learning, we only use the segmentation network's output in our query strategy. Compared to the approaches above [17, 34, 2] we do not focus on finding a minimal dataset to achieve a satisfying model accuracy. We consider the costs in terms of required clicks for annotating a region and aim to reduce the human annotation effort (therefore we refer to those clicks as costs). To this extent, we take an estimation of the labeling effort during acquisition into account. In addition, we estimate the segmentation quality to identify regions of interest.

As in [21], our candidate regions for acquisition are square-shaped and of fixed size, and we also use a cost metric based on the number of clicks required to draw a polygon overlay for an object. Hence, [21] is in spirit closest to our work. However, instead of using an uncertainty measure to identify high informational regions, we use information about the estimated segmentation quality in terms of the segment-wise Intersection over Union (IoU) (see [28, 15]). Entropy is a common measure to quantify uncertainty. However, in semantic segmentation we observe increased uncertainty on segment boundaries, while uncertainty in the interior is often low. This is in line with the observation that, neural network in general provide overconfident predictions [11, 14]. We solve this problem by evaluating the segmentation quality of whole predicted segments.

Furthermore, also our cost estimation method differs substantially from the one presented in [21]. While the authors of [21] use another CNN to regress on the number of clicks per candidate region, we infer an estimate of the number of required clicks directly from the prediction on the segmentation network and show in our results, that our measure is indeed strongly correlated with the true number of clicks per candidate region.

3 Region based Active Learning

In this section we first describe a region based AL method, which queries fixed-size and quadratic image regions. Afterwards, we describe our new AL method. This method is subdivided into a 2-step process: first, we predict the segmentation quality by using the

segment-wise meta regression method proposed in [28]. Second, we incorporate a cost estimation of the click number required to label a region, we term this add on.

3.1 Method Description

For the AL method we assume a CNN as semantic segmentation model with pixel-wise softmax probabilities as output. The corresponding segmentation mask, also called *segmentation*, is the pixel-wise application of the $\arg\max$ function to the softmax probabilities. The dataset is given as data pool $\mathcal{P}$. The set of all labeled data is denoted by $\mathcal{L}$ and set of unlabeled data by $\mathcal{U}$. At the beginning we have no labeled data, i.e., $\mathcal{L} = \emptyset$, and we wish to provide labels over a given number of classes $c \in \mathbb{N}, c \geq 2$ for all images.

A generic AL method can be summarized as follows: Initially, a small set of data from $\mathcal{U}$ is labeled and added to $\mathcal{L}$. Then, two steps are executed alternatingly in a loop. Firstly, the model (the CNN) is trained on $\mathcal{L}$. Thereafter, a chosen amount of unlabeled data from $\mathcal{U}$ is queried according to a query strategy, labeled and added to $\mathcal{L}$.

In region based AL, we add newly labeled regions to $\mathcal{L}$ instead of whole images. An image x remains in $\mathcal{U}$ as long as it is not entirely labeled. In order to avoid multiple queries of the same region, a region that is contained in both $\mathcal{U}$ and $\mathcal{L}$ is tagged with a query priority equal to zero. In the remainder of this section, we describe the query function in detail and introduce an appropriate concept of priority in the given context.

Region based Queries. The query strategy is a key ingredient of an AL method. In general, most query function designs strive for maximally leveraging training progress (i.e., achieving high validation accuracy after short time) at reduced labeling costs. In the field of semantic segmentation, it seems unnecessary to label whole images. Thus, we aim at querying regions of images which leads to a region-wise concept of query priority.

In what follows, we only compute *measures of priority* by means of the softmax output of the neural network. To this end, let

$$f : [0,1]^{w \times h \times 3} \to [0,1]^{w \times h \times c} \tag{1}$$

be a function given by a segmentation network providing softmax probabilities for a given input image, where w denotes the image width, h the height and c the number of classes.

A *priority map* can be viewed as another function

$$g : [0,1]^{w \times h \times c} \to [0,1]^{w \times h} \tag{2}$$

that outputs one priority score per image pixel. The output of g can be viewed as a heatmap that indicates priority. A higher score of priority should presumably correlate with the attractiveness of the corresponding ground truth. A typical example for g is the pixel-wise entropy H which for a chosen pixel (i, j) is given by

$$H(y_{i,j,.}) = -\sum_{k=1}^{c} y_{i,j,c} \log(y_{i,j,c}) \tag{3}$$

where $y_{i,j,c} = f(x)_{i,j,c} \in [0,1]$ for a given input $x \in [0,1]^{w \times h \times 3}$. The priority maps that we use in our method are introduced in the subsequent section. Note that, if an image pixel has already been labeled, we overwrite the corresponding pixel value of the priority map by zero.

Our AL method queries regions that are square-shaped (*boxes*) and of fixed width $b \in \mathbb{N}$. A box-wise overall priority score is obtained via aggregation. To this end, we simply choose to sum up the scores. That is, given a box $B \subset [0,1]^{w \times h}$, the aggregated score is given by

$$g_{agg}(y, B) = \sum_{(i,j) \in B} g_{i,j}(y)\,. \tag{4}$$

Given the set $\mathcal{B}$ of all possible boxes of width b in $[0,1]^{w \times h}$, we can define an *aggregated priority map*

$$g_{\mathcal{B}}(y) = \{g_{agg}(y, B) : B \in \mathcal{B}\} \tag{5}$$

which can be viewed as another heatmap resulting from a convolution operation with a constant filter. Given t aggregated priority scores, for the sake of brevity named $h^{(1)}(y, B), \ldots, h^{(t)}(y, B)$, we define a joint priority score by

$$h(y, B) = \prod_{s=1}^{t} h^{(s)}(y, B)\,. \tag{6}$$

Analogously to Equation (5) we introduce a joint priority map $h_{\mathcal{B}}(y)$. However, in what follows we do not distinguish between joint priority maps and singleton (aggregated) priority maps as this follows from the context. Furthermore, we only refer to priority maps while performing calculations on priority score level.

Algorithm. In summary, our AL method proceeds as follows. Initially, a randomly chosen set of m_{init} entire images from $\mathcal{U}$ is labeled and then moved to $\mathcal{L}$. Afterwards, the AL method proceeds as previously described in the introduction of this section. Defining the set of all *candidate boxes* as

$$C = \{(y, B) : y = f(x),\, x \in \mathcal{U},\, B \in \mathcal{B}\}, \tag{7}$$

we query in each iteration a chosen number m_q of *non-overlapping* boxes $Q = \{(y_{i_j}, B_j) : j = 1, \ldots m_q\} \subset C$, with the highest scores $h(y, B)$, i.e.,

$$\begin{aligned} &(y, B) \in Q, (y', B') \notin Q \\ \implies &h(y, B) \geq h(y, B') \text{ or } (B \cap B' \neq \emptyset \text{ and } y = y')\,. \end{aligned} \tag{8}$$

A sketch of the whole AL loop is depicted by Figure 1.

3.2 Joint priority maps based on meta regression and click estimation

It remains to specify the priority maps $h^{(i)}(y, B)$ defined in the previous section. In our method, we have $t = 2$ priority maps. As an estimate of prediction quality, we use *MetaSeg* [28] which provides a quality estimate in $[0,1]$ for each segment predicted by f. This aims at querying ground truth for image regions that presumably have been predicted badly. Mapping predicted qualities back to each pixel of a given segment and thereafter aggregating the values over boxes, we obtain our first priority map $h^{(1)}$.

On the other hand, we wish to label regions that are easy (or cheap) to label. Therefore, we estimate the number of clicks required to annotate a box B. From this, we define another priority map $h^{(2)}$ which contains high values for regions with low estimated numbers of clicks and vice versa (details follow in the upcoming paragraphs). We query boxes according to the product of priorities, i.e.,

$$h(y, B) = h^{(1)}(y, B) \cdot h^{(2)}(y, B) \tag{9}$$

as being done in [21], but with both $h^{(1)}$ and $h^{(2)}$ being different. In what follows, we describe the priority maps $g^{(1)}(y)$ and $g^{(2)}(y)$ more precisely, where the aggregated priority maps $h^{(1)}(y, B)$ and $h^{(2)}(y, B)$ are constructed as in Section 3.1.

Priority via MetaSeg. As priority map $g^{(1)}(y)$ we use *MetaSeg* [28] which estimates the segmentation quality by means of predicting the IoU of each predicted segment with the ground truth.

MetaSeg uses regression models with different types of hand-crafted input metrics. These include pixel-wise dispersion measures like (Shannon) entropy and the difference between the two largest softmax probabilities. These pixel-wise dispersions are aggregated on segment level by computing the mean over each segment. Here, a segment is a connected component of a predicted segmentation mask of a given class.

In addition, for each predicted segment we consider shape-related quantities, i.e., the segment size, the fractality and the surface center of mass coordinates. Furthermore, averaged class probabilities for each predicted segment are presented to the regression model.

Training the regression model of MetaSeg requires segmentation ground truth to compute the IoU for each predicted segment and the corresponding ground truth. Since the prediction changes in every iteration of the AL method, we train the regression model for MetaSeg once in every AL iteration. In order to have ground truth available for training the regression model, we randomly select and label a further initial dataset $\mathcal{M}$ of n_{meta} samples, which will be fixed for the whole AL process. To predict the quality of network predictions via MetaSeg, we perform the following steps after updating the semantic segmentation model:

1. Infer the current CNN's predictions for all images in $\mathcal{M}$,
2. Compute the metrics for each predicted segment (from step 1.),
3. **Train MetaSeg** to predict the IoU by means of the metrics from step 2.
4. Infer the current CNN's predictions for the unlabeled data $\mathcal{U}$,
5. Compute the metrics for each predicted segment that belongs to $\mathcal{U}$ (as in step 2.),
6. **Apply MetaSeg** in inference mode to each predicted segment from $\mathcal{U}$ (from step 4.) and its metrics (from step 5.) to predict the IoU.

For each unlabeled (i.e., not entirely labeled) image, MetaSeg provides a segmentation quality heatmap $q(y)$ by registering the predicted IoU values of the predicted segments for each of their corresponding pixels. An example of the segmentation quality heatmap is given in Figure 3. The corresponding priority map as defined in Equation (2), is obtained via $g^{(1)}(y) = 1 - q(y)$. Hence, regions of $g^{(1)}(y)$ containing relatively high values are considered as being attractive for acquisition. More details on MetaSeg can be found in [28].

Priority via Estimated Number of Clicks. As an additional priority map $g^{(2)}(y)$ we choose an estimate of annotation costs. Multi-class semantic segmentation datasets are generally labeled with a polygon based annotation tool, i.e., the objects are described by a finite number of vertices connected by edges such that the latter form a closed loop. [6, 24]. If the ground truth is given only pixel-wise, an estimate of the number of required clicks can be approximated by applying the Ramer-Douglas-Peucker (RDP) algorithm [26, 9] to the segmentation contours.

To estimate the true number of clicks required for annotation in the AL process, we correlate this number with how many clicks it approximately requires to annotate the predicted segmentation (provided by the current CNN) using the RDP algorithm. The approximation accuracy of the RDP algorithm is controlled by a parameter ϵ. We define a cost map $\kappa(y)$ via $\kappa_{i,j}(y) = 1$ if there is a polygon vertex in pixel (i, j) and $\kappa_{i,j}(y) = 0$ else. Since we are prioritising regions with low estimated costs, the priority map is given by $g^{(2)}(y) = 1 - \kappa(y)$. Following the construction in Section 3.1 yields the aggregated priority map $h^{(2)}(y, B)$. A visual example of the cost estimation is given in Figure 5 (left panel).

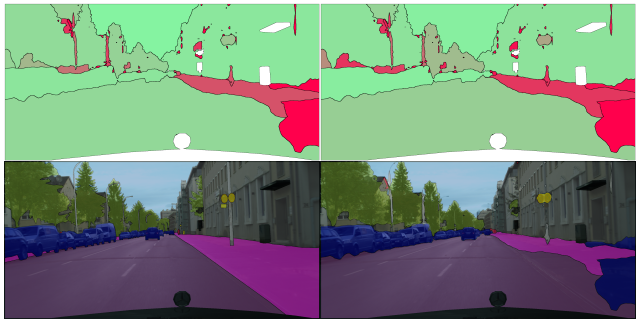

Figure 3. Prediction of the IoU values. The figure consists of ground truth *(bottom left)*, predicted segments *(bottom right)*, true IoU for the predicted segments *(top left)* and predicted IoU for the predicted segments *(top right)*. In the top row, green color corresponds to high IoU values and red color to low ones, for the white regions there is no ground truth available.

In our tests with the RDP algorithm applied to ground truth segmentations, we observed that on average the estimated number of click is fairly close to the true number of clicks provided by the Cityscapes dataset. Therefore, if we assume that over the course of AL iterations, the model performance increases, approaching a level of segmentation quality that is close to ground truth, then the described cost estimation on average will approach the click numbers in the ground truth.

In the following, we distinguish between the following methods: *MetaBox* uses only the priority via MetaSeg and *MetaBox+* uses both the joint priority of MetaSeg and the estimated number of clicks. An overview of the different steps of our AL method is given by Figure 1 and an exemplary visualization of the different stages of MetaBox+ is shown in Figure 4. Note that there are different conventions for counting clicks which we discuss in Section 4.1.

Further priority maps and baseline methods. For the sake of comparison, we also define a priority map based on the pixel-wise entropy as in Equation (3). Analogously to MetaBox and MetaBox+, we introduce *EntropyBox* and *EntropyBox+*: EntropyBox uses only the priority via entropy and EntropyBox+ uses the joint priority of the entropy and the estimated number of clicks. The method EntropyBox+ is similar to the method introduced in [21]. The corresponding authors also use a combination of the entropy and a cost estimation, but the cost estimation is computed by a second DL model. Furthermore, as a naive baseline we consider a random query function that performs queries by means of random priority maps. We term this method *RandomBox*.

4 Experiments

Before presenting results of our experiments , we introduce metrics to measure the annotation effort. To this end, we discuss different types of clicks required for labeling and how they can be taken into account for defining annotation costs. Afterwards we specify the experiments settings and the implementation details. Thereafter, we present numerical experiments where we compare different query

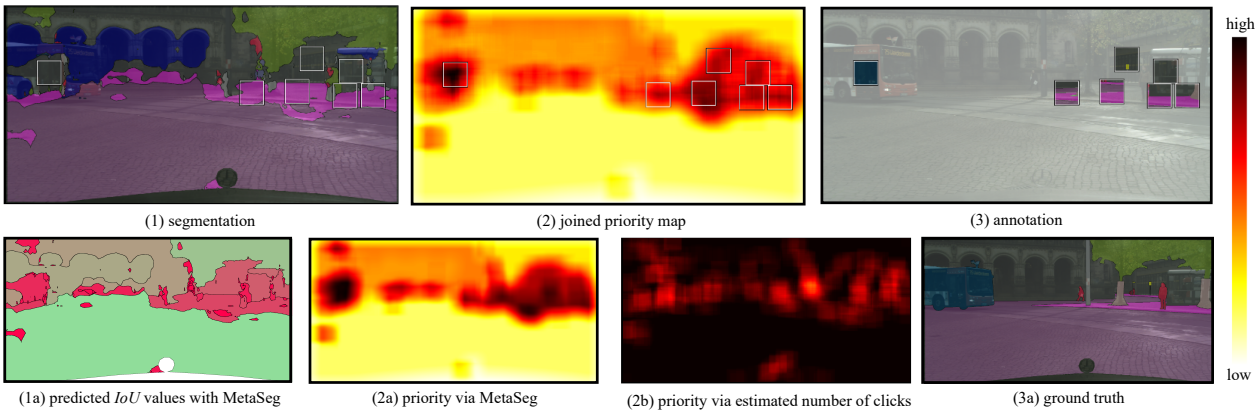

Figure 4. Visualisation of our AL method MetaBox+ at a specific AL iteration. The top row shows the segmentation (1), the joined priority map (2) and the acquired annotation (3). The joined priority map (2) is based on the priority via MetaSeg (2a) and the estimated number of clicks (2b). High values represent prioritized regions for labeling: in (2b) regions with low predicted IoU values are of interest in (2b) regions with low estimated clicks. (1a) shows the predicted IoU values via MetaSeg.

strategies with respect to performance and robustness. Furthermore we study the impact of incorporating annotation cost estimations.

4.1 Measuring Annotation Effort

In semantic segmentation, annotation is usually generated with a polygon based annotation tool. A connected component of a given class is therefore represented by a polygon, i.e., a closed path of edges. This path is constructed by a human labeler clicking at the corresponding vertices. We term these vertices *polygon clicks* $c_p(B) \in \mathbb{N}_0$. Since we query quadratic image regions (boxes), we introduce the following additional types of clicks:

- *intersection clicks*, $c_i(B) \in \mathbb{N}_0$, occur due to the intersection between the contours of a segment and the box boundary,
- *box clicks*, $c_b(B) \in \mathbb{N}_0$, specify the quadratic box itself,
- *class clicks*, $c_c(B) \in \mathbb{N}$, specify the class of the annotated segment.

For an annotated image, the class clicks correspond to the number of segments. Like the polygon clicks, they can also be considered for the cost evaluation of fully annotated images.

For the evaluation of a dataset $\mathcal{P}$ (with fully labeled images), $c_p(\mathcal{P}) \in \mathbb{N}$ is the total number of polygon clicks and $c_c(\mathcal{P}) \in \mathbb{N}$ the total number of class clicks. Let $\mathcal{L}_0$ be the the initially annotated dataset (with fully labeled images) and Q the set of all queried and annotated boxes, then we define the cost metrics

$$cost_A = \frac{c_p(\mathcal{L}_0) + c_c(\mathcal{L}_0) + c_p(Q) + c_i(Q) + c_c(Q)}{c_p(\mathcal{P}) + c_c(\mathcal{P})} \tag{10}$$

$$cost_B = \frac{c_p(\mathcal{L}_0) + c_p(Q) + c_i(Q) + c_b(Q)}{c_p(\mathcal{P})} \tag{11}$$

$$\text{with } c_\sharp(Q) = \sum_{B_j \in Q} c_\sharp(B_j), \quad \sharp \in \{p, i, b, c\}.$$

In addition to that,

$$cost_P \tag{12}$$

defines the costs as amount of labeled pixels with respect to the whole dataset.

The amount of required clicks depends on the annotation tool. The box clicks $c_b(B)$ are not necessarily required: with a suitable tool, the chosen image regions (boxes) are suggested and the annotation process restricted accordingly. Required are the polygon clicks $c_p(B)$ and the intersections clicks $c_i(B)$ to define the segment contours as well as the class clicks $c_c(B)$ to define the class of the annotated segment. Cost metric $cost_A$ (Equation (10)) is based on this consideration.

Cost metric $cost_B$ (Equation (11)) is introduced in [21]. Due to a personal correspondence with the authors we are able to state details that go beyond the description provided in [21]: Cost metric $cost_B$ is mostly in accordance with $cost_A$, except for two changes. The box clicks $c_b(B) = 4$ are taken into account while the class clicks $c_c(B)$ are omitted. Theoretically both metrics can become greater than 1. Firstly, fully labeled images do not require intersection clicks $c_i(B)$. Secondly, ground truth segments that are labeled by more than one box produce multiple class clicks c_c to specify the class. In the following, the cost metrics $cost_A$, $cost_B$, $cost_P$ are given in percent (of the costs for labeling the full dataset without considering regions). An illustration of the click types is shown in Figure 5 (right panel).

4.2 Experiment Settings

For our experiments, we used the Cityscapes [6] dataset. It contains images of urban street scenes with 19 classes for the task of semantic segmentation. Furthermore, the annotation clicks / polygons are given. We used the training set with 2,975 samples as data pool $\mathcal{P}$. For all model and experiment evaluations we used the validation set containing 500 samples. We used two CNN models: *FCN8* [19] (with width multiplier 0.25 introduced in [31]) and Deeplabv3+ [4] with an Xception65 [5] backbone, (short: *Deeplab*). Using all training data, also referred to as full set, we achieve a $mIoU$ of 60.50% on the validation dataset with the FCN8 model and a $mIoU$ of 76.11% for the Deeplab model. We have not resized the images, i.e., we used the original resolution of height $h = 1{,}024$ and width $w = 2{,}048$. In each AL iteration, we train the model from scratch. The training is stopped, if no improvement in term of validation $mIoU$ is achieved over 10 consecutive epochs. Details regarding the training parameters are given in Section 4.3 below. All experiments started from an initial dataset of 50 samples. For experiments with MetaSeg

Figure 5. *(left)*: Visualization of the estimated annotation clicks obtained by the RDP algorithm applied to the segmentation contours of a predicted segment of class "car". The segment contours are highlighted by red color, the obtained vertices (estimated clicks) are highlighted green. *(right)*: Example of possible types of clicks we can take into account for a cost definition. The white (and gray) pixels depict true annotation clicks obtained from the data of (here 5 polygon clicks within the box). The red ones represent additional types of clicks: the ones required for annotating intersection points of segment contour and box edges (here 2 intersection clicks), the ones for defining the box itself (4 box clicks, one for each corner) and one click per segment to specify the class of the segment (here 2 class clicks: for car and street).

based queries (MetaBox, MetaBox+), we took 30 additional samples to train MetaSeg. In each AL iteration we queried 6,400 boxes with a width of $b = 128$, which corresponds to 50 full images in terms of the number of pixels.

Each experiment was repeated three times. For each method we present the mean over the $mIoU$ and the mean over the cost metrics ($cost_A$, $cost_B$, $cost_P$) of each AL iteration. All CNN trainings were performed on NVIDIA Quadro P6000 GPUs. In total, we trained the Deeplab model 180 times, which required approximately 8,000 GPU hours and the FCN8 model 290 times, which required approximately 3,500 GPU hours. This amounts to 11,500 GPU hours in total. On top of that, we consumed a few additional GPU hours for the inference as well as a moderate amount of CPU hours for the query process.

4.3 Implementation Details

To train the CNN models with only parts of the images, we set the labels of the unlabeled regions to ignore. Both models (FCN8 and Deeplab) are initialized with pretrained weights of imagenet [8].

FCN8. For training of the FCN8 model [19], with the width multiplier 0.25 [31], we used the Adam optimizer with learning rate, alpha, and beta set to 0.0001, 0.99, and 0.999, respectively. We used a batch size of 1 and did not use any data augmentation.

Deeplab. For the training of the Deeplab model, with the Xception-backbone [5] we proceeded as in [4]: we set decoder output stride to 4, train crop size to 769×769, atrous rate to 6, 12, 18 and output stride to 16. To consume less GPU memory resources we used a batch size of 4: We have not fine-tuned the batch norm parameters. For the training in the AL iterations, we used as polynomial decay learning rate policy:

$$lr^{(i)} = lr_{base} * \left(\frac{1 - s^{(i)}}{s_{tot}}\right)^p$$

where $lr^{(i)}$ is the learning rate in step $s^{(i)} = i$, $lr_{base} = 0.001$ the base learning rate, $p = 0.8$ the learning power and $s_{tot} = 150{,}000$ the total number of steps. For training with the full set we used the same learning rate policy with a base learning rate $lr_{base} = 0.003$. With these settings we achieve a $mIoU$ of 76.11% (mean of 5 runs). The original model achieves a $mIoU$ of 78.79% (with a batch size of 8).

MetaSeg. We used the implementation of `https://github.com/mrottmann/MetaSeg` with minor modifications in the regression model. Instead of a linear regression model, we used a gradient boosting method with 100 estimators, max depth 4 and learning rate 0.1. In our tests, a gradient boosting method led to better results than a linear regression model. For the training of MetaSeg we used 30 images. We tested MetaSeg for different numbers of predictions and of differently performing CNN models. With the given parameters, we achieve results in terms of R^2 values similar to those presented in the original paper [28].

4.4 Evaluation

Comparison of MetaBox, EntropyBox and RandomBox. First we compare the methods that do not include the cost estimation. As can be seen in Figure 6, MetaBox outperforms EntropyBox in terms annotation required to achieving 95% full set $mIoU$: for the FCN8 model, MetaBox produces click costs of $cost_A = 38.63\%$ while EntropyBox produces costs of $cost_A = 44.60\%$. For analogous experiments with the Deeplab model, MetaBox produces click costs of $cost_A = 14.48\%$ while EntropyBox produces costs of $cost_A = 19.61\%$. Furthermore, for the Deeplab model both methods perform better compared to RandomBox, which requires costs of $cost_A = 22.04\%$. However, for the FCN8 model RandomBox produces the least click costs of $cost_A = 34.54\%$. Beyond the 95% (full set $mIoU$) frontier, all three methods perform very similar on the FCN8 model. For the Deeplab model, RandomBox does not significantly gain performance while MetaBox and EntropyBox achieve the full set $mIoU$ requiring approximately the same costs of $cost_A \approx 36.56\%$.

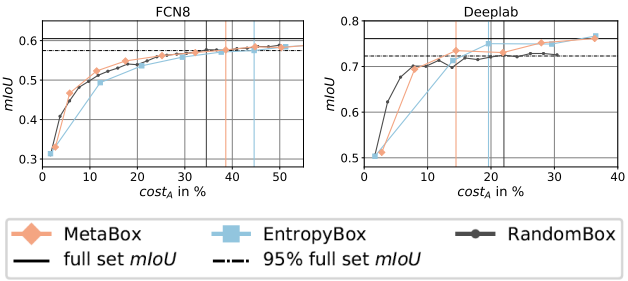

Figure 6. Results of the AL experiments for MetaBox, EntropyBox and RandomBox. Costs are given in terms of cost metric $cost_A$. The vertical lines indicate where a corresponding method achieves 95% full set $mIoU$. Each method's curve represents the mean over 3 runs.

Figure 7 shows a visualisation of prioritised regions for annotation. In general, high entropy values are observed on the boundaries of predicted segments. Therefore EntropyBox queries boxes, which overlap with the contours of predicted segments. Since MetaBox prioritises regions with low predicted IoU values, queried boxes often lie in the interior of predicted segments. Furthermore, EntropyBox produces higher costs in each AL iteration compared to MetaBox and RandomBox. RandomBox produces relatively small but very consistent costs per AL iteration.

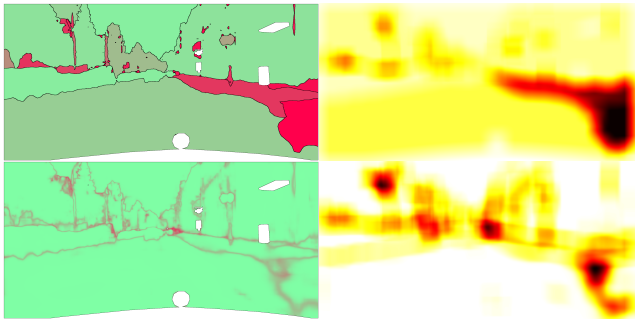

Figure 7. Comparison of query strategies based on MetaSeg and entropy. In the MetaSeg priority map *(top left)*, low estimated IoU values are colored red and high ones green. Accordingly, in the entropy priority map *(bottom left)* low confidence is colored red and high confidence is colored green. The (line-wise) corresponding aggregations are given in the right hand column. The higher the priority, the darker the color.

Comparison of MetaBox+ and EntropyBox+. Incorporating the estimated number of clicks improves both methods MetaBox and EntropyBox, see Figure 8. For the FCN8 model, EntropyBox+ still produces more clicks $cost_A = 40.06\%$ compared to RandomBox. On the other hand, MetaBox+ requires the lowest costs with $cost_A = 32.01\%$. For the Deeplab model, EntropyBox+ and MetaBox+ produce almost the same click costs ($cost_A = 10.25\%$ and $cost_A = 10.47\%$, respectively) for achieving 95% full set $mIoU$. By taking the estimated costs into account, the produced costs per AL iteration are lower for both methods. Although, in comparison with EntropyBox and MetaBox, the methods EntropyBox+ and MetaBox+ require more AL iterations, both methods perform better in terms of required clicks to achieve 95% full set $mIoU$.

In general, we observe that the Deeplab model gains performance quicker than the FCN8 model. This can be attributed to the fact that the FCN8 framework does not incorporate any data augmentation while the Deeplab framework uses state-of-the-art data augmentation and provides a more elaborate network architecture.

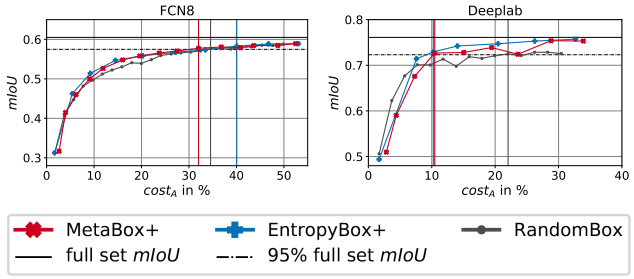

Figure 8. Results of the AL experiments for MetaBox+ and EntropyBox+. Costs are given in terms of cost metric $cost_A$. The vertical lines indicate where a corresponding method achieves 95% full set $mIoU$. Each method's curve represents the mean over 3 runs.

Robustness and Variance Considering Figure 10, where the experiments for each CNN model are shown in one plot, we observe that all methods show a clear dependence on the CNN model. In our experiments with the FCN8 we also observe that results show only insignificant standard deviation over the different trainings. Hence we did not include a figure for this finding and rather focus on discussing the robustness of the methods with respect to the Deeplab model.

For the Deeplab model, the methods show a significant standard deviation over trainings, especially the methods MetaBox, MetaBox+ and RandomBox, see Figure 9. In the first AL iterations, RandomBox rapidly gains performance at low costs. However, in the range of 95% full set $mIoU$ it rather fluctuates and only slightly gains performance. Beyond the 95% full set $mIoU$ frontier, the methods MetaBox(+) and EntropyBox(+) still improve at a descent pace. MetaBox+ and EntropyBox+ nearly achieve the full set $mIoU$ with approximately the same costs.

Furthermore, when investigating the variation of results with respect to two different CNN models, we observe that the discrepancy between the FCN8 and the Deeplab model is roughly 8 percent points smaller for MetaBox+ than for EntropyBox+. This shows that MetaBox+ tends to be more robust with respect to the choice of CNN model.

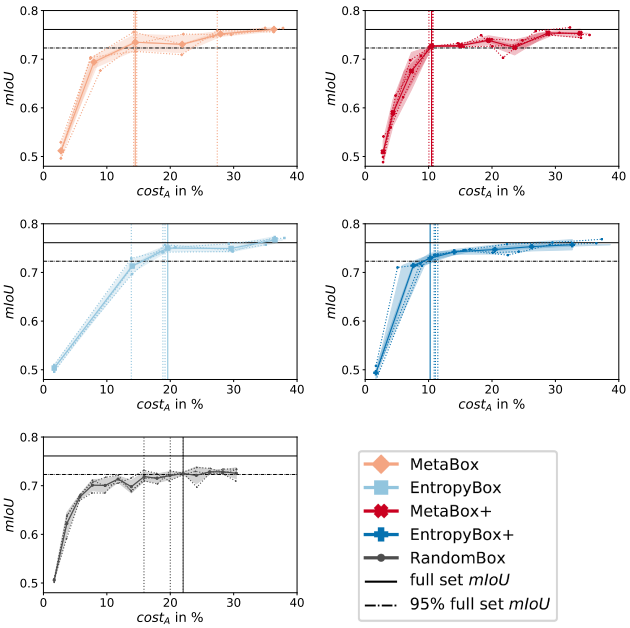

Figure 9. Results of single AL experiments (consisting of 3 runs each) for each AL method with the Deeplab model. Costs are given in terms of cost metric $cost_A$. In each plot, the single runs are given as dotted lines, the mean (over costs and $mIoU$) as a solid line. The vertical lines show where each run achieves 95% full set $mIoU$.

Comparison of cost metrics. In the evaluation above, we only consider the cost metric $cost_A$. A comparison of cost metrics for both CNN models is given in Table 1. Note that EntropyBox+* and MetaBox+* refer to methods that are equipped with the true costs from the Cityscapes dataset. We elaborate further on this aspect in next paragraph. Except for RandomBox, the required costs to achieve 95% full set $mIoU$ is up to 3 percent points lower when considering $cost_B$ instead of $cost_A$. Considering the proportion of labeled pixels $cost_P$ makes the costs seem significantly lower. Noteworthily, for the FCN8 model EntropyBox requires only costs of $cost_P = 10.08\%$ while RandomBox does require costs of $cost_P = 28.57\%$, which is

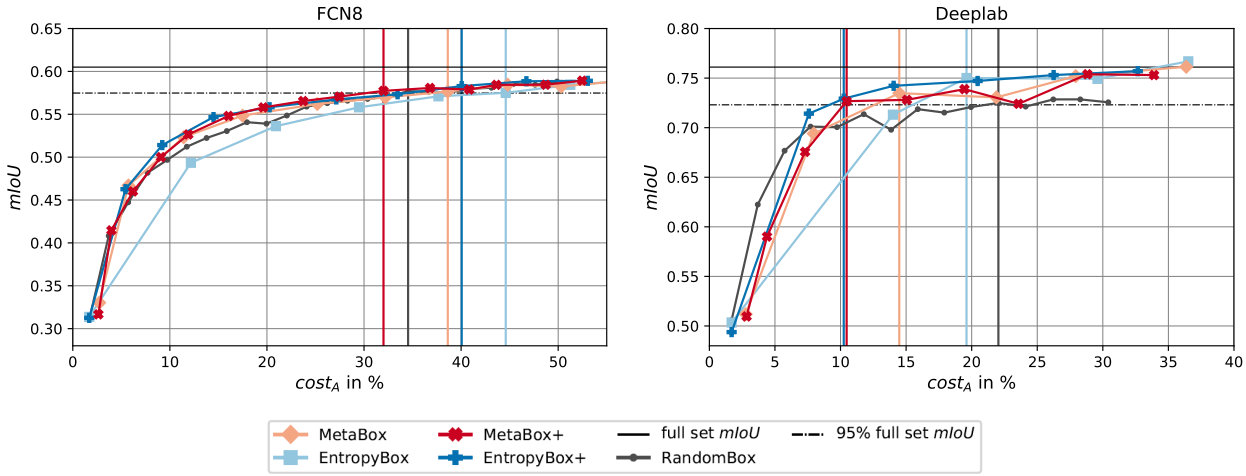

Figure 10. Summary of the results of the AL experiments with the FCN8 model *(top)* and the Deeplab model *(bottom)*. The costs are given in cost metric $cost_A$. The vertical lines display where the 95% full set $mIoU$ are achieved. Each method's curve represents the mean over 3 runs.

CNN model	Cost metric	RandomBox	EntropyBox			MetaBox		
				+	+*		+	+*
FCN8	$cost_A$	34.54	44.60	40.06	*19.82*	38.63	**32.01**	*26.61*
	$cost_B$	35.76	40.74	37.55	*18.87*	35.58	**30.85**	*25.74*
	$cost_P$	28.57	**10.08**	13.45	*10.08*	12.77	17.81	*16.13*
Deeplab	$cost_A$	22.04	19.61	**10.25**	*10.95*	14.48	10.47	*16.21*
	$cost_B$	22.77	17.92	**9.85**	*10.60*	13.56	10.43	*15.85*
	$cost_P$	18.49	5.04	**5.04**	*6.72*	6.05	7.73	*11.09*

Table 1. Annotation costs in % (row) produced by each method (column) to achieve 95% full set $mIoU$. Cost metrics $cost_A$ (Equation (10)) and $cost_B$ (Equation (11)) are based on annotation clicks while cost metric $cost_P$ (Equation (12)) indicates the amount of labeled pixels. The costs with respect to the cost metric of the best performing methods are highlighted. The strategies EntropyBox+* and MetaBox+* represent a hypothetical "optimum" by knowing the true costs. Each value was obtained as the mean over 3 runs.

roughly a factor of 3 higher. Comparing this with $cost_A = 44.60\%$ it becomes clear, that these 10% of the pixels in the dataset constitute to almost half of the actually required click work. This comparison highlights the importance of cost measurement (definition of a cost metric) and that the annotation of image regions requires different human annotation effort.

Click estimation. To evaluate our cost estimation (Section 3.2), we compare it to the provided clicks in the Cityscapes dataset by considering the latter as a "perfect" cost estimation. That is, we supply EntropyBox and MetaBox with the true costs and term these methods EntropyBox+* and MetaBox+*. A comparison of the different click estimations and the true clicks is given in Table 1. For the FCN8 model, the experiments show that knowing the true costs in most cases improves the results: EntropyBox+* produces costs of $cost_A = 19.82\%$. This is the half of the costs of EntropyBox+. MetaBox+* produces costs of $cost_A = 26.61$, which is 6 percent points less costs compared to MetaBox+. For the Deeplab model, using true rather than estimated costs do not lead to better results. However, in terms of cost metric $cost_A$, EntropyBox+* produces 1 percent point more costs then EntropyBox+. MetaBox+* produces even 6 pp. more costs then MetaBox+. Similarly, we see such an increase also with respect to the other cost metrics $cost_B$ and $cost_P$.

5 Conclusion and Outlook

We have introduced a novel AL method MetaBox+, which is based on the estimated segmentation quality, combined with a practical cost estimation. We compared MetaBox(+) to entropy based methods. Using a combination of entropy and our introduced cost estimation shows also remarkable results. Our experiments include in-depth studies for two different CNN models, comparisons of cost metrics, cost / click estimates, three different query types (Random, Entropy, MetaSeg) as well as a study on the robustness. The new methods MetaBox+ proposed by us lead to robust reductions in annotation cost, resulting in requiring 10-30% annotation costs for achieving 95% full set $mIoU$. All our tests were conducted using a query function that minimizes the product of two targets, i.e., minimizing the annotation effort and minimizing the estimated segmentation quality for a given query region. We leave the question open, whether a weighted sum of priorities instead of a product Equation (6) would lead to additional improvements of our methods. Since each method produces different annotation costs per AL iteration, it could be of interest to vary the number of queried boxes (per AL iteration) or to start each experiment with some RandomBox iterations. Furthermore, it would be interesting to also incorporate pseudo labels, i.e., to label regions of high estimated quality with the predictions of the CNN model. Semi-supervised approaches remain a promising direction for further improvements and will be investigated in the future.

REFERENCES

[1] William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler, 'The power of ensembles for active learning in image classification', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9368–9377, (2018).

[2] Arantxa Casanova, Pedro O Pinheiro, Negar Rostamzadeh, and Christopher J Pal, 'Reinforced active learning for image segmentation', *arXiv preprint arXiv:2002.06583*, (2020).

[3] Robin Chan, Matthias Rottmann, Fabian Hüger, Peter Schlicht, and Hanno Gottschalk, 'Application of decision rules for handling class imbalance in semantic segmentation', *CoRR*, **abs/1901.08394**, (2019).

[4] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam, 'Encoder-decoder with atrous separable convolution for semantic image segmentation', *CoRR*, **abs/1802.02611**, (2018).

[5] François Chollet, 'Xception: Deep learning with depthwise separable convolutions', *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1800–1807, (2017).

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele, 'The cityscapes dataset for semantic urban scene understanding', in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2016).

[7] Ido Dagan and Sean P Engelson, 'Committee-based sampling for training probabilistic classifiers', in *Machine Learning Proceedings 1995*, 150–157, Elsevier, (1995).

[8] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, 'Imagenet: A large-scale hierarchical image database', in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, (2009).

[9] David H Douglas and Thomas K Peucker, 'Algorithms for the reduction of the number of points required to represent a digitized line or its caricature', *Cartographica: the international journal for geographic information and geovisualization*, **10**(2), 112–122, (1973).

[10] Yarin Gal, Riashat Islam, and Zoubin Ghahramani, 'Deep bayesian active learning with image data', *CoRR*, **abs/1703.02910**, (2017).

[11] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, 'Explaining and harnessing adversarial examples', *arXiv preprint*, (2014).

[12] Marc Gorriz, Axel Carlier, Emmanuel Faure, and Xavier Giro-i Nieto, 'Cost-effective active learning for melanoma segmentation', *arXiv preprint arXiv:1711.09168*, (2017).

[13] Lukas Hahn, Lutz Roese-Koerner, Peet Cremer, Urs Zimmermann, Ori Maoz, and Anton Kummert, 'On the robustness of active learning', *EPiC Series in Computing*, **65**, 152–162, (2019).

[14] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf, 'Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 41–50, (2019).

[15] Paul Jaccard, 'The distribution of the flora in the alpine zone', *New Phytologist*, **11**(2), 37–50, (February 1912).

[16] S. D. Jain and K. Grauman, 'Active image segmentation propagation', in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2864–2873, (June 2016).

[17] T. Kasarla, G. Nagendar, G. M. Hegde, V. Balasubramanian, and C. V. Jawahar, 'Region-based active learning for efficient labeling in semantic segmentation', in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1109–1117, (Jan 2019).

[18] Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua, 'Introducing geometry in active learning for image segmentation', in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2974–2982, (2015).

[19] Jonathan Long, Evan Shelhamer, and Trevor Darrell, 'Fully convolutional networks for semantic segmentation', in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (June 2015).

[20] Kira Maag, Matthias Rottmann, and Hanno Gottschalk, 'Time-dynamic estimates of the reliability of deep semantic segmentation networks', *CoRR*, **abs/1911.05075**, (2019).

[21] Radek Mackowiak, Philip Lenz, Omair Ghori, Ferran Diego, Oliver Lange, and Carsten Rother, 'CEREALS - cost-effective region-based active learning for semantic segmentation', *CoRR*, **abs/1810.09726**, (2018).

[22] Dwarikanath Mahapatra, Behzad Bozorgtabar, Jean-Philippe Thiran, and Mauricio Reyes, 'Efficient active learning for image classification and segmentation using a sample selection and conditional generative adversarial network', *CoRR*, **abs/1806.05473**, (2018).

[23] Agata Mosinska, Jakub Tarnawski, and Pascal Fua, 'Active learning and proofreading for delineation of curvilinear structures', in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 165–173. Springer, (2017).

[24] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder, 'The mapillary vistas dataset for semantic understanding of street scenes', in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4990–4999, (2017).

[25] Firat Özdemir, Zixuan Peng, Christine Tanner, Philipp Fürnstahl, and Orcun Goksel, 'Active learning for segmentation by optimizing content information for maximal entropy', *CoRR*, **abs/1807.06962**, (2018).

[26] Urs Ramer, 'An iterative procedure for the polygonal approximation of plane curves', *Computer graphics and image processing*, **1**(3), 244–256, (1972).

[27] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, 'U-net: Convolutional networks for biomedical image segmentation', in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, (2015).

[28] Matthias Rottmann, Pascal Colling, Thomas-Paul Hack, Fabian Hüger, Peter Schlicht, and Hanno Gottschalk, 'Prediction error meta classification in semantic segmentation: Detection via aggregated dispersion measures of softmax probabilities', *CoRR*, **abs/1811.00648**, (2018).

[29] Matthias Rottmann, Karsten Kahl, and Hanno Gottschalk, 'Deep bayesian active semi-supervised learning', in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 158–164. IEEE, (2018).

[30] Matthias Rottmann and Marius Schubert, 'Uncertainty measures and prediction quality rating for the semantic segmentation of nested multi resolution street scene images', *CoRR*, **abs/1904.04516**, (2019).

[31] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen, 'Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation', *CoRR*, **abs/1801.04381**, (2018).

[32] Burr Settles, 'Active learning literature survey', Computer Sciences Technical Report 1648, University of Wisconsin–Madison, (2009).

[33] Claude Elwood Shannon, *A mathematical theory of communication*, volume 5, 3–55, ACM New York, NY, USA, 2001.

[34] Yawar Siddiqui, Julien Valentin, and Matthias Nießner, 'Viewal: Active learning with viewpoint entropy for semantic segmentation', *arXiv preprint arXiv:1911.11789*, (2019).

[35] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta, 'Revisiting unreasonable effectiveness of data in deep learning era', in *Proceedings of the IEEE international conference on computer vision*, pp. 843–852, (2017).

[36] Alexander Vezhnevets, Joachim M Buhmann, and Vittorio Ferrari, 'Active learning for semantic segmentation with expected change', in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3162–3169. IEEE, (2012).

[37] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao, 'Deep high-resolution representation learning for visual recognition', *TPAMI*, (2019).

[38] Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin, 'Cost-effective active learning for deep image classification', *IEEE Transactions on Circuits and Systems for Video Technology*, **27**(12), 2591–2600, (2016).

[39] Lin Yang, Yizhe Zhang, Jianxu Chen, Siyuan Zhang, and Danny Z Chen, 'Suggestive annotation: A deep active learning framework for biomedical image segmentation', 399–407, (2017).

[40] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia, 'Pyramid scene parsing network', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2881–2890, (2017).