IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 25, 2021, accepted December 14, 2021, date of publication December 20, 2021,
date of current version December 29, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3136647

A Reinforced Active Learning Algorithm for Semantic Segmentation in Complex Imaging

USMAN AHMAD USMANI[1], JUNZO WATADA[2], JAFREEZAL JAAFAR[1], (Senior Member, IEEE), IZZATDIN ABDUL AZIZ[1], AND ARUNAVA ROY[3]

[1]Department of Computer and Information Science, Faculty of Science and IT, Universiti Teknologi PETRONAS (UTP), Seri Iskandar, Perak 32610, Malaysia
[2]Production and Systems, Graduate School of Information, Waseda University, Kitakyushu 808-0135, Japan
[3]Department of Computer Science, School of Information Technology, Monash University Malaysia, Subang Jaya, Selangor 47500, Malaysia

Corresponding author: Usman Ahmad Usmani (usman_19001067@utp.edu.my)

This work was supported by Yayasan Universiti Teknologi Petronas Prestigious Scholarship (YUTP) under Universiti Teknologi Petronas with cost centre 015LC0-281.

ABSTRACT Semantic segmentation annotation helps train computer vision based Artificial Intelligence models where each image pixel is assigned to a specific object class. The model developers try to identify the features helpful for determining the objects of interest by using various supervised deep learning techniques. However, this is a difficult task due to the complexity of object structures. Two difficulties arise in the current approaches for semantic segmentation. The pixel-wise label approach is costly to obtain and is time consuming. Second, the datasets taken for the semantic segmentation task are not balanced since certain classes are present more than the others. This biases the model performance to the most represented ones. We propose a new reinforced active learning strategy based on a deep reinforcement learning algorithm. This work presents a modified Deep Q Learning formulation for active learning. An agent learns the strategy of selecting a subset of small image regions, which are more knowledgeable than the whole set of images from an unlabeled data pool. The decision on the area of selection is dependent on the assumptions and segmentation model uncertainties taken for training purposes. We use the CamVid and RGB indoor test scenes dataset to evaluate the proof of concept. Our results infer that our approach demands more labels from under-represented groups than the baselines, thus enhancing their efficiency and mitigating the class imbalance. Our method's performance is superior to the conventional deep learning models in detecting 8 out of 11 classes on the Camvid road segmentation scene dataset. It achieves an accuracy of 90.56%, a mIoU score of 87.17%, and a BF score of 93.14%. On the SUNRGB indoor scenes dataset, it gives an accuracy of around 75.82% and a BF score of 77.25%, thus outperforming the current state-of-the-art methods.

INDEX TERMS Reinforcement learning, deep query networks, active learning, semantic segmentation.

I. INTRODUCTION

Semantic image segmentation is classifying and mapping the natural world for several critical applications such as robotic navigation, localization, autonomous driving, and scene understanding. The popular Machine Learning (ML) algorithms have rapidly outperformed the conventional methods based on low-level visual inputs. Voice recognition, handwriting recognition, classifying whole images, and object recognition in images have all lately seen a lot of success [1], [2]. The use of semantic pixel-by-pixel wise labeling is becoming increasingly popular [3]–[5]. Recent techniques have adopted the deep neural network architectures depiction for the pixel-wise labeling to category prediction [6]. The obtained results are better but seem to be preliminary [7]. This is mainly because the maximum sub-sampling and pooling reduce the feature map resolution.

Many publicly accessible scene annotation datasets are available to help inspire new methods for semantic segmentation. It is a general inference that building object categorical models with many training images is more effective. In recent years, many image datasets on a large scale are made available for training the models. Our method is inspired by the applications which need the capability to model the look (building, road), spatial-relationship (context), shape (people, cars) of distinct classes. Because the majority of pixels in the typical road images are associated with the larger classes like building and road, the network must produce smooth

The associate editor coordinating the review of this manuscript and approving it for publication was Li Zhang.

segmentation results. Despite their small size, the engine must be able to delineate the objects based on their shape. Thus, the boundary information must be preserved in the final image representation. Our method is evaluated using two standard scene segmentation datasets: CamVid [8] and the SUN RGB-D interior scene segmentation dataset [9]. The benchmark challenge for segmentation has long been Pascal VOC12 [10]. A significant portion of this dataset consists of one or two classes in the foreground set against a vibrant background.

Active learning (AL) solves the issue of adaptively and intelligently annotating a data portion. AL often uses informativeness measures to locate the unlabeled data objects where the labels primarily benefit the trained model's performance. If the data is labeled randomly with much fewer annotations, an acceptable result can be achieved. The traditional AL techniques are handmade, based on the researcher's intuition and expertise, or by simulating conceptual requirements [11]. They're often customized to specific objectives, and the experimental studies reflect that there are no methods that consistently outperform others across all the datasets [12], [13]. They also make up a tiny portion of the overall number of strategies possible. Recently, it has been proposed to create data-driven solutions based on past AL experience [14], [15]. By considering the trained ML model state for the annotation of data, we can go beyond human intuition and discover new potential methods as a whole.

We solve these problems by effectively deciding which parts of the images should be labeled next. We use AL, which is the process of finding the majority of valuable examples for labeling. The learning algorithm performs well with a few amounts of data than a non-selective method, which labels the total data collection. Although the acquisition label for the semantic segmentation task is highly challenging and consumes more time than classifying the image, there are a few literature in the domain of AL for semantic segmentation. There are two types of AL methods: methods that combine several AL strategies designed manually [16], [17] and AL based data driven approaches [18], [19].

We perform the AL by modeling the process of annotation as a Markov Decision Process (MDP), creating universal action and state spaces, and formulating a new reward function. This function helps in properly reflecting the AL objective of lowering the annotation costs. The conventional approaches label just one region at a time during the semantic segmentation process. Because every step updates the segmentation network and computes the rewards, thus making the task inefficient. In this work, we show how to train an AL model for semantic segmentation using Reinforcement Learning (RL) by maximization of the performance metric, mean Intersection over Union (mIoU) [20], [21].

Two difficulties arise in the current approaches for semantic segmentation. The pixel-wise label approach is costly to obtain and is time-consuming. Second, the datasets taken for the semantic segmentation task are not balanced since certain classes are present more than the others. This biases the model performance to the most represented ones. For training the recently supervised ML methods, we need many annotated datasets, which have proved prohibitively costly. Some categories of objects can appear more frequently than others, leading to undesired performance attributes and biases for the learned models. Although the earlier research on the class imbalance in the segmentation datasets have been done, these studies focus on issues that arise during the data collecting phase. This is critical when a new dataset is created by gathering the annotated data with an oracle in the framework or add the data annotated to a pre-existing dataset. To overcome these drawbacks, our data-driven technique selects and requests labels for the most relevant regions from an unlabeled image collection. This helps the segmentation network in producing enhanced results with only a small amount of annotated pixels. The algorithm can extract the essential parts of the images by choosing regions rather than full images. We further infer that our proposed method helps in solving the problem of the data annotation process. Figure 1 shows the segmentation results of our reinforced AL algorithm. As seen from the segmentation results our segmentation masks are quite clear.

To the best of our knowledge, all AL techniques for semantic segmentation depend on hand-crafted AL heuristics. Learning the AL label strategy over the dataset allows the query agent to request the labeled data based on the characteristics and class imbalances across datasets. Since this work optimizes the mIoU taken per class, it learns to request more under-represented labels from class regions compared with the baselines. Furthermore, our AL framework uses batch-mode Deep Query Networks (DQN), which selects batches of regions efficiently for labeling in each iteration, thus optimizing our method. Our significant contributions are summarized as below:

1) We propose a reinforced-based AL technique to perform semantic segmentation on complex imaging datasets. The AL technique is proposed as an MDP. An agent learns the strategy of selecting a subset of small image regions, which are more knowledgeable than the whole set of images from an unlabeled data pool.
2) During each iteration of AL, our proposed architecture is based on a batch-mode DQN and tags in parallel several regions. Our approach works well for largescale datasets and is consistent with the traditional mini-batch gradient descent.
3) Finally, we use two scene segmentation tasks to evaluate the efficacy of our model: SUN RGB-D indoor scene segmentation [9] and CamVid road scene segmentation [8]. The qualitative and quantitative results infer that our work performs better than the current state-of-the-art techniques using entropy-based selection parameters and uniform sampling baselines.

The rest of the paper is organized as follows. Section III explains our semantic segmentation framework. In this section, we formulate the AL as an MDP. We define the states,

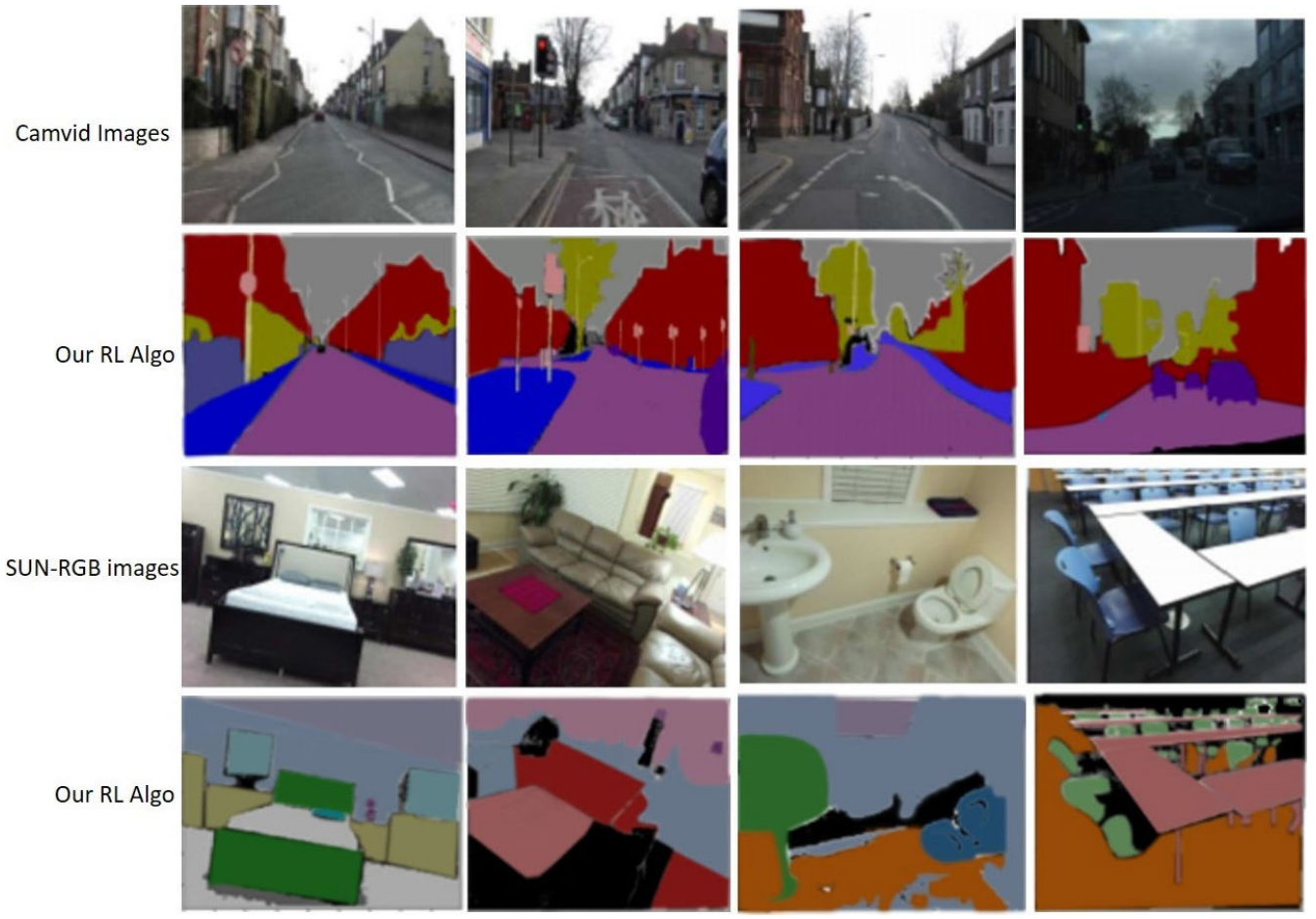

FIGURE 1. Segmentation Results of our reinforced AL algorithm. As seen from the segmentation results on the CamVid Data Scenes and the SUN-RGB images in the first and third row, the results in the second and fourth row are very clear and are much similar to the ground truth. This infers we can segment the objects very accurately.

actions, and rewards that help reflect the AL objective to minimize the number of annotations, ensuring that transferability and flexibility are provided. The benchmarking is then given in detail in Section IV, where we discuss the results on two standard datasets. We finally conclude in Section V.

II. LITERATURE REVIEW

Thanks to the available, challenging datasets, pixel-wise semantic segmentation has attracted attention [22], [23]. At the beginning of deep neural networks, the most effective conventional methods depended largely on hand-engineered features that categorized pixels one by one. The researchers have recently resorted to data-driven AL methods, in which the AL strategies are trained using the annotated data [24], [25]. Given the present state of the trained ML model, they learn which kind of datapoints are most helpful for training the model. Despite several constraints, this has proven to be successful due to the use of previous experience for developing a more successful selection method. The AL method is often designed to learn exclusively from the domains and related datasets appropriate for one-shot learning or transfer [26]–[30]. Second, many have restricted applicability since they depend on specific ML model characteristics, such as conventional classifiers [31] or few-shot learning models [32]–[34]. Finally, when some techniques, such as supervised [35] or imitation learning [36], are employed, the resultant strategy is greedy, leading to poor data selection.

The MDP is used in the AL techniques driven by data for pool-based AL (where datapoints are chosen from a large pool of unlabeled data) and stream-based AL (where the datapoints are taken from a small labeled data pool). The stream datapoints that enter are decided by AL, whether or not to perform the datapoint annotation (as it arrives). In stream-based AL, the annotation process is a discrete action, and Q-learning [37] is the preferred RL technique [20], [21]. Pool-based AL, on the other hand, is concerned with all the possible annotated datapoints, and it is characterized naturally by using continuous vectors, making it impossible for Q-learning. As a result, the policy gradient [38] techniques are often used. We concentrate on pool-based AL in this research. However, we make use of the advantages of Q- learning. This includes the improved data complexity and reduced variance due to bootstrapping. For performing this,

we keep in mind, that although operations are continuous in a pool-based AL, their amount is limited. As a result, we use tailor Q-learning to meet our specific needs.

The traditional AL techniques rely on hand-crafted heuristics derived from sample uncertainty to estimate sample informativeness: entropy, query-by-committee, maximizing the error reduction, expert disagreement, or Bayesian methods, which are needed in the estimation of the posterior distribution. Several techniques are used in many ways to improve the AL performance. A bandit formulation is based on the exploration-exploitation trade-offs, as in RL. These methods are limited since they rely on hand-crafted tactics rather than learning new ones. The current techniques of AL are dependent on an acquisition-based function that uses a learned measure to assess the sample informativeness.

Konyushkova *et al.* [31] labeled a particular sample by computing the error reduction and choosing the samples with the lowest error minimization. Fan *et al.* [37] proposed a low-cost method that employed the predictions having the confidence as pseudo ground truth labels. RL is a method for learning a labeling approach that increases the effectiveness of the training algorithm and has lately gained favor. Peris and Casacuberta [39], Bachman *et al.* [18] and Padmakumar *et al.* [40], for example, utilized knowledge expertise from Oracle- based policies for training a labeled policy, while Pang *et al.* [41], [42] learned the acquisition function using the policy gradient methods. The other methodologies, in one huge step, gathered all of the labeled data. Casonova *et al.* [43] used a bidirectional RNN to pick all the samples in a single step to achieve one-shot learning. Some of them recommended selecting representative sample batches for maximizing the whole unlabeled set coverage.

On the other hand, when the total number of grown classes reaches a threshold, the limited core-set loss performs poorly. Previous research suggests using a Deep Q- Network (DQN) formulation to train the acquisition function, which is more similar to our approach. These studies looked at both stream-based AL, in which the unlabeled samples are given one at a time, and the decision to label or not label them is made later. In the pool-based AL, all the unlabeled data is provided ahead of time, and the decision to use which samples are made later. Our method makes use of the Q-learning benefits to handle the AL pool-based problems. The scope of the problem requires a radical shift in how we think about states, actions, and rewards. For making the problem computationally feasible, we also need to modify the DQN formulation. The technique of AL for semantic segmentation has been given less attention than other methods due to its large-scale nature.

Handmade algorithms have also been employed to enhance the representatives and diversity of the tagged samples. Some researchers utilize superpixel-based unsupervised segmentation techniques. Others focus on hand-crafted algorithms for biological-image foreground-background segmentation. They focused on low-cost methods that provided acquisition functions designed manually for categorizing image regions with low cost. However, these details are not always available, limiting their use. When the cost of classifying an image is not expected to be the same for all images, Mackowiak *et al.* [44] focused on several cost effective techniques. For handling the segmentation dataset large samples number, they used a region-based approach. In contrast to our method, their labeling is dependent on the manually created heuristics, which limits the representation of the acquisition function. This is the first work that we know that uses an AL-based data-driven RL-based approach for semantic segmentation.

III. FRAMEWORK

For the task of semantic segmentation, we propose a new reinforced AL method. Because of its iterative nature, the AL strategy is excellent for the MDP formulation. An agent is rewarded on the basics of the quality of the pre-trained model with the new label. It executes an action specifying which datapoints to annotate for each stage of an AL method. Section A formulates the AL as an MDP. The AL strategy is converted into an MDP policy, which maps the state to action. For achieving flexibility and seamless transferability, we define the generic states, actions, and rewards in this section. After the formulation of the AL problem as an MDP, we utilize RL to train the strategy. The learning of the AL policy is given in Algorithm 1. We put the annotation method to test for the data taken from a range of labeled, unrelated datasets to ensure that it can be used on new unlabeled datasets. Our approach for searching the optimal policy is based on the DQN method in Section B. To utilize the pool-based AL with DQN, we change it in two ways. We formulate it as an MDP, which stores the actions in vectors rather than discrete numbers, correlating it to specific datapoints. Second, we look at the actions set A_t. A change occurs in the actions set between the t iterations because we consider the annotation only once of a datapoint.

A. FORMULATING ACTIVE LEARNING AS A MARKOV DECISION PROCESS

Consider the following AL problem, in which we do the annotation of a dataset D. The AL technique is evaluated using the D' test dataset. Then we choose a datapoint $x^{(t)} \in D$ for annotating several times. Let f_t be a segmentation network which undergoes training on an annotated labeled dataset L_t after an amount of t iterations. This segmentation network assigns a numerical score $\hat{y}_t(x_i) \in R$ to each datapoint, which is then mapped to a label $y_i \in \{0, 1\}$, f_t: $\hat{y}_t(x_i) \rightarrow \hat{y}_i$. If the expected probability is $\hat{y}_t(x_i) = p(y_i = 0|L_t, x_i)$, the mapping function simply restricts the expected probability to 0.5. Instead of regression, we use $\hat{y}_t(x_i)$ as a mapping function and the predicted label as the identity. The quality of the segmentation network f_t *is* evaluated in AL by calculating its empirical performance on D'.

To begin with, we have four distinct data splits in our setup. To maximize the output with a B area budget, we define a

subset of labeled data named D_L and use the AL strategy to learn a successful acquisition feature. A separate split D_Q is used to test the query network. The reward function is obtained by placing the segmentation network to the test on a single subset D_R. Figure 3 shows the state representation generated using the set $D_S(|D_S| \ll |D_L|)$. We use a set-aside set D_S *to* reflect the state space S [45]. To ensure that all the groups are correctly defined, we just use a limited portion of the train set's results. We assume it is a representative sample of the dataset and that any improvements in the performance measures on the subset D_S *is* noticeable. We choose a limited number of regions clipped from the original dataset images from a wide unlabeled range to optimize the output of the f_t segmentation network, which is parameterized by θ. The selected M regions are labeled from an unlabeled set U_t, to be labeled by an oracle for each t iterations, using the query network π which is parameterized by $\emptyset$. Following that, the added samples to the D_L labeled list is then used for training the f_t segmentation network. The mIoU, a standard semantic segmentation metric, is used for the performance measurements.

The MDP is defined by the transformation sequence $(s_t, a_t, r_{t+1}, s_{t+1})$. For each state $s_t \in$S, the agent selects the actions at $\in$ A which samples to annotate from U_t (the segmentation network function at time-step t). The actions $a_t = \{a_m^t\}_{m=1}^M$ which is made up of sub-actions M, where the labeled and unlabeled sets are a function of the segmentation network. Each sub-action requests that a certain area is allocated. After being trained with the chosen samples, the segmentation network gains an r_{t+1} reward dependent on the mIOU improvement. It is essential to mention that the segmentation network's architecture has no impact on the states and behaviors. We now look for a policy to train the query network π in selecting samples that will increase the efficiency of segmentation process.

To train the segmentation network, we use a deep Q network [46] and samples taken from an experience buffer samples $\in$ for training a query network π. There are total of T steps involved in each episode e. The setting of the segmentation network f is started by setting it to a set of initial weights θ_0 and having no data annotated with initializing $L_0 = \Phi$ and $U_0 = D_L$. The AL strategy is formulated as an episodic MDP. Every AL run starts with a small labeled set, $L_0 \subset$ D, and a large unlabeled set, $U_0 = D \backslash L_0$. The following steps are completed during the iteration t:

1) An f_t segmentation network is trained using L_t.

2) The classifier f_t, L_t, and U_t are used to identify a state s_t.

3) The AL agent selects an action at $a_l \subset A_m$ by following the policy π: $s_l \rightarrow a_l$ that specifies a datapoint $x^{(t)} \in U_t$ for the annotation process. The restricted action space is build with M pools P_m^l with R regions. For every pool region, the computation of its sub-action representation is done as $a_l^{m,n}$. M sub-actions are selected by the query agent and an $\in$-greedy policy is used. Each sub-action a_l^m is calculated by the selection of one region x_m (out of R) for the annotation from a P_m^l pool.

4) We find the label $y(m)$ of $x(m)$ and replace $L_{t+1} = L_t \cup \{x^{(m)}, y^{(m)}\}$, $U_{t+1} = U_t \backslash x^{(m)}$. The regions are labeled by an oracle, with the updated sets.

5) The agent is given a reward based on the empirical performance value l_t. The reward r_{t+1} is received by the agent as the performance difference between f_{t+1} and f_t on the set D_R.

The detailed framework is described in Figure 2. This procedure is continued until the desired s_T condition is met. We reach the terminal state s_T for the target quality objective when $l_t \geq q$, where q is a user-specified value, or $T = |U_0|$. The agent can only see s_t, r_{t+1}, and the possible actions set A_t. while f_t, q and D', all are present in the environment. $R_0 = r_1 + \ldots + r_T - 1$ attempts to enhance the AL run's return by using a method that chooses the datapoints and actions intelligently to annotate. We now define the states, actions, and the rewards that will be associated with the AL objective for reducing the number of annotations while enabling transferability and flexibility.

1) STATES

When the quantity of the unlabeled data is huge, AL is performed. We can keep a subset $V \subset U_0$ at the start of each AL run and replace U_0 with $U_0 \backslash V$ without losing generality. To keep track of where the learning process is, we use the segmentation network score $\hat{y}_t(x_i)$. As a result, the state representation for each x_i in V is given as a vector s_t of sorted values $\hat{y}_t(x_i)$. The state representation intuitively includes information, such as the average prediction score or the classifier's uncertainty. For reducing the memory consumption induced by the pixel-wise estimates, we need a compact representation. In D_S, the samples are patched in groups, and the computation of the compact function vectors is done for each of them. Then two sets of features are concatenated to encode each field: one based on f_t's class predictions and the other based on Shannon entropy's prediction uncertainty. In the first series of features I, the maximum pixels numbers are counted for each subclass (normalized).

This function saves a single patch's segmentation computation while omitting the spatial detail, which is less critical for tiny patches. To calculate the predictor's variance, we use the entropy over the likelihood of expected groups. To create a spatial entropy diagram, we calculate the entropy of each pixel position in each area. We use downsampled function maps to compact this representation by applying the average, min, and max-poolings of the entropy diagram. The second collection of features is obtained by flattening and concatenating these entropy functions. Finally, an ensemble of each region's feature representation in D_S reflects the state s_t. The s_t for each region is determined as seen in Figure 3. Owing to the large-scale existence of semantic segmentation, functionality for each area in the unlabeled array at each point will be prohibitively costly. As a consequence, for every phase t, the estimation of the entire unlabeled collection by M pools sampling of regions unlabeled denoted by P_m^l,where each

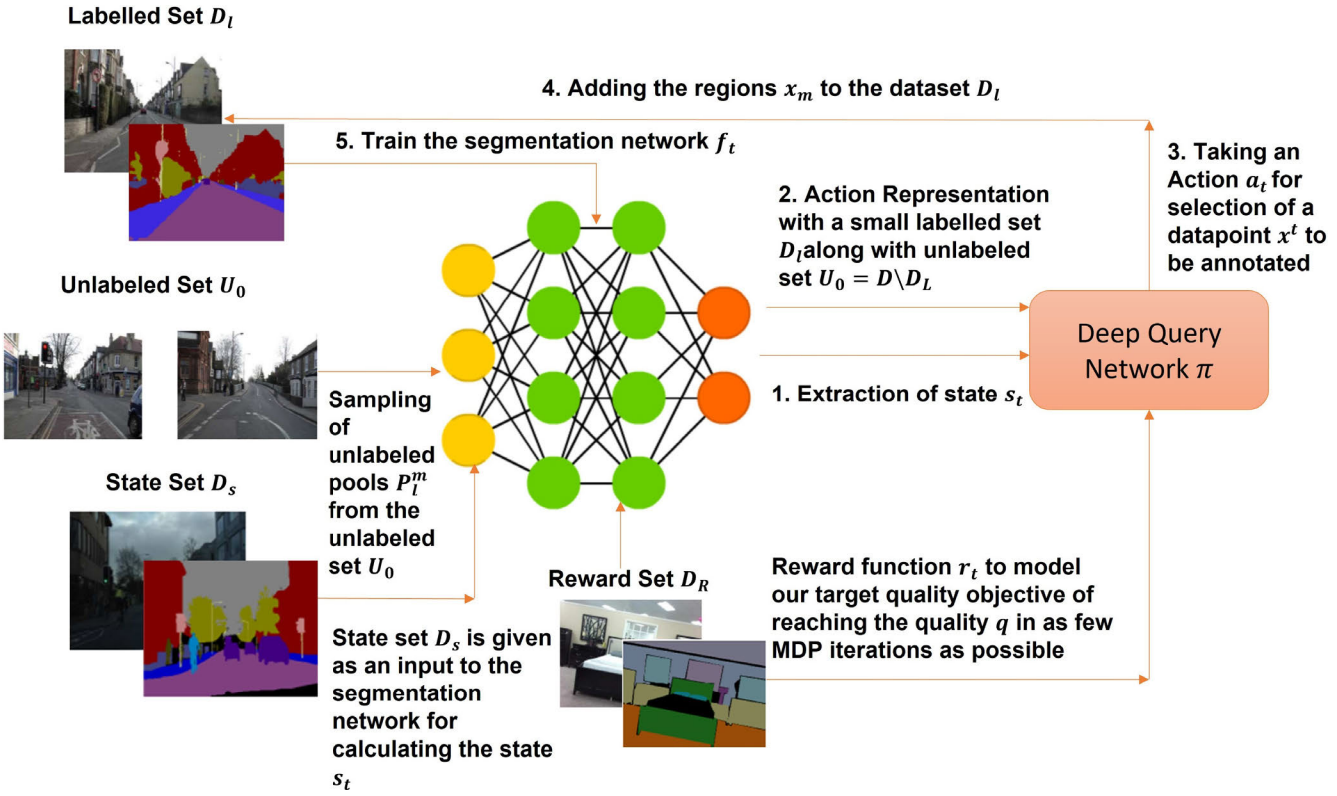

FIGURE 2. Many episodes e with MDP transformations $(s_l, a_l, r_{l+1}, s_{l+1})$ are used to train the query network. 1) The state set D_S and the segmentation network f describe the state s_l. 2) M unlabeled pools P_m^l are randomly selected from the collection U_t unlabeled dataset. The potential sub-actions representation is computed using labeled set L_T, f, and unlabeled set U_t. 3) The action a_l, which is made up of M sub-actions a_m^l, is chosen by the query network. One is selected from a different pool of candidates. 4) Named regions are allocated to specific regions (and omitted from U_t). 5) The most current labeled samples are used for training the f segmentation network. 6) The r_{t+1} reward is calculated using D_R. This procedure is repeated until the labeled regions budget B is met.

of which includes N (uniformly) sampled regions. We now define the representation of actions for the RL process.

2) ACTIONS

We select a datapoint $x^{(m)}$ to be annotated that corresponds to executing an action a_l in our MDP. We demonstrate how a vector a_i can be used to pick a datapoint x_i based on the current segmentation network f_t's score $\hat{y}_t(x_i)$ and the average distances from x_i to L_t and U_t, resulting in the following values of $g(x_i, L_t)$ and $g(x_i, U_t)$ as shown in the Equation 1 below:

$$g(x_i,\ L_t) = \sum\nolimits_{x_j \in L_t} d(x_i, x_j)/|L(t)|,$$
$$g(x_i,\ U_t) = \sum\nolimits_{x_j \in L_t} d(x_i, x_j)//|U(t)| \qquad (1)$$

At t iterations, the choosing of an action a_i from the set A_t where $a_i = [\hat{y}_t(x_i),\ g(x_i;\ L_t);\ g(x_i;\ U_t)]$ and $x_i \subset U_t$. It's worth mentioning that a_t is represented by numbers that aren't limited to datasets or classifiers. In addition to the classification score, two statistics are related to data sparsity and represent the heuristic density estimate. Every sub-action is computed by concatenating four distinct features: class distribution features, entropy (representation of the state), a similarity measure between the labeled set and the area x_k, and another measure of similarity between the region and the unlabeled set. The work of the query network is learning the creation of a more categorized collection (class-balanced) while sampling the unlabeled range. The action representation is shown in Figure 4.

By increasing the segmentation datasets' hard imbalance, the net efficiency can be improved. Jehnsen Shannon divergence score (JSD) is used to measure the similarity between the two probability distributions. The JSD divergence for computing the class distributions from the region x projection map (estimated as projected pixels normalized counts in each category) and the class distributions for every categorized and unlabeled region is determined for each candidate field, x, in a pool (using the network predictions and the ground truth annotations, respectively).

For the labeling set, we measure a JSD between a region x and the class distributions of the labeled regions. To summarize these JSD divergences [47], the histograms from different sections of the same object should appear similar, while histograms from separate objects should look different. It can also be used for the location of the edges. To utilize JSD as an edge detector, two neighbor samples S_1 and S_2 are

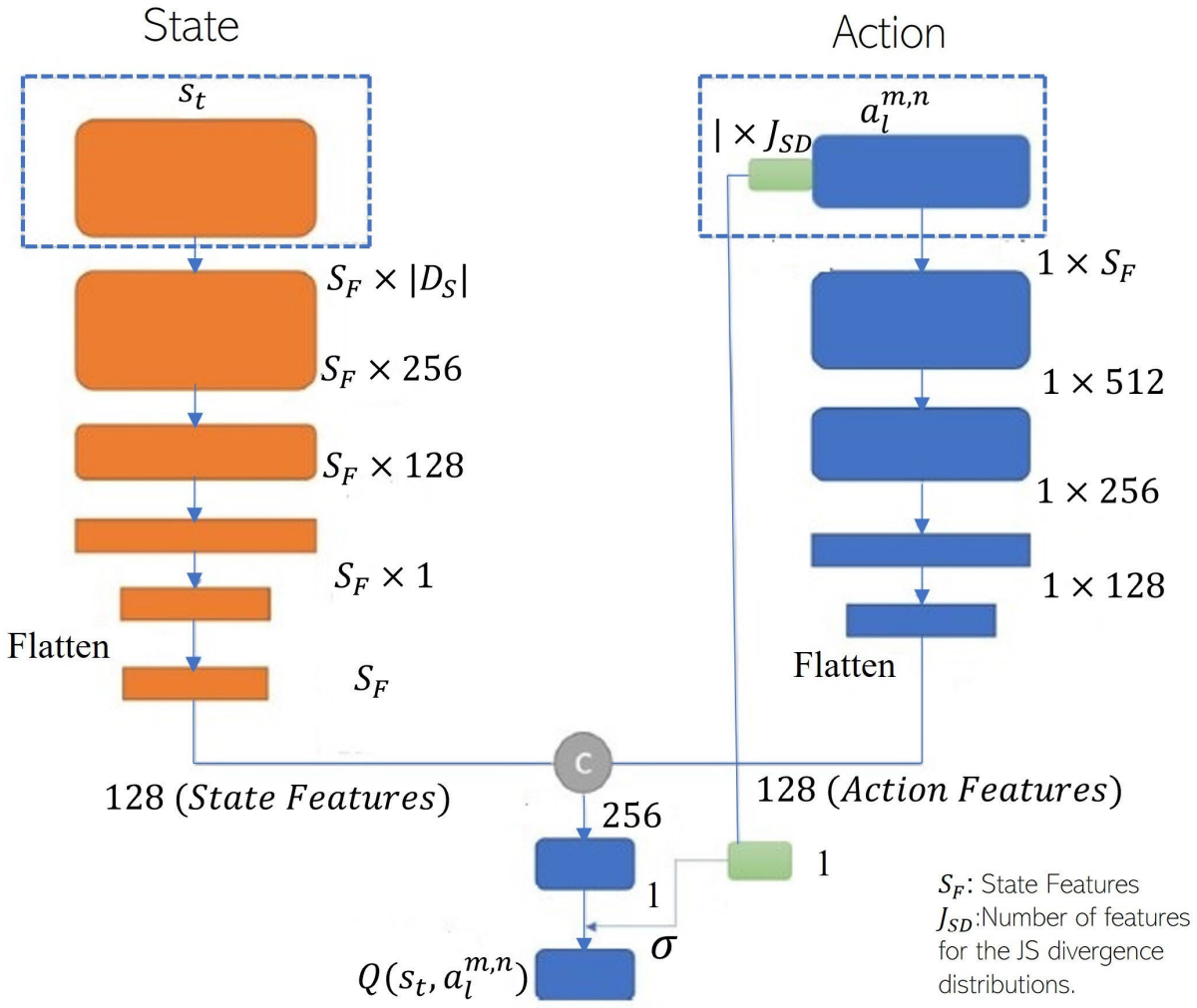

FIGURE 3. DQN includes the action representation a_t and the state representation s_t for the possible action (label region x_m in a pool unlabeled P_m^l). The number of action and state features (entropy-based features and class distributions) is represented by S_F, while the number of JS divergence features is represented by JSD. The computed features are computed using ReLU activation, Batch Normalization and fully connected layers. Both feature vectors are flattened and concatenated before applying a final linear layer that generates a single scalar score. The gated score, controlled by a represented feature generated from the JS divergence distributions of the action representation, is used to measure the *Q*-values.

gathered over the whole image using a double sliding window centered on each pixel. The unweighted JSD divergence of the corresponding normalized histograms P_1 and P_2 shows sample similarity and, most likely, helps identify whether the two samples are from the same object, region, or not present in the image. Consequently, if the difference between S_1 and S_2 is small, the two samples are quite comparable and came from the same region. However, if the divergence is significant, S_1 and S_2 are quite different, and they are most likely from two separate places.

The several alternative orientations of the double window are recorded for each pixel image. This guarantees that edges are properly identified in all directions and avoids bias toward edges along a particular route. For the unlabeled set, the procedure is repeated, resulting in a different JSD divergences distribution. They're mixed and added to the representation of an action. Studying the condition and behavior interpretations directly from a Convolutional Neural Network (CNN) lacks the RL paradigm's functionality.

3) REWARDS

We chose $r_t = -1$ as our reward function to reflect our objective of attaining q quality in as small MDP iterations as possible. As a consequence, when an AL run finishes after T iterations, the R_0 value is $r_1 + \ldots . + r_{T-1} = -T + 1$. In terms of our objective, the best MDP policy corresponds to the best AL method since the lower the number of iterations, the higher the reward. The reward formulation is not a greedy approach since the agent's choices are not restricted if the terminal condition is met after a few iterations. Next, we discuss the policy learning using RL. The learning of an AL strategy determines the optimum (most profitable) MDP policy π^* that converts a state s_t into an action a_t to execute, i.e. $\pi*$: $s_t \rightarrow a_t$. We use the annotated data and the DQN [37]

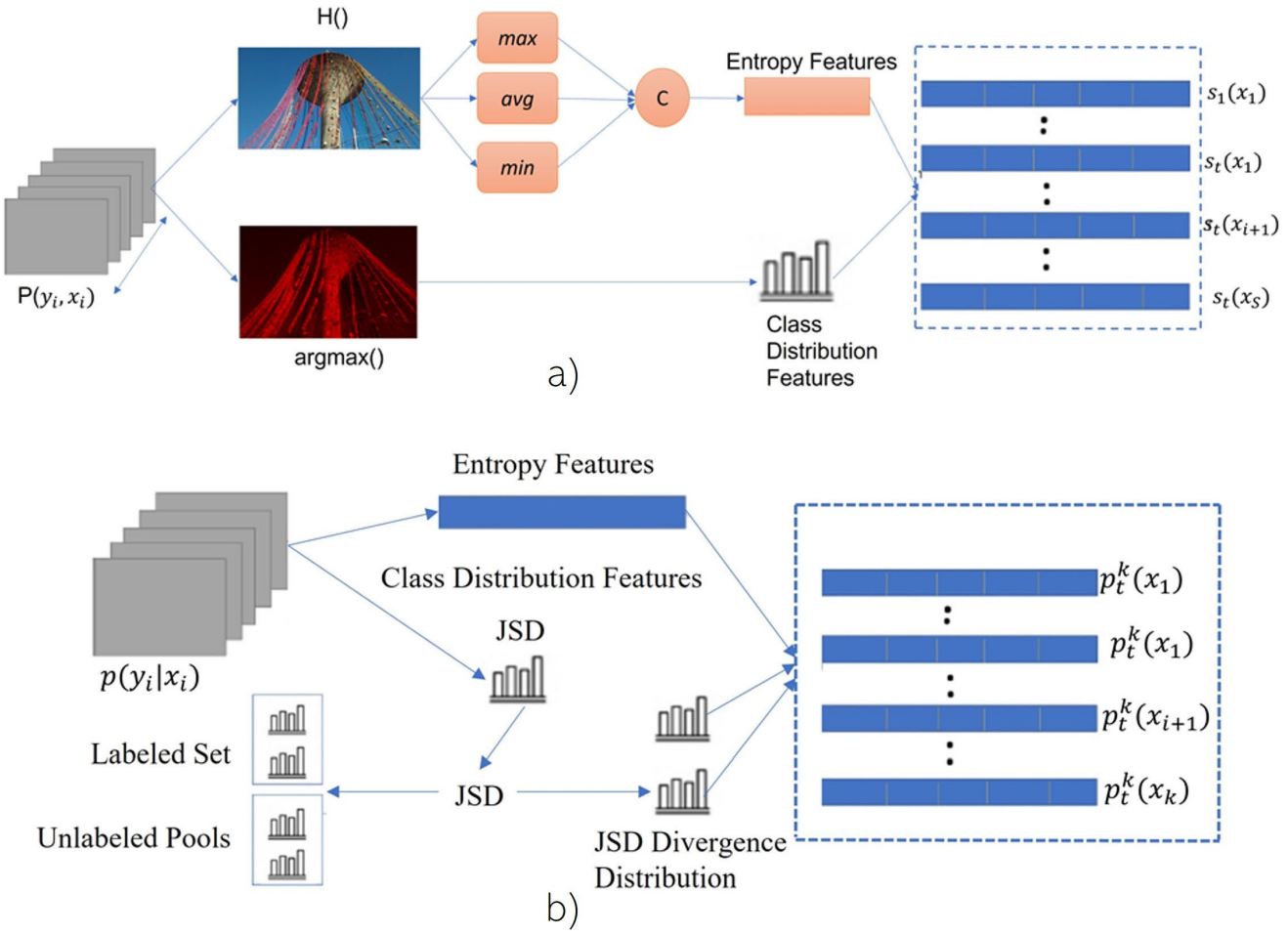

FIGURE 4. (a) Represents the State Representation. Two sets of features are concatenated to encode each region: one based on f_t class predictions and the other based on the prediction uncertainty, represented by the Shannon entropy. (b) Represents the Action Representation. In the Action Representation a sum of four features describes each pool *Pm* region *xm*: class estimates, entropy-based features, and two JS divergence distributions that equate every region *xm* to the unlabeled and the labeled datasets.

method to find the best policy. In our case, $Q(s_t; a_t)$ tries to predict $-(T\text{-}t)$: the number of iterations needed to reach a goal from state s_t after executing action a_t and according to the policy π. To account for the diversity of AL experiences, we use a collection of X annotated datasets $\{X_i\}_{1\leq i\leq X}$ to mimic the AL events. We begin with a random strategy. The following steps are then repeated to accomplish the learning process:

1) The labeled dataset $X \in \{X_i\}$ is picked and divided into two subsets D and D'.

2) π is used to simulate the AL episodes on X. We follow an MDP as described above, where we first hide the labels in D and then follow an MDP. The experience is kept in the form of transitions $(s_t; a_t; r_{t+1}; s_{t+1})$.

3) Based on DQN's update rule experience, we make policy changes π according to the experience. Even though each X has its own set of features, all datasets share the same transition experience $(s_t; a_t; r_{t+1}; s_{t+1})$, allowing for learning a single approach for the whole collection.

B. BATCH MODE DQN

The Q-function takes a state representation s_t as an input and returns multiple values corresponding to the discrete actions in a typical DQN implementation. However, we utilize actions by vectors a_t to characterize the actions, and each one can only be chosen once in each episode since it's pointless to annotate the same point again. We treat actions and states as Q-function inputs, and the standard DQN architecture is adopted to account for this. The Q values of the required actions are then computed on demand for $a_i \in a_t$. It's a good idea to use a feed-forward pass over the network. We utilize the same optimization technique as traditional DQN since our modified architecture is still suitable for Q-learning. We learn the estimates for each action's optimal value, which is defined as the expected sum of future rewards after the activity is completed and the best policy is implemented. In a Q-Learning framework, a DQN is a neural network that approximates a state-value function. Following the execution of some actions and after viewing some sequences s, the policy mapping is given by Q^* given below

Algorithm 1 AL Strategy for Learning the Policy

Input: Data d, Budget B,
Output: π
Episode e, Shuffle s, Labeled Dataset D_l, Learned Model $\emptyset$;
for e 1, 2, ... N **do**,
  $D_l \leftarrow \emptyset$ and sh D
  $\emptyset \leftarrow$Random
  for $i \in \{x \in \{0, 1, 2. . . . . |D|\}$ **do**
    Construction of state s_i by the use of x_i
    The decision is made by the agent according to $a_i = arg\_max\ Q^{\pi}\ (s_i, \text{a})$
    if $a_i = 1$ **then**:
      Obtaining the y_i annotation
      $D_l \leftarrow D_l + (x_i, y_i)$
      Update model $\emptyset$ on D_l
    end if
    Receive a reward r_i out of a held-out set
    if $|D_l| = B$ then
      storing(s_i, a_i, r_w_i, Termination) in N
      Break:
    end if
    Construction of the new state s_{i+1}
    Storing the N transitions (s_i, a_i, r_i, s_{i+1})
    Sampling mini random batches of the (s_j, a_j, r_j, s_{j+1}) transitions from N and performing gradient Descent Step $L(\theta)$
    Updation with θ and policy π
  end for
end for
return the latest policy π

in Equation 2 as follows:

$$Q^*(s, a) = maxE[R_t|s_t = s, a_t = a,], \tag{2}$$

where Q^* is a policy mapping sequence to actions (or distributions over actions). The Bellman equation, a fundamental identity, determines the optimum action-value function. If the optimum value $Q(s_0, a_0)$ of the sequence s_0 at the next timestep is known for all potential actions a_0, the best course of action is to take any action that maximizes the anticipated value of $r + \gamma Q'(s', a')$. The Bellman equation is given below in Equation 3 as follows:

$$Q^*(s, a) = E_{s' \sim \varepsilon}[r + \gamma max_{a'} Q^*(s', a')|s, a] \tag{3}$$

It is used in many RL methods as an iterative update to estimate the action value function. The E stands for Bellman Expectation Equation and is formulated in Equation 4 as follows:

$$v_{\pi}(s) = E_{\pi}[R_{t+1} + \gamma v_{\pi}(s_{t+1})|S_t = s] \tag{4}$$

The value of a state can be decomposed into the immediate reward R_{t+1} plus the value of successor state $v_{\pi}(s_{t+1})$ with a discount factor γ. We are finding the value of a particular state subjected to some policy π. These value iteration methods ultimately lead to the optimum action value function, $Q_i \rightarrow Q$. In reality, since the action-value function is calculated individually for every sequence with no generalization, this fundamental method is completely impractical. A function approximator is often used for estimating the action-value function, $Q(s, a; \theta) \approx Q^*(s, a)$. A linear function approximator is often employed in the RL, although a neural network, a non-linear function approximator, can also be utilized. The Q-network is called a weighted neural network function approximator. By minimizing a collection of loss functions, a Q-network is trained as seen in Equation 5 as follows:

$$y_j = E_{s' \sim \varepsilon}[r + \gamma max_{a'} Q^*(s', a')|s, a] \tag{5}$$

The query agent takes the most effective policy. Each state has behavior associated with it that maximizes the number of expected rewards. For finding the optimal policy, we use a DQN parameterized by θ'.

To train our DQN and computing the rewards, we use a held-out split D_R and a named set D_L. The query agent in this method involves selecting K regions before moving to the next state, as previously mentioned. Each region is assumed to be independently selected for the K annotators, simultaneously labeling one region in parallel. The action a_l is selected and made up of M separate sub-actions $\{a_m^l\}_{m=1}^M$ each with having a restriction of the action space, preventing the action space from combinatorially expanding. To avoid selecting the same region and simplifying the computation multiple times in the same time stage, we limit each sub-action a_m^l to selecting a region x_m in P_m^l specified in Equation 6 as follows:

$$a_l = \underset{a_l}{\text{argmax}} Q(s_t, a_l; \theta') \tag{6}$$

In timestep t, we perform an operation for each $k \in \{1, \ldots, K\}$. A loss function dependent on temporal difference (TD) error is refined to fine-tune the network [48]. The equation above is the goal for iteration i, and $p(s, a)$ is the behavior distribution over actions a and sequences s, where y_j is the objective for iteration I. The loss is described in Equation 7 by the statement over decomposed transformations $T_m = (s_t, a_m^l, r_{t+1}^k, s_{l+1})$ obtained by approximating $r_{l+1}^m \sim r_{l+1}$:

$$L_i\theta_i = E_{s,a \sim p(.)}[(y_i - Q(s_l, a_l; \theta'))^2] \tag{7}$$

where the TD is the goal for each sub-action and ε represents the experience replay buffer. We used target networks having weights φ' and the double DQN [49] formulation for training stabilization. The parameters derived from the θ_{i-1} previous iterations undergo fixes when the loss function $L_i(\theta_i)$ is optimized. The targets depend on the network's weights; this concerns the targets that are taken in use for the supervised learning and are fixed before the beginning of the learning process. The differentiation of the loss function is done concerning the weights, and the following gradient is achieved as

shown in the Equation 8 as follows:

$$\nabla\theta_i L_i(\theta_i) = E_{s,a\sim p;s'\sim\in}[(r + \gamma max_{a'}(Q, s', a'; \theta_{i-1}) - (Q, s; \theta_i))\nabla\theta_i(Q, s; \theta_i)] \quad (8)$$

The query network evaluates the action, which is then chosen by the target network; the evaluation and the action is decoupled. The TD goal for every sub-action is described in Equation 9 as follows:

$$y_l = r_{l+1} + \gamma Q(s_{l+1}, \underset{a_{l+1}^{m,n}\in P_{l+1}^{m}}{argmax}\ Q(s_{l+1}, a_{l+1}^{m,n}; \theta); \theta') \quad (9)$$

where the discount factor is γ. This technique generates a large yet finite MDP, with each sequence indicating a different state. Consequently, we use the whole sequence s_t as the state representation at time t when using conventional RL techniques on MDPs. The agent's objective while engaging with the environment is to choose actions that maximize the future rewards represented in Equation 10 below by R_t.

$$R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_t \quad (10)$$

where the time-step T is when the process ends and is used to calculate discounted future rewards. The DQN algorithm is shown in Algorithm 2.

Algorithm 2 Deep Q Network Policy

Episode e
Initializing a total capacity of N of replay memory D
Initializing the random weights of the action-value function Q
for $e = 1, N$ **do**
Initializing the s_1 sequence $= \{x_1\}$ and pre-processed sequenced $\emptyset_1 = \emptyset(s_1)$
for $x = 1, M$ **do**
The probability $\in$ selecting a random action random a_t otherwise selection of $a_t = max_a Q * (\emptyset(s_t), a; \theta)$
Executing action a_t in emulator and observe reward r_t and image x_{t+1}
Set $s_{t+1} = s_t; a_t; x_t + 1$ and pre-process $\emptyset_{t+1} = \emptyset(s_t + 1)$
Storing transitions $(\emptyset_t; a_t; r_t; \emptyset_t + 1)$ in D
Sampling transitions randomly $(\emptyset_j; a_j; r_j; \emptyset_{j+1})$ from D
Set $\begin{cases} r_j = y_j \text{ for terminal } \emptyset_{j+1} \\ r_j + \gamma max_{a'} Q(\emptyset_{j+1}, a'; \emptyset \text{ for terminal } \emptyset_{j+1} \end{cases}$
Perform a gradient descent step on $(y_j - Q(\emptyset_j; a_j; \emptyset))^2$ according to equation 8
end for
end for

C. ANALYSIS

We use three widely used output metrics to compare the quantitative Accuracy (*Acc*) of our model: Global Acc (G) is the percentage for the classified pixels correctly in the dataset, the mean of the predictive accuracy is the class average Acc (C), and mIoU is the average of the IoU, as described in the Pascal VOC12 challenge across all classes [10]. It is a popular semantic image segmentation estimation metric that calculates the IoU for each semantic class before calculating the average across classes. The Jaccard Index, also known as the mIoU metrics, is the most often used benchmarking metric. Since the mIoU penalizes false positive predictions, this metric criterion is more stable than the average class precision. The mIoU metric is incompatible with the balanced cross-entropy loss class. Another approach in the semantic segmentation task is counting the number of pixels in the image that are correctly identified. The pixel precision for each object class is widely calculated individually.

However, as Csurka and Perronnin [50] pointed out, this metric does not often correlate to the judgments of human qualitative rankings in a high-quality segmentation task. They demonstrate that mIoU prefers area smoothness over boundary accuracy (*Acc*) by examples. As a result, they propose combining the mIoU metric with a boundary measure based on the Berkeley contour matching score, which is often used to determine the precision of unsupervised image segmentation. They [50] extended this to semantic segmentation, demonstrating that using the mIoU parameter for calculating the semantic contour precision agrees well with the human segmentation scores. To calculate the F-Measure of the segmented image, we multiply the Precision (*Pre*) with the Recall (*Rec*) for every class present in the ground truth test frame. It is determined by dividing the number of true positive results by the total number of positive results (including erroneously recognized ones). The number of genuine positive results divided by the total number of samples detected as positive is called a *Rec*. In diagnostic binary classification, *Pre* is also known as a positive predictive value, while *Rec* is also known as sensitivity. We tested our approach on fully labeled datasets, in which the preference is to mask out a portion of the labels and show them while the AL algorithm picks them up.

The CamVid dataset contains 360×480 street scene view images divided into 11 groups. For the train, validation, and test sets, there are 370, 104, and 234 images, respectively. We used uniform sampling for measuring and comparing our baselines acquisition role by dividing the train collection into 120 labeled images (ten for D_S and the rest for D_L) and 250 for D_Q. By limiting the sample of D_S for having a comparable class distribution to that of D_L, the state set is selected to be reflective of D_L. Each image is divided into 20 regions, each of which is 80×90 pixels wide.

For D_R, we use the validation set of the dataset. During the test set, we report the final segmentation findings. We used $K = 24$ regions per step in our experiments. Our model is quite robust on the number of regions picked at each time step. We analyze the effect of asking for labels in regions rather than the complete images and the number of regions requested at each step. The amount of regions chosen at each time point does not affect our model. When we choose the value of K as 1,12,24,36,48 and 72 our validation IoU is around 88%. Thus, we infer that the number of regions added

at each phase has little impact on our selection network. The split D_R is used to evaluate the DQN rewards, and the selection of hyperparameters relies on the right baseline and method configuration. We use uniform sampling for dividing the dataset into training, testing, and validation sets. In uniform sampling, when a sample is selected from a population that has been grouped into strata, a uniform sampling fraction is utilized since the number of units pulled from each stratum is proportionate to the total number of units in that stratum. The more entropy there is, the more consistent the distribution across classes, and our method has the highest entropy.

The five different runs average, and the standard deviation is measured (5 random seeds). To fill out the results, we use 224×224 crops and random horizontal flips. Although, we can apply AL in an unlabeled data environment with an oracle in the loop for the label of selected regions, we choose to test our technique on datasets fully labeled since masking labels and exposing them when the AL algorithm selects them is more accessible. The Cityscapes dataset has a resolution of 2048×1024 pixels in 19 semantic categories and real street scene views. The validation dataset consists of 500 images, while the training set consists of 2975 images with fine-grained segmentation labels. At random, we select 420 annotated images from the train collection. These 20 images are for D_S, 180 for D_L, and 220 for D_R, on which we calculate our rewards. The remaining 2555 train set images are used for D_Q as they would be if they weren't labeled. The results of the validation set are shown (test set not available). Each image is split into 115 regions, each of which is 115×115 in size. We choose $K = 230$ regions per phase.

We evaluate the learned acquisition function and baselines on D_L by requesting labels until the specified budget is met. It should be emphasized that the baselines process the learnable component. After the budget has been fulfilled, we use D_L for training the segmentation network until it converges (with early stopping in D_R). For our method, Camvid uses a 30 size pool. Because Cityscapes had a larger amount of data, we selected pool sizes of 400, 300, and 100. The Pool sizes are calculated using the mIoU. Surprisingly, when training with the newly acquired labels offers no more information, our algorithm works with low budget circumstances. It quickly adapts to the training and produces similar effects to the original weights.

Even though we use class balancing when training the variants, smooth segmentation also necessitates strong global *Pre*. Another hypothesis is that the segmentation is used to demarcate categories such as highways, houses, sidewalks, and skies in autonomous driving. These groups account for the vast majority of pixels in an image, and the accurate segmentation of these important classes is linked to high overall accuracy (*Acc*). We infer from the findings that the class average is at its peak and leads to low global precision, suggesting perceptually noisy segmentation.

Using the same dataset as Segnet [77], we put our segmentation algorithm for evaluation and compared it to other state-of-the-art approaches. CamVid [8] is a dataset that can be used for a variety of purposes. The SUN RGB-D [9] data collection consists of 5285 indoor training sets and 5050 test sets. Multiple types of cameras record the images and hence have varying resolutions. It is our job to preserve the frame's segmentation. It is a strenuous exercise since the objects are in different sizes, heights, and poses. Partial occlusions are very normal, and some of the images in the dataset display them in very odd forms. As a consequence, these characteristics make it one of the most challenging segmentation operations. Another problem is the large number of different sizes depicted in the scene. The results of our research were compared to well-known deep architectures evaluated on the huge SUN RGB-D dataset. The results of our algorithm yield promising results when compared to the various segmentation techniques.

D. TRAINING

The D_L dataset is used to train the query network specified by an amount of budget to make it simpler to choose regions that will improve the efficiency in a data-limited setting ($4k$ regions for Cityscapes, $0.5k$ regions for Camvid). On D_V, we look at the baselines and the learned acquisition function, which requires labels before a budget can reach a specific amount B. It is worth noting that the baselines have no information that can be learned. We exercise the segmentation network f with L_T before convergence once the budget is reached (early stopping in D_R). The segmentation networks of both approaches are pre-trained on the GTA dataset [8], a virtual dataset that can collect large quantities of classified data without human intervention, and D_L on the GTA dataset. (Where markers were used to train the DQN) The final segmentation results are verified using the CamVid test collection [8] and the Cityscapes validation dataset [51] (measured in mIoU). With a ResNet50 backbone [52] and ImageNet [53] pre-training, the segmentation network f is a semantic segmentation adapted to the pyramid network function. Since the network is pre-trained on the entire training large-scale synthetic dataset set, GTAV [54], no human labeling is needed.

The terms used in this dataset are often the same as those used in our work. The query network is divided into two parts: computing the state features and calculating the behavior features, then merged. Each layer includes ReLU activation, batch normalization, and a fully connected layer. In the action representation, the action and the state structure have four and three layers, respectively. The number of operation functions and states (class distributions and entropy-based features) is represented by S_F, while J_{SD} represents the number of JSD divergence distribution features. A final layer performs the fusion of them together to represent the global features, which are sigmoid gated and governed by JSD distance distributions.

At each stage of the AL phase, the weights are adjusted by 16 experience tuples sampling batches from an experience replay buffer (600 for Camvid [8] and 3200 for Cityscapes [52]). Both the networks undergo training using

the stochastic gradient descent (SGD) with momentum. For both the segmentation and query networks, the same learning rate is used: 10^{-3} and 10^{-4} for Camvid [8] and Cityscapes [52] respectively. For Camvid [8], a training batch size of 32 is used, and for Cityscapes [52], a batch training batch size of 16 is used. We put our segmentation network to the test using the CamVid [8] road scenes dataset. There are just 367 trained and 233 test RGB images in this dataset with a resolution of 360×480 (day and dusk scenes). The 11 distinct types include lanes, buildings, motorcycles, people, signage, columns, and sidewalks, to name a few.

The other methods have calculated the precision metrics such as SegNet-Basic, FCN-Basic, FCN BasicNoAddition. These network adjustments are discussed in the SegNet work [77] and use less memory during inference since it just has to store max-pool indices. FCN-Basic, on the other hand, saves encoders with complete tables, which use far more memory (11 times more). Each decoder sheet in the SegNet-Basic [77] has 64 working decoders. FCN-Basic, on the other hand, uses dimensional reduction and has less than 11 character maps per decoder sheet. Therefore, the convolutions number in the decoder network is decreased, allowing the FCN-Basic to run faster during inference (forward pass). The SegNet-Basic [77] decoder network utilizes a particular point of view than the FCN-Basic network to create a more comprehensive network.

For the same amount of iterations, the precision is higher, resulting in better training results than FCN-Basic. Although the memory inference time is small, SegNet-Basic performs better than the FCN-Basic. However, the inference time is more. The SegNet-Basic [77] decoder is better and most comparable to the FCN-Basic-NoAddition decoder. FCN-Basic-NoAddition learns how to make dense feature graphs by either mastering deconvolution directly or perform the sampling process first and then mixing with the qualified decoder filters. SegNet-Basic [77] performs better because of the increased decoder power. FCN-Basic-NoAddition precisions are much lower than FCN-Basic. This emphasizes the importance of understanding the encoder properties of the maps for achieving the best outcomes.

IV. BENCHMARKING

Using the Caffe framework, we test the effectiveness of our AL strategy for semantic segmentation algorithms. The first task involves road segmentation, which is already a potential application to various autonomous driving issues. Second, several augmented reality (AR) technologies are of significant importance to indoor segmentation. The RGB input files for all segmentation tasks are 360×480. We have compared our RL algorithm with the well-known deep segmentation architectures such as SegNet [77], FCN [54], DeepLabLargFOV [55] and DeconvNet [56]. Batch normalization is used to differentiate between the external and internal covariate variations resulting from the distribution of outcomes. Standardizing data points is an alternative, but the standardization of batches means studying how to standardize the data. In DQN, we use the replay information and a wider batch size captured from the replay buffer at any time. Batch normalization is carried out in the convolutional layers. In this way, the training process can see the data uniformity as a whole.

A. ROAD SCENE SEGMENTATION

The various road scene data sets are required to parse the semantic segmentation [3], [4], [5]. We use the CamVid dataset [8] to benchmark our reinforced AL algorithm since it contains all the videos sequences. This allows us to compare our proposed RL model framework with other architectures that use motion and structure [7]–[9] and video segments [10]. Figure 5 compares the qualitative results of our method to those of other deep architectures. The qualitative findings demonstrate that the proposed framework segment the image's small classes, which results in a good segmentation mask. As compared to some of the best-performing methods, our RL model performs well. Our approach is also compared to several non-deep learning methods using this benchmark. Random Forests [57], Boosting [58], and CRF-based methods [59], [60] are just a few examples. This is done so that the users could get an idea of the increasing performance of deep learning methods compared to the several conventional computation techniques. Table 1 compares the quantitative comparisons of our AL [61] algorithm to standard approaches in the CamVid 11 road segmentation scenes [62].

Our AL algorithm has the highest metrics and the predictions are more reliable in 8 out of 11 classes compared to CRF-based approaches (see Table 1). When it comes to defining the boundaries, our method is much more concise and precise. Segnet [77] and FCN DeconvNet [6] with fully connected layers (converted to convolutional layers) train much slower than our method and have a forward-backward pass time greater than our method. Overfitting is not a problem in training these larger models because their metrics, including our AL model [63], [64] demonstrated an increasing pattern at iterations. When the deconvolutional layers are trained rather than the set with bi-linear interpolation weights, the FCN model's output, especially the BF value, improves. It also provides better results in a shorter period.

The results in Table 2 show that using our AL algorithm gives good results. This shows how our architecture can extract essential features from an input image, and then the mapping is done for the named class segments. The pixel *Acc* of our system is 90.56 percent, which means we correctly identified a significant percentage of the pixels in the image. The mIoU score (87.17 percent) is substantial. Our semantic segmentation prediction is computed separately for each class before being summed over all classes to give us a global mIoU score. The BF ranking is higher than the other methods, with a value of 93.14 percent, showing that we have low false positives and false negatives, indicating that we are marking pixels correctly and are not bothered by false negatives. The most intriguing result is that when we train our model with

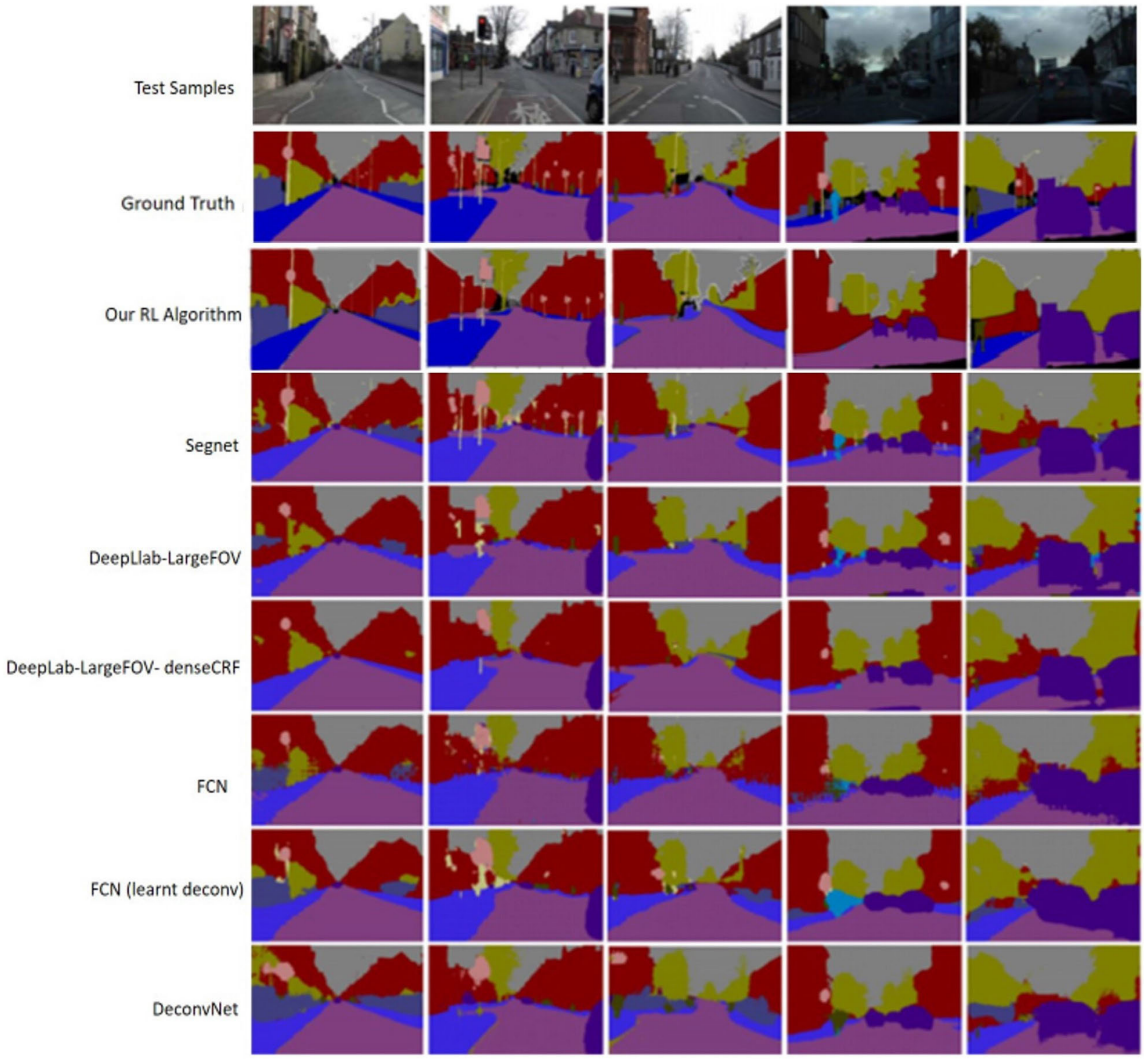

FIGURE 5. The following results are obtained on the CamVid day and dusk test samples. When all models are trained in a controlled environment, our reinforced AL algorithm outperforms some of the larger models, especially in terms of boundary delineation. SegNet [77] is able to delineate regions but it is not very clear. It misses some objects and classes. DeepLab-LargeFOV with CRF post-processing performs poor and misses the smaller groups. Fully Convolutional Networks (FCN) [6] and DeconvNet [7] despite being the largest models and having longer training period, its predictions are inaccurate in small groups.

a robust training dataset obtained by combining [22], we see a significant improvement in mIoU scores and class average metrics. Our model's qualitative and quantitative results (see Fig. 4) outperform those of the other models. It is also capable of effectively segmenting large and small classes. The quantitative results of our method are compared to that of other widely utilized, fully abstract segmentation architectures in Table 3.

As compared to our AL method, SegNet [77] and DeconvNet [56] both have low scores in both tests. DeconvNet is more accurate when it comes to defining boundaries. FCN [6] and DeconvNet [56] train more steadily and have an equal or superior forward-backward pass period than SegNet [77] since they have fully connected layers (which have been converted into convolutional layers). We see that the deconvolutional layers are learning rather than fix them with bi-linear interpolation weights, thus improving the FCN [6] model's efficiency, especially the BF score. Furthermore, it yields better results in a shorter period. DeepLab-LargeFOV [55], which can predict labels at a 45 $\times$ 60 pixels resolution and thus is the smallest model in terms of parameterization and therefore the quickest to understand and produces competitive results. Boundary precision, on the other hand, is lower and is a characteristic shared by all architectures. After a long period of training, DeconvNet's [56] BF score outperforms the other networks. At the time point DeepLab-LargeFOV-final [55] dense CRF, the influence of dense CRF [56] post-processing can be observed. As the global and mIoU averages rise, the class average falls. On the other hand, the BF score has shifted dramatically. The dense CRF hyperparameters are obtained in these methods through a time-consuming procedure based on grid-search on a training array subset

TABLE 1. Quantitative comparisons of our reinforced AL algorithm with state-of-the-art methods on the CamVid 11 road segmentation scenes. Our Method outperforms all other approaches, including those that use video, depth, and/or CRF in most groups. Our predictions are more reliable in 8 out of 11 classes compared to CRF-based approaches.

Method	Building	Tree	Sky	Car	Sign-Symbol	Road	Person	Fence	Column-Pole	Side-Walk	Bicyclist	Class Avg.	Global Avg.	MIoU	BF
SfM+Appearance [65]	46.2	61.9	89.7	68.6	89.5	89.5	53.6	46.6	0.7	60.5	22.5	53.0	69.1	n/a	
Boosting [66]	61.9	67.3	91.1	71.1	92.9	92.9	49.5	37.6	25.8	77.8	24.7	59.8	76.4	n/a	
Dense Depth Maps [57]	85.3	57.3	95.4	69.2	98.5	98.5	23.8	44.3	22.0	38.1	28.7	55.4	82.1	n/a	
Structured Random Forests [60]	n/a											51.4	72.5	n/a	
Neural Decision Forests [67]	n/a											56.1	82.1	n/a	
Local Label Descriptors [68]	80.7	61.5	88.9	16.4	98.0	98.0	1.09	0.05	4.13	12.4	0.07	36.3	73.6	n/a	
Super Parsing [69]	87.0	67.1	96.9	62.7	95.9	95.9	14.7	17.9	1.7	70.0	19.4	51.2	83.3	n/a	
SegNet (3.5K dataset training - 140K) [77]	89.6	83.4	96.1	87.7	96.4	96.4	62.2	53.45	32.1	93.3	36.5	71.20	90.40	60.10	46.84
Boosting + pairwise CRF [59]	70.7	70.8	94.7	74.4	94.1	94.1	45.7	37.2	13.0	79.3	23.1	59.9	79.8	n/a	
Boosting+Higher order [58]	84.5	72.6	97.5	72.7	95.3	95.3	34.2	45.7	8.1	77.6	28.5	59.2	83.8	n/a	
Boosting+Detectors+ CRF [77]	72.5	76.6	96.2	78.7	93.9	93.9	43.0	47.6	14.3	81.5	33.9	62.5	83.8	n/a	
Our RL Method	**91.08**	**88.01**	**97.1**	**92.19**	**98.6**	**98.5**	**84.98**	**81.85**	**85.51**	**94.5**	**87.17**	**89.26**	**90.56**	**87.17**	**93.14**

TABLE 2. The comparison of our reinforced AL method with the deep neural networks for the task of semantic segmentation on the test set of CamVid dataset when the training is done on the 3433 road scenes corpus without class balancing. At a defined learning pace, when an end-to-end training is done at the same, our network performs better than the other methods. Our BF score has relatively high value than the other methods. The BF score (The parameter for the measurement of the interclass boundary delineation *Acc*) is much higher than the other models. DeconvNet is comparable to the SegNet metrics but it has a high computation cost.

Network/Iterations	G	C	mIoU	BF
SegNet [77]	88.81	59.93	50.02	35.78
DeepLab-LargeFOV [55]	85.95	60.41	50.18	26.25
DeepLab-LargeFOV-denseCRF [7]	Not being computed.			
FCN [6]	81.97	54.38	46.59	22.86
FCN (learnt deconv) [6]	83.21	56.05	48.68	27.40
DeconvNet [55]	85.26	46.40	39.69	27.36
Our RL Algorithm	**90.56**	**84.98**	**87.17**	**93.14**

TABLE 3. Quantitative comparison of our reinforced AL algorithm with the state-of-the-art methods on the 5250 indoor corpus of the SUNRGB-D dataset. The RGB modality is taken in account in these results. There are 37 classes in the complex task of the segmentation task, with all the methods performing poorly, especially because the classes which are skew in class distribution and small. It can be noted that our method received 75.82, 48.54, 62.94, 39.67, 77.25 (160K) as a metric while training with a median frequency class balancing.

Network/Iterations	G	C	mIoU	BF
SegNet [77]	70.73	30.82	22.52	9.16
DeepLab-LargeFOV [7]	70.73	41.75	30.67	7.28
DeepLab-LargeFOV-denseCRF [7]	Not Being Computed			
FCN (learnt deconv) [6]	67.31	34.32	24.05	7.88
DeconvNet [55]	59.62	12.93	8.35	6.50
Our RL Algorithm	**75.82**	**48.54**	**62.94**	**77.25**

due to the lack of a correct validation list. We infer from the results that our AL method performs significantly well with the other methods. It is able to demand more labels from under-represented group.

B. SUN RGB-D INDOOR SCENES

With 5285 training and 5050 test images, SUN RGB-D [9] is a dynamic and detailed indoor data collection. A network of sensors captures the images, but they are of different resolutions. The aim is to categorize 37 indoor classrooms using walls, floor, roof, table, chair, sofa, and others. The fact that the types of objects come with a wide range of sizes, heights, and forms also complicates the segmentation task. The partial occlusions are normal since there are typically several different groups participating in and of the sample images.

As a consequence, it is one of the most challenging datasets. In our method, the RGB modality is used for both the preparation and review. We see that by implementing the depth modalities, the architecture changes and redesigns [2]. Extensive post-processing is done to remove incorrect measurements from digital camera images. A vast number of images are used to measure the properties of secure segmentation. There are some variations between road scene images in terms of their spatial configurations. When the capture is done from a car moving, the camera is almost often parallel to the ground floor, reducing the variability of viewpoints. As a consequence, the deep networks learn to segment them effectively. On the other hand, the indoor scenes are more difficult for the images since the points of view and the number of people in the scene is inconsistent. The scene's target groups' different sizes introduce a new level of complexity to the scenario. The test samples from the most recent SUN RGB-D dataset [9] are seen in Figure 6. A few scenes from various large classes, even those with a lot of disturbance, are seen (bottom row and right). In an indoor environment, the object can be expressed in various ways (texture and shape). Other challenges, such as Pascal VOC12 [10] Salient Object Segmentation, have gained more attention. Still, we infer that indoor segmentation is more complicated and has more recent

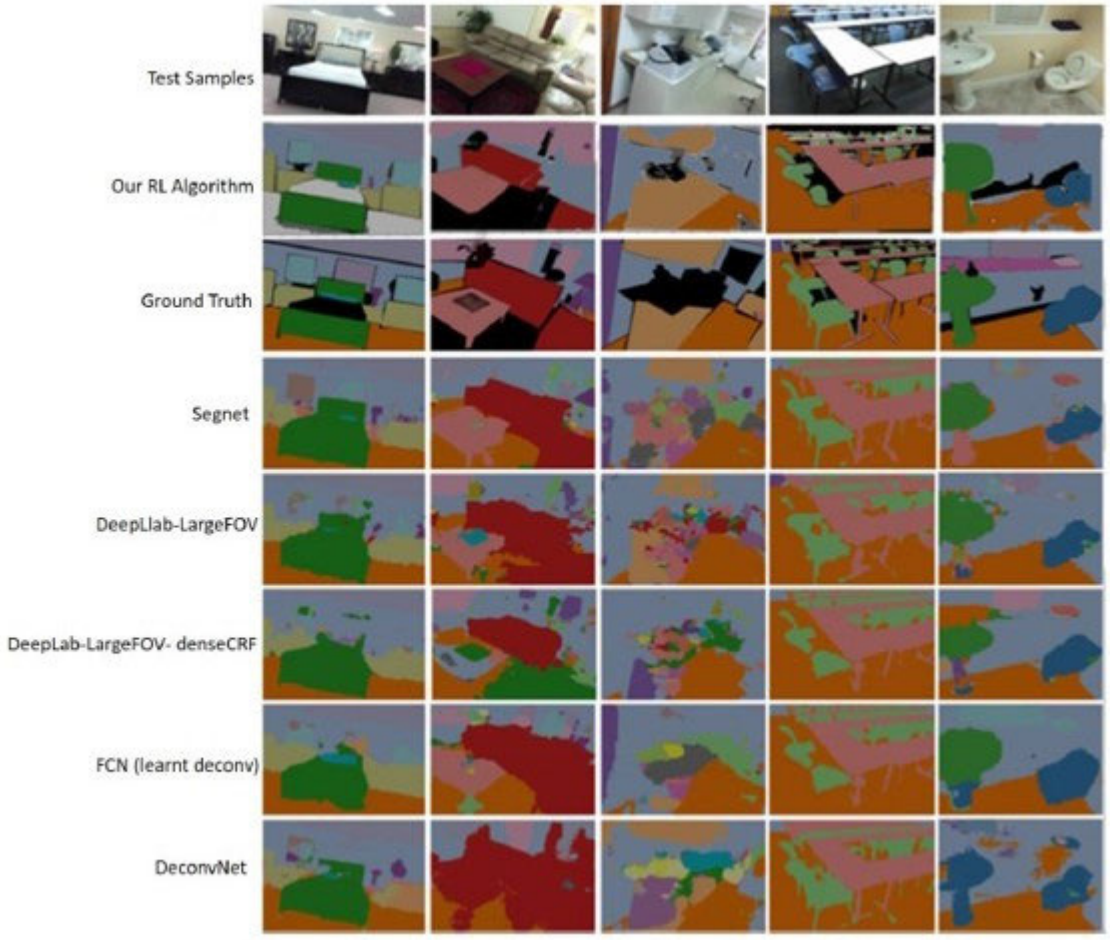

FIGURE 6. Our RL algorithm results on the recently released SUN RGB-D dataset which contains RGB indoor test scenes from the recently released SUN RGB-D dataset is evaluated qualitatively and compared to the current state of the art methods. Our model predicts by delineating the inter class borders better for groups of object in a number of view-points and scenes in this difficult task. Overall, the segmentation efficiency is best when the object classes are of reasonable size, but it performs well even if the scene is cluttered and it is quite noisy. It is worth noting that portions of the scene image that don't have ground truth labels are always displayed in black. The elements do not undergo masking in the corresponding deep neural network model projections.

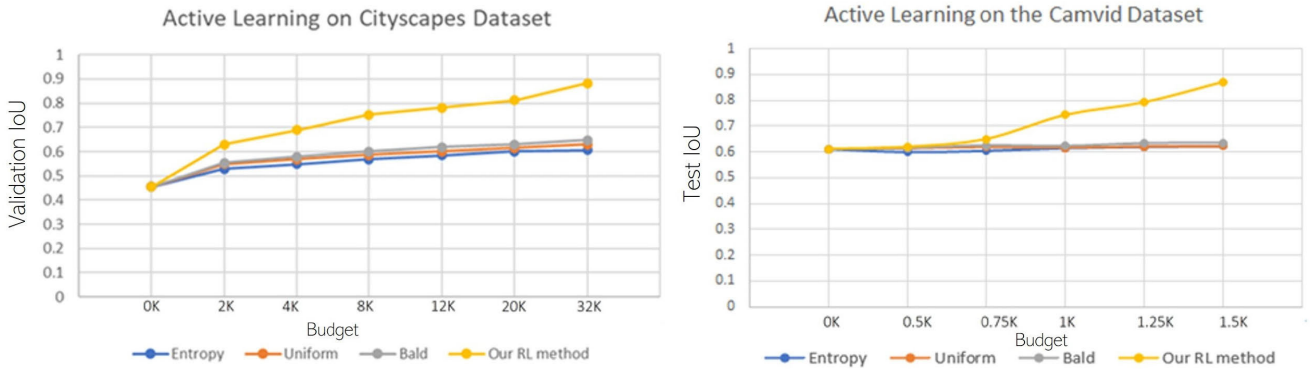

FIGURE 7. Comparison of our method's performance with the current state of the art methods on the increasing active learning budget, expressed as the % of additional labeled data and the number of 128 × 128-pixel regions. All methods have been pre-trained with a small subset of their target datasets and GTAV. The budget indicates the additional number of labeled regions (Unlabeled data percentage used).

applications, such as robotics and virtual reality. We compared our AL algorithm to well-known deep architectures evaluated on the SUN RGB-D dataset [9]. Figure 6 shows how our algorithm enhances the segmentation of various indoor scenes, such as kitchens, dining rooms, workplaces, conference rooms, and bathrooms. Since the class size is

large, we can see that our model allows correct predictions when seen from various angles.

This is particularly [65], [70]–[73] intriguing since RGB is the only input modality. Our method can also segment smaller objects such as chair and table legs, lamps, and other complex features to capture in-depth images from well-known sensors. This is seen in Figure 6 for our RL algorithm. It is also helpful in AR circumstances for segmenting decorative objects like wall paintings [74]–[76]. On the other hand, the *Acc* of segmentation is less accurate than in the outdoor scene. According to Table 3 quantitative results, both deep architectures have poor boundary metrics and mioU. The class averages and the global average of the mIoU are both slow. Both the deep architectures have identical mIoU and boundary metrics, according to the quantitative findings in Table 3. The global and class averages (both equal to mIou) are also low. In terms of G, C, and BF parameters, our device outperforms all other approaches. The vast group numbers in this segmentation task, each of which occupies a limited portion of the image and occurs infrequently, is one explanation for the overall bad performance.

The larger groups are of good consistency, but the precision of the smaller categories is poor. The more extensive collections and training methods prioritize highly beneficial class integration. We overcome the drawbacks of the previous techniques having poor results due to their deep architectures' failure to deal with a solid indoor-scene heterogeneity (all of which are based on the VGG architecture). Our AL method uses the smallest model and produces the highest precision in mIoU, thus overcoming the drawbacks of larger parameterizations in DeconvNet [7], FCN [6], SegNet [77] and other methods [55], [78]–[80]. The performance of the current state-of-the-art methods do not increase even after much longer training. Our AL algorithm proposes excellent results and can extract all the objects clearly, thus overcoming all the drawbacks in the current literature.

V. CONCLUSION

In this paper, we proposed a data-driven, region-oriented approach based on reinforced AL for semantic segmentation. The aim is to make the time-consuming and expensive method of manually obtaining pixel-by-pixel points with a human in a loop. Here, we presented a new formulation of DQN for learning the acquisition function that is well-tailored to the semantic segmentation's large-scale nature. Consequently, the approach is more computationally effective than active baselines, requiring fewer labeled data to obtain the same performance. Furthermore, our approach advocated for more labels for under-represented groups than baselines by expressly accounting for the per-class mIoU and specifying behavior and states class-aware representations. This increases efficiency and aids in the reduction of class imbalance. We also highlighted the possibility of defining an area better, which might help boost overall performance and incorporate domain adaptation for the learned policy, which would enable it to be transferred between datasets. Our deep RL region-based DQN method performed better in detecting eight groups in around 11 classes and achieves an *Acc* of 90.56%, a mIoU score of 87.17%, and a BF score of 93.14% on the SUNRGBD dataset. It also reaches an *Acc* of around 75.82% and a BF score of 77.25% on the SUNRGB indoor scenes where the objects/classes are challenging to interpret, thus outperforming the current state-of-the-art methods. Further, we would better emphasize that using a limited number of data, our proposed unsupervised deep enforce AL DQN can pursue it without the pre-processing of a huge number of data. It enables us to cope with our environmental process.

REFERENCES

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.
[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[3] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1915–1929, Aug. 2013.
[4] N. Höft, H. Schulz, and S. Behnke, "Fast semantic segmentation of RGB-D scenes with GPU-accelerated deep neural networks," in *Proc. Joint German/Austrian Conf. Artif. Intell. (Künstliche Intelligenz)*, 2014, pp. 80–85.
[5] R. Socher, C. C.-Y. Lin, A. Y. Ng, and C. D. Manning, "Parsing natural scenes and natural language with recursive neural networks," in *Proc. ICML*, 2011, pp. 1–8.
[6] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.
[7] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," 2014, *arXiv:1412.7062*.
[8] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 102–118.
[9] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 567–576.
[10] G. Pastore, F. Cermelli, Y. Xian, M. Mancini, Z. Akata, and B. Caputo, "A closer look at self-training for zero-label semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 2693–2702.
[11] Y. Gal, R. Islam, and Z. Ghahramani, "Deep Bayesian active learning with image data," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1183–1192.
[12] Y. Baram, R. El-Yaniv, and K. Luz, "Online choice of active learning algorithms," *J. Mach. Learn. Res.*, vol. 5, pp. 255–291, Mar. 2004.
[13] M. Liu, W. Buntine, and G. Haffari, "Learning how to actively learn: A deep imitation learning approach," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics (Long Papers)*, vol. 1, 2018, pp. 1874–1883.
[14] M.-H. Chung, M. Chignell, L. Wang, A. Jovicic, and A. Raman, "Interactive machine learning for data exfiltration detection: Active learning with human expertise," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2020, pp. 280–287.
[15] L. Yang, S. Hanneke, and J. Carbonell, "A theory of transfer learning with applications to active learning," *Mach. Learn.*, vol. 90, no. 2, pp. 161–189, Feb. 2013.
[16] N. Roy and A. McCallum, "Toward optimal active learning through Monte Carlo estimation of error reduction," in *Proc. ICML*, vol. 2, Williamstown, NJ, USA, 2001, pp. 441–448.
[17] T. Osugi, D. Kun, and S. Scott, "Balancing exploration and exploitation: A new algorithm for active machine learning," in *Proc. 5th IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2005, p. 8.
[18] P. Bachman, A. Sordoni, and A. Trischler, "Learning algorithms for active learning," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 301–310.
[19] M. Fang, Y. Li, and T. Cohn, "Learning how to active learn: A deep reinforcement learning approach," 2017, *arXiv:1708.02383*.
[20] R. Chan, M. Rottmann, F. Hüger, P. Schlicht, and H. Gottschalk, "Application of decision rules for handling class imbalance in semantic segmentation," 2019, *arXiv:1901.08394*.

[21] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," in *Proc. Int. Workshop Deep Learn. Med. Image Anal.*, 2017, pp. 240–248.

[22] M. Everingham, S. A. Eslami, L. V. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes challenge: A retrospective," *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, 2015.

[23] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognit. Lett.*, vol. 30, no. 2, pp. 88–97, Jan. 2009.

[24] S. Budd, E. C. Robinson, and B. Kainz, "A survey on active learning and human-in-the-loop deep learning for medical image analysis," *Med. Image Anal.*, vol. 71, Jul. 2021, Art. no. 102062.

[25] A. Cordier, D. Das, and P. Gutierrez, "Active learning using weakly supervised signals for quality inspection," 2021, *arXiv:2104.02973*.

[26] K. Yoon, J. Gwak, Y.-M. Song, Y.-C. Yoon, and M.-G. Jeon, "OneShotDA: Online multi-object tracker with one-shot-learning-based data association," *IEEE Access*, vol. 8, pp. 38060–38072, 2020.

[27] A. Ullah, K. Muhammad, K. Haydarov, I. U. Haq, M. Lee, and S. W. Baik, "One-shot learning for surveillance anomaly recognition using Siamese 3D CNN," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.

[28] T. Fechter and D. Baltas, "One-shot learning for deformable medical image registration and periodic motion tracking," *IEEE Trans. Med. Imag.*, vol. 39, no. 7, pp. 2506–2517, Jul. 2020.

[29] Z. Shi, J. A. Zhang, Y. D. R. Xu, and Q. Cheng, "Environment-robust device-free human activity recognition with channel-state-information enhancement and one-shot learning," *IEEE Trans. Mobile Comput.*, early access, Jul. 28, 2020, doi: 10.1109/TMC.2020.3012433.

[30] K. Ni, X. Yin, A. F. Laguna, S. Joshi, S. Duenkel, M. Trentzsch, J. Müller, S. Beyer, M. Niemier, X. S. Hu, and S. Datta, "Ferroelectric ternary content-addressable memory for one-shot learning," *Nature Electron.*, vol. 2, no. 11, pp. 521–529, Nov. 2019.

[31] K. Konyushkova, R. Sznitman, and P. Fua, "Learning active learning from data," 2017, *arXiv:1703.03365*.

[32] M. Woodward and C. Finn, "Active one-shot learning," 2017, *arXiv:1702.06559*.

[33] G. Contardo, L. Denoyer, and T. Artieres, "A meta-learning approach to one-step active learning," 2017, *arXiv:1706.08334*.

[34] S. Ravi and H. Larochelle, "Meta-learning for batch mode active learning," in *Proc. ICLR*, 2018.

[35] K. Konyushkova, R. Sznitman, and P. Fua, "Geometry in active learning for binary and multi-class image segmentation," *Comput. Vis. Image Understand.*, vol. 182, pp. 1–16, May 2019.

[36] E.-C. Huang, H.-K. Pao, and Y.-J. Lee, "Big active learning," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 94–101.

[37] J. Fan, Z. Wang, Y. Xie, and Z. Yang, "A theoretical analysis of deep Q-learning," in *Learning for Dynamics and Control*. ML Research Press, 2020, pp. 486–489.

[38] S. Gu, T. Lillicrap, Z. Ghahramani, R. E. Turner, B. Schölkopf, and S. Levine, "Interpolated policy gradient: Merging on-policy and off-policy gradient estimation for deep reinforcement learning," 2017, *arXiv:1706.00387*.

[39] Á. Peris and F. Casacuberta, "Active learning for interactive neural machine translation of data streams," 2018, *arXiv:1807.11243*.

[40] A. Padmakumar, P. Stone, and R. J. Mooney, "Learning a policy for opportunistic active learning," 2018, *arXiv:1808.10009*.

[41] K. Pang, M. Dong, Y. Wu, and T. Hospedales, "Meta-learning transferable active learning policies by deep reinforcement learning," 2018, *arXiv:1806.04798*.

[42] K. Pang, M. Dong, Y. Wu, and T. M. Hospedales, "Dynamic ensemble active learning: A non-stationary bandit with expert advice," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 2269–2276.

[43] A. Casanova, P. O. Pinheiro, N. Rostamzadeh, and C. J. Pal, "Reinforced active learning for image segmentation," 2020, *arXiv:2002.06583*.

[44] R. Mackowiak, P. Lenz, O. Ghori, F. Diego, O. Lange, and C. Rother, "CEREALS–cost-effective REgion-based active learning for semantic segmentation," 2018, *arXiv:1810.09726*.

[45] W.-N. Hsu and H.-T. Lin, "Active learning by learning," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2659–2665.

[46] Y. Zhang, P. Sun, Y. Yin, L. Lin, and X. Wang, "Human-like autonomous vehicle speed control by deep reinforcement learning with double Q-learning," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1251–1256.

[47] X. Zhang, C. Delpha, and D. Diallo, "Jensen–Shannon divergence for non-destructive incipient crack detection and estimation," *IEEE Access*, vol. 8, pp. 116148–116162, 2020.

[48] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, 1988.

[49] H. V. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, 2016.

[50] G. Csurka and F. Perronnin, "An efficient approach to semantic segmentation," *Int. J. Comput. Vis.*, vol. 95, no. 2, pp. 198–212, 2011.

[51] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[52] D. Theckedath and R. R. Sedamkar, "Detecting affect states using VGG16, ResNet50 and SE-ResNet50 networks," *Social Netw. Comput. Sci.*, vol. 1, no. 2, pp. 1–7, Mar. 2020.

[53] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[54] B. Hurl, K. Czarnecki, and S. Waslander, "Precise synthetic image and LiDAR (PreSIL) dataset for autonomous vehicle perception," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 2522–2529.

[55] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1520–1528.

[56] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian SegNet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," 2015, *arXiv:1511.02680*.

[57] C. Zhang, L. Wang, and R. Yang, "Semantic segmentation of urban scenes using dense depth maps," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2010, pp. 708–721.

[58] P. Sturgess, K. Alahari, L. Ladicky, and P. H. S. Torr, "Combining appearance and structure from motion features for road scene understanding," in *Proc. Brit. Mach. Vis. Conf.*, 2009, pp. 1–11.

[59] L. Ladicky, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr, "What, where and how many? Combining object detectors and CRFs," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2010, pp. 424–437.

[60] P. Kontschieder, S. Rota Bulo, H. Bischof, and M. Pelillo, "Structured class-labels in random forests for semantic image labelling," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2190–2197.

[61] M.-F. Balcan, A. Beygelzimer, and J. Langford, "Agnostic active learning," *J. Comput. Syst. Sci.*, vol. 75, no. 1, pp. 78–89, 2009.

[62] M. Wrenninge and J. Unger, "Synscapes: A photorealistic synthetic dataset for street scene parsing," 2018, *arXiv:1810.08705*.

[63] S. D. Jain and K. Grauman, "Active image segmentation propagation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2864–2873.

[64] J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[65] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2008, pp. 44–57.

[66] P. Sturgess, K. Alahari, L. Ladicky, and P. H. S. Torr, "Combining appearance and structure from motion features for road scene understanding," in *Proc. Brit. Mach. Vis. Conf.*, 2009, pp. 1–11.

[67] S. R. Bulo and P. Kontschieder, "Neural decision forests for semantic image labelling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 81–88.

[68] Y. Yang, Z. Li, L. Zhang, C. Murphy, J. V. Hoeve, and H. Jiang, "Local label descriptor for example based semantic image labeling," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2012, pp. 361–375.

[69] J. Tighe and S. Lazebnik, "Superparsing: Scalable nonparametric image parsing with superpixels," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 352–365.

[70] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 391–405.

[71] L. Ladicky, C. Russell, P. Kohli, and P. H. S. Torr, "Associative hierarchical CRFs for object class image segmentation," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 739–746.

[72] P. Kohli, L. Ladický, and P. H. S. Torr, "Robust higher order potentials for enforcing label consistency," *Int. J. Comput. Vis.*, vol. 82, no. 3, pp. 302–324, May 2009.

[73] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," 2016, *arXiv:1606.02147*.
[74] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "TextonBoost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context," *Int. J. Comput. Vis.*, vol. 81, no. 1, pp. 2–23, Dec. 2007.
[75] S. Gould, R. Fulton, and D. Koller, "Decomposing a scene into geometric and semantically consistent regions," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 1–8.
[76] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for real-time semantic segmentation on high-resolution images," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 405–420.
[77] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.
[78] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 325–341.
[79] J. Xiao and L. Quan, "Multiple view semantic segmentation for street view images," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 686–693.
[80] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, "Understanding convolution for semantic segmentation," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 1451–1460.

USMAN AHMAD USMANI was born in Aligarh, India, in April 1993. He is currently pursuing the Ph.D. degree in computer science with Univeriti Teknologi Petronas, Malaysia. He worked as a Research Assistant with IIT Kanpur and a Researcher with Massey University, New Zealand. He has built up a social network named Zamber that has been published in around 14 national newspapers. His research interests include artificial intelligence, computer vision, computer security, wearable sensors, and cloud computing.

JUNZO WATADA received the B.Sci. and M.Sci. degrees in electrical engineering from Osaka City University, Japan, and the Ph.D. degree from Osaka Prefecture University, Japan. After retiring from Waseda University, Japan, he contributed as a Full Professor with the Department of Computer and Information Sciences, Universiti Technologi Petronas, Malaysia, and a Professor Emeritus with Waseda University. His research interests include big data analytics, soft computing, image processing systems to track human behaviors and understand pictures and videos, knowledge engineering, and management engineering. He is a Life Fellow of the Bio Medical Fuzzy System Association (BMFSA) and the Japan Society of Fuzzy Theory and Intelligent Informatics.

JAFREEZAL JAAFAR (Senior Member, IEEE) received the Ph.D. degree from The University of Edinburgh, U.K., in 2009. He is currently an Associate Professor and the former Head of the Computer and Information Sciences Department, Universiti Teknologi Petronas, Malaysia. He has secured a number of research projects from the industry and government agencies. His main research interests include big data analytics, soft computing, and software engineering. Based on his publication track records, he had been appointed as the chief editor and a reviewer for several journals, and the chair, the technical chair, and a committee member for several international conferences. He is also active in the IEEE Computer Society, Malaysia Chapter, and has been appointed as a member of the Executive Committee for 2016 and 2017.

IZZATDIN ABDUL AZIZ received the Ph.D. degree in information technology from Deakin University, Australia, working on the domain of hydrocarbon exploration and cloud computing. He is currently a Researcher with the High Performance Cloud Computing Centre (HPC3), Universiti Teknologi Petronas (UTP), where he focuses on solving complex upstream oil and gas (O&G) industry problems from the view point of computer sciences. He also serves as the Deputy Head of the Computer and Information Sciences Department, UTP. He is working closely with O&G companies in delivering solutions for complex problems, such as offshore O&G pipeline corrosion rate prediction, O&G pipeline corrosion detection, securing data on clouds, designing and implementing Metocean prediction system, and bridging upstream and downstream oil and gas businesses through data analytics. He is also working on big data transmission, security, and optimization problems on high performance clouds.

ARUNAVA ROY received the Ph.D. degree from the Department of Applied Mathematics, Indian School of Mines, Dhanbad. He currently works as a Researcher with the Department of Industrial and Systems Engineering, National University of Singapore, Singapore. Previously, he was a Postdoctoral Fellow with the Department of Computer Science, The University of Memphis, TN, USA. His research interests include web software reliability, software reliability, cyber security, algorithm design and analysis, data structure, and statistical and mathematical modeling.