Learning Dynamic Hierarchical Models for Anytime Scene Labeling

Buyu Liu[1,2(✉)] and Xuming He[1,2(✉)]

[1] The Australian National University, Canberra, Australia
{buyu.liu,xuming.he}@anu.edu.au
[2] Data61, CSIRO, Canberra, Australia

Abstract. With increasing demand for efficient image and video analysis, test-time cost of scene parsing becomes critical for many large-scale or time-sensitive vision applications. We propose a dynamic hierarchical model for anytime scene labeling that allows us to achieve flexible trade-offs between efficiency and accuracy in pixel-level prediction. In particular, our approach incorporates the cost of feature computation and model inference, and optimizes the model performance for any given test-time budget by learning a sequence of image-adaptive hierarchical models. We formulate this anytime representation learning as a Markov Decision Process with a discrete-continuous state-action space. A high-quality policy of feature and model selection is learned based on an approximate policy iteration method with action proposal mechanism. We demonstrate the advantages of our dynamic non-myopic anytime scene parsing on three semantic segmentation datasets, which achieves 90 % of the state-of-the-art performances by using 15 % of their overall costs.

1 Introduction

A fundamental and intriguing property of human scene understanding is its efficiency and flexibility, in which vision systems are capable of interpreting a scene at multiple levels of details given different time budgets [1,2]. Despite much progress in the pixel-level semantic scene parsing [3–6], most efforts are focused on improving the prediction accuracy with complex structured models [7,8] and learned representations [9–11]. Such computation-intensive approaches often lack the flexibility in trade-off between efficiency and accuracy, making it challenging to apply them to large-scale data analysis or cost-sensitive applications.

In order to improve the efficiency in scene labeling, a common strategy is to develop active inference mechanisms for the structured models used in this task [12,13]. This allows users to adjust the trade-off between efficiency and accuracy for a *given* model, which is learned using a separate procedure with unconstrained test-time budget. However, this may lead to a sub-optimal performance for the cost-sensitive tasks.

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-46466-4_39) contains supplementary material, which is available to authorized users.

B. Leibe et al. (Eds.): ECCV 2016, Part VI, LNCS 9910, pp. 650–666, 2016.
DOI: 10.1007/978-3-319-46466-4_39

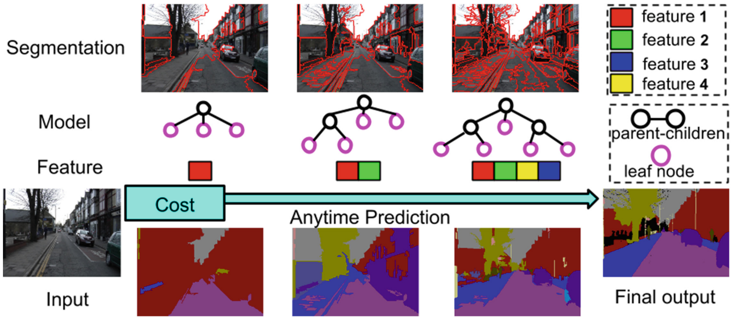

Fig. 1. Overview of our approach. We propose to incrementally increase model complexity in terms of used image features and model structure. Our approach generates high-quality prediction at any given cost.

A more appealing approach is to learn a model representation *for* Anytime performance, which can stop its inference at any cost budget and achieve an optimal prediction performance under the cost constraint [14]. While such learned representations have shown promising performance in anytime prediction, most work address the unstructured classification problems and focus on efficient feature computation [2,15,16]. Only recent work of Grubb et al. [17] proposes an anytime prediction method for scene parsing, which relies on learning a representation for individual segments. Nevertheless, to achieve coherent scene labeling, it is important to learn a representation that also encodes the relations between scene elements (e.g., segments).

In this work, we tackle the anytime scene labeling problem by learning a family of structured models that captures long-range dependency between image segments. Labeling with structured models, however, involves both feature computation *and* inference cost. To enable anytime prediction, we propose to generate scene parsing from spatially coarse to fine level and with increasing number of image features. Such a strategy allows us to control both feature computation cost *and* the model structure which determines the inference cost.

Specifically, we design a hierarchical model generation process based on growing a segmentation tree for each image. Starting from the root node, this process gradually increases the overall model complexity by either splitting a subset of leaf-node segments or adding new features to label predictors defined on the leaf nodes. At each step, the resulting model encodes the structural dependency of labels in the hierarchy. For any cost budget, we can stop the generation process and produce a scene labeling by collecting the predictions from leaf nodes. An overview of our coarse-to-fine scene parsing is shown in Fig. 1. We note that a large variety of hierarchical models can be generated with different choices of node splitting and feature orders.

To achieve aforementioned Anytime performance, we seek a policy of generating the hierarchical models which produce high-quality or optimal pixel-level label predictions for any given test budget. We follow the anytime setting in [15,16], in which the test-time budget is *unknown* during model learning. Instead of learning a greedy strategy, we formulate the anytime scene labeling as a sequential decision making problem, and define a finite-horizon Markov Decision Process (MDP). The MDP maximizes the average label accuracy improvement per unit cost (or the average 'speed' of improvement if the cost is time) over a range of cost budgets as a surrogate for the anytime objective, and has a parametrized discrete action space for expanding the hierarchical models.

We solve the MDP to obtain a high-quality policy by developing an approximate least square policy iteration algorithm [18]. To cope with the parametrized action space, we propose an action proposal mechanism to sample a pool of candidate actions, of which the parameters are learned based on several greedy objectives and on different subsets of images. We note that the key properties of our learned policy are *dynamic*, which generates an image-dependent hierarchical representation, and *non-myopic*, which takes into account the potential future benefits in a sequence of predictions.

We evaluate our dynamic anytime parsing method on three publicly available semantic segmentation benchmarks, CamVid [19], Stanford Background [20] and Siftflow [21]. The results show that our approach is favorable in terms of anytime scene parsing compared to several state-of-the-art representation learning strategies, and in particular we can achieve 90 % of the state-of-the-art performances within 15 % of their total costs.

2 Related Work

Semantic scene labeling has become a core problem in computer vision research [3]. While early efforts tend to focus on structural models with hand-crafted features, recent work shift towards deep convolutional neural network based representation with significant improvement on prediction accuracy [4,6,10]. Hierarchical models, such as dynamic trees [22], segmentation hierarchies [23–26] and And-Or graphs [27], have adopted for semantic parsing. However, in general, those methods are expensive to deploy due to complex model inference or costly features.

Most of prior work on efficient semantic parsing focus on the active inference, which assumes redundancy in pre-learned models and achieves efficiency by allocating resource to an informative subset of model components. Roig et al. [12] use perturb-and-MAP inference model to select informative unary potentials to compute. Liu and He [28] actively select most-rewarding subgraphs for video segmentation. In [29], a local classifier is learned to select views for multi-view semantic labeling. Unlike these methods, we explicitly learn a representation for achieving strong performance at any test-time budget.

Learning anytime representation has been extensively explored for unstructured prediction problems (e.g., classification) [15,16]. Karayev et al. [2] learn an anytime representation for object and scene recognition, focusing on dynamic feature selection under a total budget. Weiss and Taskar [13] develop a reinforcement learning framework for feature selection in structured models. In contrast, we consider both feature computation and model inference cost. More importantly, we incorporate the cost in an MDP reward which encourages anytime property. Unlike [2], the test-time budget is explicitly unknown during learning in our setting. Perhaps the most related work is [17], which learns a segment-based anytime representation consisting of a selection function and a boosted predictor for individual segments. Their policy of segment and feature selection is trained in a greedy manner based on [16] and a single strategy is applied to all the images. By contrast, we build a structured hierarchical model on segmentation trees and learn an image-adaptive policy.

More generally, cost-sensitive learning and inference have been widely studied in learning and vision literature under various different contexts, including feature selection [30], learning classifier cascade by empirical risk minimization [31,32] or Wald's sequential ratio test [33], model selection [34,35], prioritized message passing inference [36], object detection [37], and activity recognition [38]. However, few approaches have been designed for optimizing the anytime prediction performance [14], or considering both feature and inference costs. We note that while the MDP framework has been extensively used in those methods, our formulation of discrete-continuous MDP is tailored for anytime scene parsing.

Unfolding and learning inference in graphical models has been explored in various inference machines [26,39]. Nevertheless, such methods usually use a greedy approach to learn the messages or model predictions. [40] use reinforcement learning to obtain a dynamic deep network model, but they do not address the structured prediction problem. Lastly, we note that, although some search-based structured prediction methods [41,42] are capable of terminating inference and generating outputs at any time, they usually do not consider feature computation cost and are not optimized for anytime performance.

3 Anytime Scene Labeling with a Hierarchical Model

We aim to learn a structured model representation with anytime performance property for semantic scene labeling. As structured prediction involves both feature computation and inference, we need a flexible representation that allows us to control the cost of feature and inference computation. To this end, we first introduce a family of hierarchical models based on image segmentation trees in Sect. 3.1, which is capable of incrementally increasing its complexity in terms of used image features and model structure.

We then formulate the anytime scene labeling as a sequential feature and model selection process in this model family with a cost-sensitive labeling loss in Sects. 3.2 and 3.3. Based on an MDP framework, our goal is to learn an optimal selection policy to generate a sequence of hierarchical models from a set of annotated images. In Sect. 4, we develop an iterative procedure to solve the policy learning problem approximately.

3.1 Coarse-to-Fine Scene Parsing with a Segmentation Hierarchy

We now introduce a flexible hierarchical representation for semantic parsing that enables us to control the test-time complexity. To achieve effective semantic labeling, we want to design a model framework capable of incorporating rich image features, modeling long-range dependency between regions and achieving anytime property. To this end, we adopt a coarse-to-fine scene labeling strategy, and consider a family of hierarchical models built on image segmentation trees, which has a simplified form of the Hierarchical Inference Machine (HIM) [26].

Specifically, given an image I, we construct a sequence of segmentation trees by recursively partitioning the image using graph-based algorithms [43,44]. We then develop a sequence of hierarchical models that predict label marginal distributions on the leaf nodes of the segmentation trees. Formally, let the semantic label space be $\mathcal{Y}$. We start from an initial segmentation tree $\mathcal{T}^0$ with a single node and a marginal distribution $\mathcal{Q}^0 = \{\mathbf{q}^0\}$ on the node, which can be uniform or a global label prior. We incrementally grow the tree and update the prediction of marginal distributions on the leaf nodes by two update operators described in detail below, which generates a sequence of hierarchical models for labeling, denoted by $\mathcal{M}^1, \cdots, \mathcal{M}^T$, where T is the total number of steps.

At each step t, the hierarchical model $\mathcal{M}^t$ consists of a tree $\mathcal{T}^t$ and a set of predicted label distributions on the tree's leaf nodes, $\mathcal{Q}^t$. More concretely, we denote the leaf nodes of $\mathcal{T}^t$ as $\mathcal{B}^t = \{b_1, \cdots, b_{N_t}\}$ where N_t is the number of leaf nodes. We associate each leaf-node segment b_i with a label variable y_i^t indicating its dominant label assignment. Let the label distributions $\mathcal{Q}^t = \{\mathbf{q}_i^t\}_{i=1}^{N_t}$, where $\mathbf{q}_i^t$ is the current label marginals at node b_i. We generate the next hierarchical model $\mathcal{M}^{t+1}$ by applying the following two update operators.

Split-inherit update. We choose a subset of leaf-node segments and split them into finer scale segments in the segmentation tree. The selection criterion is based on the entropy of the node marginals $H(\mathbf{q}_i^t)$, and all the nodes with $H(\mathbf{q}_i^t) > \theta_t$ will split into their children [17]. θ_t is a parameter of the operator and $\theta_t \in \mathbb{R}$. The new leaf-node segments inherit the marginal distributions of their parents.

$$\mathbf{q}_i^{t+1}(k) = \mathbf{q}_{pa(i)}^t(k), \quad k \in \mathcal{Y},\ i \in \mathcal{B}^{t+1} \tag{1}$$

where $\mathcal{B}^{t+1}$ is the new leaf node set and $pa(i)$ indicates the parent node of i in the new tree $\mathcal{T}^{t+1}$. We denote the parameter space of the operator as Θ.

Local belief update. For the newly generated leaf nodes from splitting, we improve their marginal distributions by adding more image cues or context information from their parents. Specifically, we extract a set of input features $\mathbf{x}_i$ from segment b_i, and adopt a boosting-like strategy: Using a weak learner taking the image feature $\mathbf{x}_i$ and the marginal of its parent $\mathbf{q}_{pa(i)}^t$ as input [26], we update the marginals of leaf nodes as follows,

$$\mathbf{q}_i^{t+1}(k) \propto \mathbf{q}_i^t(k) \exp\left(\alpha_t h_k^t(\mathbf{f}_i^t(j))\right), \quad k \in \mathcal{Y}, \quad \mathbf{f}_i^t = [\mathbf{x}_i, \mathbf{q}_{pa(i)}^t] \tag{2}$$

where $\mathbf{f}_i^t(j)$ is the j-th feature used in the weak learner; $\mathbf{h}_t = [h_t^1, ..., h_t^{|\mathcal{Y}|}]$ and α_t are the newly added weak learners and their coefficient, respectively. We denote the weak learner space as $\mathcal{H}$ and $\alpha_t \mathbf{h}_t \in \mathcal{H}$.

By applying a sequence of these update operators to the segmentation tree from its root node, we can generate a dynamically growing hierarchical models

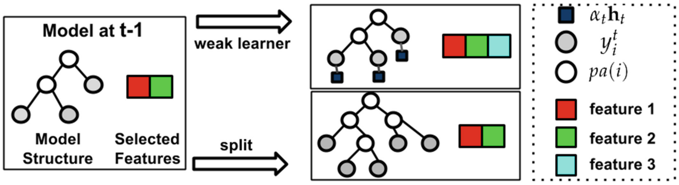

Fig. 2. Example of operators at step t. We choose either to add weak learners or split a subset of leaf nodes, which leads to gradually increasing model complexity.

for scene labeling (see Fig. 2 for an illustration). We refer to the resulting structured models as the *Dynamic Hierarchical Model* (DHM). We use 'dynamic' to indicate that our model generation process can vary from image to image or given different choices of the operators, which is not predetermined by greedy learning as in [26]. Using DHM as our representation for anytime scene labeling has several advantages. First, a DHM is capable of generating a sequence of model predictions with incrementally increasing cost as every update operator can be computed efficiently. In addition, it utilizes multiscale region grouping to create models from coarse to fine level, leading to gradually increasing model complexity. Furthermore, it has a flexible structure to select image features by weak learners and to capture long-range dependency between segments, which is critical to achieve the state-of-the-art performance for any test-time budget.

3.2 Anytime Scene Labeling by Cost-Sensitive DHM Generation

Given the dynamic hierarchical scene models defined in Sect. 3.1, we now formulate the anytime scene labeling as a cost-sensitive DHM generation problem. Specifically, we want to find a model generation strategy, which selects an sequence of image-dependent update operators, such that the incrementally built hierarchical models achieve good performance (measured by average labeling accuracy) at all possible test-time cost budgets. To address this sequential selection problem, we model the cost-sensitive model generation as a Markov Decision Process (MDP) that encourages good anytime performance with a cost-sensitive reward function. By solving this MDP, we are able to find a policy of selection that yields a sequence of hierarchical models with high-quality anytime performance.

Concretely, we first model an episode of coarse-to-fine DHM generation as an MDP with finite horizon. This MDP consists of a tuple $\{\mathcal{S}, \mathcal{A}, T(\cdot), R(\cdot), \gamma\}$, which defines its state space, action space, state transition, reward function and a discounting factor, respectively.

State: At time t, the state $s_t \in \mathcal{S}$ represents the current segment set corresponding to the leaf nodes of the segmentation tree and the label marginal distributions on the leaf nodes. As in Sect. 3.1, we denote the leaf-node segment set and the corresponding marginal label distributions as $\mathcal{B}^t = \{b_i\}_{i=1}^{N_t}$ and $\mathcal{Q}^t = \{\mathbf{q}_i^t\}_{i=1}^{N_t}$ respectively. We also introduce an indicator vector $Z^t \in \{0,1\}^{N_t}$ to describe an active set of leaf-nodes at t, in which $Z^t(k) = 1$ indicates the leaf node b_k is newly generated by splitting. Altogether, we define $s_t = \{\mathcal{B}^t, \mathcal{Q}^t, Z^t\}$.

Action: The action set $\mathcal{A}$ consists of the two types of update operators defined in Sect. 3.1. We denoted them by $\{u_s(\theta), u_b(\alpha\mathbf{h})\}$. For $a_t = u_s(\theta_t)$, we choose to split a subset of leaf-node segments of which the entropy of predicted marginal distributions are greater than θ_t. For $a_t = \alpha_t\mathbf{h}_t$, we apply the local belief update to the active nodes in Z^t using the weak learner $\alpha_t\mathbf{h}_t \in \mathcal{H}$. Note that the action space $\mathcal{A}$ is a discrete-continuous space $\Theta \cup \mathcal{H}$ due to their parameterization.

State Transition: The state transition $T(s_{t+1}|s_t, a_t)$ is a deterministic function in our MDP. For $a_t = \theta_t$, it expands the tree and generates a new set of leaf-node segments $\mathcal{B}^{t+1}$ with inherited marginals $\mathcal{Q}^{t+1}$ as defined in Eq. (1). The new active regions are the newly generated leaf-nodes from splitting, denoted by Z^{t+1}. The action $a_t = \alpha_t\mathbf{h}_t$ keeps the tree structure and active regions unchanged, such that $\mathcal{B}^{t+1} = \mathcal{B}^t$ and $Z^{t+1} = Z^t$; while it only updates the node marginals $\mathcal{Q}^{t+1}$ according to Eq. (2).

Reward Function and γ: The reward function R defines a mapping from (s_t, a_t) to rewards in $\mathbb{R}$ and γ is a discount factor that determines the lookahead in selection actions. For the anytime learning problem, we design a reward function that is cost-sensitive and encourages the sequence of generated models can achieve good labeling accuracies across a range of possible cost budgets. The details of the reward function and γ will be discussed in the next subsection.

3.3 Defining Reward Function

We now define a reward function that favors a coarse-to-fine dynamic hierarchical model generation with anytime performance. To this end, we first describe the action costs of the MDP, which compute the overall cost of model prediction. We then introduce a labeling loss for our hierarchical models, based on which the cost-sensitive reward and the value function of the MDP are defined.

Action Cost: The action cost represents the cost of scene labeling using a hierarchical model, which consists of feature extraction cost c_{f_t} for computing feature set f_t (from the entire image or specific regions), region split cost c_r for pooling features for newly split regions, total weak learner cost c_{h_t} for applying the weak learner $\alpha_t\mathbf{h}_t$ to predict labels. For each action a_t, we define the action cost $c(a_t)$ as $c_{h_t} + c_{f_t}$ if $a_t = u_b$, or c_r if $a_t = u_s$. In this work, we use the CPU time used in a_t as a surrogate for the computation cost while any other type of costs can also be applied.

Labeling loss of DHMs: Given a hierarchical model represented by s_t, we introduce a loss function measuring the scene labeling performance. Particularly, we adopted an entropy-based labeling loss function defined as follows,

$$\mathcal{L}(s_t, \hat{\mathbf{Y}}|I) = -\sum_{i\in\mathcal{B}^t} w_i\mathbf{p}_i^T \log(\mathbf{q}_i^t) - \alpha \sum_{i\in\mathcal{B}^t} w_i\mathbf{p}_i^T \log(\mathbf{p}_i), \tag{3}$$

where $\mathbf{p}_i$ is the ground-truth label distribution in region $b_i \in \mathcal{B}^t$, derived from the ground-truth labeling $\hat{\mathbf{Y}}$, and w_i denotes its normalized size. The first term is the cross-entropy between the marginals and the ground-truth while the second term penalizes the regions with mixed labels. These two terms reflect the label prediction and image partition quality respectively and we further introduce a weight α to control their balance. Intuitively, the loss favors a model with a sensible image segmentation and a good prediction for the segment labels. A larger α prefers to learn predictors after reaching fine levels in hierarchy while a small value may lead to stronger predictors at all levels.

Cost-sensitive reward: To achieve the anytime performance, an ideal model generation sequence will minimize the labeling loss as fast as possible such that it can obtain high quality scene labeling for a full range of cost budgets. Following this intuition, we define the reward for action a_t as the labeling loss improvement between s_{t+1} and s_t normalized by the cost of a_t [16]. Formally, we define the reward as,

$$R(s_t, a_t|I) = \frac{1}{c(a_t)}\left[\mathcal{L}(s_t, \hat{\mathbf{Y}}|I) - \mathcal{L}(s_{t+1}, \hat{\mathbf{Y}}|I)\right] \tag{4}$$

where $c(a_t)$ summarizes all the computation cost in the action a_t.

Policy and value function: A policy of the MDP is a function mapping from a state to an action, $\pi(s) : \mathcal{S} \to \mathcal{A}$. The value function of the MDP at state s_t under policy π is the total accumulated reward defined as,

$$V_\pi(s_t) = \sum_{\tau=t}^{T} \gamma^{\tau-t} R(s_\tau, \pi(s_\tau)|I) \tag{5}$$

where T is the number of actions taken and s_0 is the initial state. Our goal is to find an optimal policy π^* that maximizes the expected value function over the image space for any state s_t. We will discuss how to learn such a policy using a training set in the following section. We note that our objective describes a weighted average speed of labeling performance improvement (c.f. (4), (5)), and γ controls how greedy the policy would be. When $\gamma = 0$, the optimal policy maximizes a myopic objective as in [16]. We choose $\gamma > 0$ so that our policy also considers potential future benefit (i.e., fast improvement in later stages).

4 Learning Anytime Scene Labeling

To learn anytime scene labeling, we want to seek a policy π^* to maximize the expected value function for any state s_t in a MDP framework. Given a training set $\mathcal{D} = \{I^{(m)}, \hat{\mathbf{Y}}^{(m)}\}_{m=1}^{M}$ with M images, the learning problem is defined as,

$$\pi^*(s_t) = \operatorname*{argmax}_{\pi} E_D[V_\pi(s_t)] = \operatorname*{argmax}_{\pi} \frac{1}{M} \sum_{m=1}^{M} \sum_{\tau=t}^{T} \gamma^{\tau-t} R(s_\tau, \pi(s_\tau)|I^m), \ \forall t \tag{6}$$

where E_D is the empirical expectation on the dataset $\mathcal{D}$. The main challenge in solving this MDP is to explore the parametrized action space $\mathcal{A}$ due to its discrete-continuous nature and high-dimensionality. In this work, we design an action generation strategy that proposes a finite set of effective parameters for the actions. We then use the proposed action pool as our discrete action space and develop a least square policy learning procedure to find a high quality policy.

4.1 Action Proposal Generation

To cope with the parameterized actions, we discretize the parameter space $\Theta \cup \mathcal{H}$ by generating a finite set of effective and diversified parameter values. Our discretization uses a greedy learning criterion to generate a sequence of actions with instantiated parameters based on the training set $\mathcal{D}$.

Specifically, we start from s_0 for all the training images, and generate a sequence of actions and states (which corresponds to a sequence of hierarchical models) as follows. At step t, we first discretize Θ by uniformly sample the 1D space. For the weak learner space $\mathcal{H}$, we generate a set of weak learners by minimizing the following regression loss as in the Greedy Miser method [45]:

$$\alpha_t, \mathbf{h}_t = \arg\min_{\alpha, \mathbf{h}} \sum_{i \in D^t} w_i \|\mathbf{p}_i - \mathbf{q}_i^{t-1} - \alpha \mathbf{h}(\mathbf{f}_i^t)\|^2 + \lambda(c_{h_t} + c_{f_t}) \tag{7}$$

where $D^t = \{i | Z_m^t(i) = 1, m = 1, \ldots, M\}$ is the set of all active nodes at step t in all M images, and $\mathbf{p}_i$ and $\mathbf{q}_i^{t-1}$ are the ground-truth marginal and the previous marginal prediction on node i respectively. The second term regularizes the loss with the cost of applying the weak learner and a weight parameter λ controls its strength. We obtain several weak learner $\alpha_t \mathbf{h}_t$ by varying the value of λ. From these discretized actions, denoted by $\mathcal{A}_s^t$, we then select a most effective action using our reward function, $a_t^0 = \arg\max_{a_t \in \mathcal{A}_s^t} E_D[R(s_t, a_t | I)]$. We continue this process until step T based on a held-out validation set, and $\{a_t^0\}_{t=1}^T$ is a sampled action sequence.

To increase the diversity of our discrete action candidates, we also apply the same action proposal generation method to different subsets of images. The image subsets are formed by K-means clustering and we refer the reader to the supplementary for the details. Finally we combine all the generated discrete action sequences as our action candidates to form a new discrete action space $\mathcal{A}^d$, which is used for learning our policy.

4.2 Least-Square Policy Iteration for Solving MDP

In order to find a high-quality policy π_d^* on $\mathcal{A}^d$, we adopt an approximate least-square policy iteration approach and learn a parametrized Q-function [13,18], which can be generalized to the test scenario. Specifically, we use a linear function to approximate the Q-function, and the approximate Q and corresponding policy can be written as

$$\hat{Q}(s_t, a_t) = \eta^T \phi(s_t, a_t), \tag{8}$$

$$\pi_d(s_t) = \arg\max_{a_t \in \mathcal{A}^d} \hat{Q}(s_t, a_t) \tag{9}$$

where $\phi(s_t, a_t)$ is the meta-feature of the model computed from the current state s_t and action a_t. η is the linear coefficient to be learned. We will discuss the details of our meta-feature in Sect. 5.1.

Our least-square policy iteration procedure includes the following three steps, which starts from an initial policy π_d^0 and iteratively improves the policy.

A. Policy initialization. We initialize the policy π_d^0 by a greedy action selection that optimizes the average immediate reward on the training set at each time step. Specifically, at each t, we choose $\pi_d^0(s_t) = \arg\max_{a_t \in \mathcal{A}^d} E_D[R(s_t, a_t|I)]$.

B. Policy evaluation. Given a policy π_d^n at iteration n, we execute the policy for each training example to generate a trajectory $\{(s_t^m, a_t^m)\}_{m=1}^M$. We then compute the value function of the policy recursively based on $Q_\pi(s_t, a_t) = R(s_t, a_t|I) + \gamma Q_\pi(s_{t+1}, a_{t+1})$. As in [13], we only consider the non-negative contribution of Q_π, which allows early stop if the reward is no longer positive,

$$Q_\pi(s_t^m, a_t^m) = R(s_t^m, a_t^m) + \gamma [Q_\pi(s_{t+1}^m, a_{t+1}^m)]_+ \tag{10}$$

C. Policy improvement. Given a set of trajectories $\{(s_t^m, a_t^m)\}_{m=1}^M$ and the corresponding Q-function value $\{Q_\pi(s_t^m, a_t^m)\}_{m=1}^M$, we update the linear approximate $\hat{Q}$ by solving the following least-square regression problem:

$$\min_\eta \beta \|\eta\|^2 + \frac{1}{TM} \sum_m \sum_t \left(\eta^T \phi(s_t^m, a_t^m) - Q_\pi(s_t^m, a_t^m) \right)^2 \tag{11}$$

where the iteration index n is omitted here for clarity. Denote the solution as η^*, we can compute the new updated policy $\pi_d^{n+1}(s_t) = \arg\max_{a_t} \eta^* \phi(s_t, a_t)$. We also add a small amount of uniformly distributed random noise to the updated policy as in [2]. We perform policy evaluation (Step B) and improvement iteration (Step C) several times until the segmentation performance does not change in a held-out validation set.

During the test, we apply the learned policy π_d to an test image, which produces a trajectory $\{(s_0, a_0), (s_1, a_1), \ldots, (s_T, a_T)\}$. The state sequence defines a coarse-to-fine scene labeling process based on the generated hierarchical models. For any given cost budget, we can stop the scene labeling process and use the leaf-node marginal label distributions (i.e., taking the most likely label) to make a pixel-wise label prediction for the entire image. More detailed discussion of the test-time procedure can be found in the supplementary.

5 Experiments

We evaluate our method on three publicly available datasets, including CamVid [19], Standford Background [20] and Siftflow [21]. We focus on CamVid [19] as it provides more complex scenes with multiple foreground object classes of various scales. We refer the reader to supplementary material for the details of datasets.

5.1 Implementation Details

Feature set and action proposal: We extract 9 different visual features (Semantics using Darwin [46], Geometric, Color and Position, Texture, LBP, HoG, SIFT and hyper-column feature [4,5]). In action proposal, the weak learner $\alpha_t \mathbf{h}_t$ is learned as in Sect. 4.1 using [45]. To propose multiple weak learners with a variety of costs, we also learn p weak learners sequentially where p is set to 5,10 and 20 empirically and use them as action candidates. As for split action, we discretize Θ into $\{0, 0.3, 0.6, 1\}$ and we generate a 8-layer hierarchy using [43] as in [17]. In our experiment, we use grid search method and choose the set of hyper-parameters that gives us the optimal pixel-level prediction.

The cost of each feature type measures the computation time for an entire image. We note that this cost can be further reduced by efficient implementation of local features. The segmentation time is taken into account as an initial cost during the evaluation in order to have a fair comparison with existing methods. More details on cost computation can be found in the supplementary materials.

Policy learning features: We design three sets of features for $\phi(s_t, a_t)$. The first are computed from marginal distributions on all regions, consisting of the average entropy, the average entropy gap between previous marginal estimation and current marginals, two binary vectors of length 9 to indicate which feature set has been used and which unseen feature set will be extracted respectively, and one vector for the statistics of difference in current marginal probabilities of the top two predictions. The second are region features on active leaf nodes, including the normalized area of active regions in current image, the average entropy of active regions and the average entropy gap between previous and current prediction in active regions. The third layer features consist of the distribution of all regions in hierarchies and the distribution of active regions in hierarchies. More details on policy learning features can be found in the supplementary material.

5.2 Baseline Methods

We compare our approach to two types of baselines as below. We also report the state-of-the-art performances on three datasets.

- Non-anytime CRF-based methods using the full feature set: (1) A fully-connected CRF (DCRF) model [47] whose data term is learned on finest layer of segmentation trees; (2) A Hierarchical Inference Machine (HIM) implemented by following the algorithm in [26]; (3) A pixel-level dense CRF model with superpixel higher-order terms (H-DCRF) as in [48]. They prove to be strong baselines for scene labeling tasks.
- Three strong anytime baselines, including a Static-Myopic (S-M), a Random Selection (RS) and a static-myopic feature selection (F-SM) anytime model. The static-myopic method (S-M) learns a fixed sequence of actions by maximizing immediate rewards on training set (cf. $\{a_t^0\}_{t=1}^T$ in Sect. 4.1). The random selection method (RS) uses our action pool and randomly takes

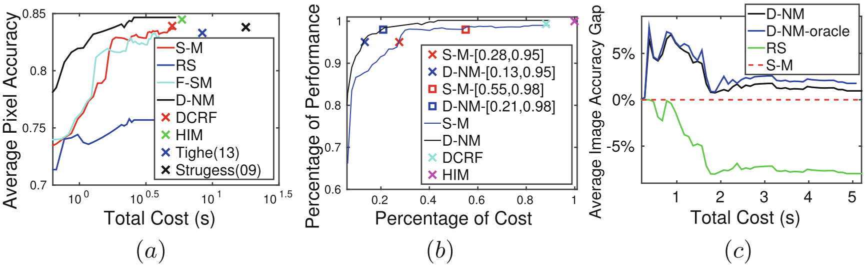

Fig. 3. (a): Average pixel accuracy vs. cost; (b): percentage of performance vs. percentage of cost; (c): average per-image accuracy gap vs. total cost in CamVid. Our D-NM consistently outperforms S-M in all figures and achieves full performance using about 50 % total cost. Moreover, it outperforms all anytime baselines consistently and achieves better performance w.r.t. non-anytime state-of-the-art.

an action at each step. The feature selection method (F-SM) uses the DCRF above as its model and greedily selects features that maximize the immediate rewards. We note that the baselines utilize some state-of-the-art feature selection methods such as [16,45], and our RS baseline is built on the learned high-quality action pool.

5.3 Results

We report the results of our experiments on anytime scene labeling in three parts: (1) overall comparison with the baselines and the state-of-the-art methods on CamVid. (2) detailed analysis of anytime property on CamVid. (3) results on Stanford Background and Siftflow datasets.

Overall performance on CamVid. We first show the quantitative results of our method and compare with state-of-the-art methods in Fig. 3.(a) and Table 1. We compute the accuracy and Intersection-Over-Union(IOU) score of semantic segmentation on CamVid. Note that here we report the performance of anytime methods at the time budget of T_{DCRF}, which is the average prediction time of DCRF. In Table 1, we can see that our method achieves better performance than DCRF in terms of per-pixel accuracy and IOU score, and DCRF is a strong baseline since it uses the full feature set. Our per-pixel accuracy is comparable to the HIM, which uses the most complex model and full feature set, while we achieve similar performance with about 50 % of its computation cost (See below for details). In addition, we outperform all the rest of state-of-the-art methods [17,49], especially in terms of average per-class accuracy (5.4 % to 9 % absolute gap). Moreover, we achieve similar or slightly better performance w.r.t. the methods that use additional information such as Structure-from-Motion(SfM) of video sequence [19,50] or pre-trained object detectors [51].

We conduct comparisons on the anytime performance of our methods and baselines in Fig. 3(a) and (b). We introduce a plot (b) showing all the performance and cost values w.r.t the HIM and its prediction cost since it is the state-of-the-art

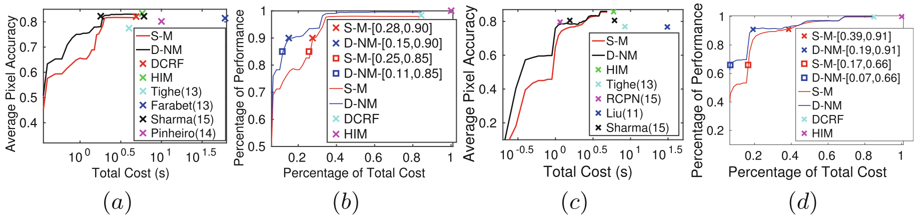

Fig. 4. Average pixel accuracy as a function of cost and the percentage performance vs. percentage of cost in SBG (a,b) and Siftflow (c,d), respectively. D-NM achieves similar performance with less cost. Cost of related work is from [52].

Table 1. Performance comparison on CamVid. D-NM outperforms [17,49], especially in average class accuracy. Our results are comparable to [19,50,51] that use additional information. We achieve a performance similar to HIM and DCRF with less cost.

CamVid	Tighe [49]	SIM [17]	Video [19]	Detector [51]	Video [50]	DCRF	H-DCRF	HIM	D-NM (ours)
Pixel	83.3	81.5	69.1	83.2	83.8	83.2	83.9	84.5	**84.7**
Class	51.2	54.8	53.0	59.6	59.2	59.8	60.0	**60.5**	60.2
IOU	NA	NA	NA	**49.3**	49.2	46.3	48.4	**49.3**	48.8

and most costly. Specifically, Fig. 3(b) shows the percentage of average pixel-level accuracy vs. percentage of total cost curves of our methods and baselines w.r.t the HIM. We note that this illustration is invariant to the specific values of prediction cost/time, and shows how the accuracy improves with increasing cost.

We first show comparison of our method with all the anytime and non-anytime baselines in Fig. 3(b), which also highlights two sets of intermediate results. Our dynamic policy D-NM achieves the 90 % of performance using only around 10 % cost and outperforms the S-M consistently. Specifically, D-NM achieves similar performance with around half S-M test-time cost (13 % and 21 % vs. 28 % and 55 %). Moreover, D-NM achieves the full performance of HIM with around 50 % total cost while S-M saturates at a lower accuracy. We refer the reader to the supplementary material for examples of our anytime output with specific actions.

Anytime property analysis on CamVid. We analyze the anytime property of our method by comparing to three different baselines. First, we validate the importance of encoding model complexity in anytime prediction model by comparing with F-SM (fixed model with feature selection). Second, we evaluate the effectiveness of policy learning by comparing with RS (random search on the same action space). Results of these two comparisons can be viewed in Fig. 3(a). Finally, we explore the effectiveness of action space exploration by generating the oracle results of D-NM on CamVid test set in Fig. 3(c).

Figure 3(a) shows that the D-NM outperforms all baselines consistently and generates superior results under the same cost. RS is almost always the worst and far below D-NM, which shows our policy learning is important and effective to achieve better trade-offs between accuracy and cost. F-SM is slightly above

S-M at the beginning and always below D-NM. Moreover, due to the limited representation power of its fixed model, F-SM quickly stabilizes at a lower performance. This demonstrates the benefits of joint feature and model selection in our method. We also visualize results of other methods (crossings) and show that we can achieve better performance more efficiently. These results evidence that our method can learn a better representation for anytime scene parsing. Detailed averaged IOU score and labeling loss as a function of cost, area-under-average-accuracy table can be viewed in supplementary materials.

Table 2. Semantic segmentation results on Stanford background dataset. We can achieve better performance w.r.t state-of-the-art methods.

SBG	RCPN [53]	Tighe [49]	Gould [20]	Farabet [9]	Pinheiro [54]	Sharma [52]	H-DCRF	S-M	D-NM (ours)
Pixel	81.8	77.5	76.4	81.4	80.2	82.3	82.6	81.7	**83.0**
IOU	61.3	NA	NA	NA	NA	64.5	**64.7**	61.4	**64.7**

Figure 3(c) shows the average per-image accuracy gap w.r.t the S-M method as a function of total cost. We note that D-NM always achieves superior performance to the S-M. We also visualize the oracle performance of D-NM. D-NM-oracle is always above S-M, which proves the effectiveness of action space exploration. Also, the early stop of oracles shows that more features or complex models will not introduce further segmentation improvement. Our D-NM is only slightly below D-NM-oracle, which shows the effectiveness of policy learning.

Table 3. Semantic segmentation results on Siftflow dataset. We can achieve comparable/better performance w.r.t. state-of-the-art methods.

Siftflow	RCPN	Yang	Pinheiro	Liu	Tighe	FCN	Farabet	Sharma	H-DCRF	S-M	D-NM
	[53]	[55]	[54]	[21]	[49]	[4]	[9]	[52]			(ours)
Pixel	79.6	79.8	77.7	76.7	77.0	85.7	78.5	80.8	**85.8**	**85.8**	**85.8**
IOU	26.9	NA	NA	NA	NA	**36.7**	NA	30.7	**36.7**	35.8	**36.7**

Stanford Background. Results on Stanford Background dataset [20] are shown in Table 2. D-NM outperforms existing work in terms of pixel-level accuracy and IOU score. We visualize the anytime property in top row of Fig. 4. Figure 4(a) shows that D-NM achieves the state-of-the-art performance (crossings) more efficiently while S-M stops at a lower performance. Figure 4(b) highlights two sets of intermediate results and shows that D-NM generates similar results with about half of the S-M cost (11 % and 15 % vs. 25 % and 28 %).

Siftflow. We report our results on Siftflow dataset [21] in Table 3. Again, D-NM achieves the state-of-the-art in terms of pixel level accuracy and IOU score. Figure 4(c) shows its anytime performance curves and Fig. 4(d) also highlights two sets of intermediate results. We can see that D-NM achieves the state-of-the-art performance (crossings) more efficiently, and produces similar accuracy with much less cost.

6 Conclusion

In this paper, we presented a dynamic hierarchical model for anytime semantic scene segmentation. Our anytime representation is built on a coarse-to-fine segmentation tree, which enables us to select both discriminative features and effective model structure for cost-sensitive scene labeling. We developed an MDP formulation and an approximated policy iteration method with an action proposal mechanism for learning the anytime representation. The results of applying our method to three semantic segmentation datasets show that our algorithm consistently outperforms the baseline approaches and the state-of-the-arts. This suggests that our learned dynamic non-myopic policy generates a more effective representation for anytime scene labeling.

Acknowledgement. DATA61 (formerly NICTA) is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre for Excellence Program. We thank NVIDIA Corporation for the donation of GPUs used in this research.

References

1. Hegdé, J.: Time course of visual perception: coarse-to-fine processing and beyond. Progress Neurobiol. (2008)
2. Karayev, S., Fritz, M., Darrell, T.: Anytime recognition of objects and scenes. In: CVPR (2014)
3. Gould, S., He, X.: Scene understanding by labeling pixels. CACM (2014)
4. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
5. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: CVPR (2015)
6. Dai, J., He, K., Sun, J.: Convolutional feature masking for joint object and stuff segmentation. In: CVPR (2015)
7. He, X., Zemel, R.S., Carreira-Perpiñán, M.: Multiscale conditional random fields for image labeling. In: CVPR (2004)
8. Yao, J., Fidler, S., Urtasun, R.: Describing the scene as a whole: joint object detection, scene classification and semantic segmentation. In: CVPR (2012)
9. Farabet, C., Couprie, C., Najman, L., LeCun, Y.: Learning hierarchical features for scene labeling. In: PAMI (2013)
10. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected CRFs. In: ICLR (2015)
11. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)

12. Roig, G., Boix, X., Nijs, R.D., Ramos, S., Kuhnlenz, K., Gool, L.V.: Active MAP inference in CRFs for efficient semantic segmentation. In: ICCV (2013)
13. Weiss, D., Taskar, B.: Learning adaptive value of information for structured prediction. In: NIPS (2013)
14. Zilberstein, S.: Using anytime algorithms in intelligent systems. AI Magazine (1996)
15. Xu, Z., Kusner, M., Huang, G., Weinberger, K.Q.: Anytime representation learning. In: ICML (2013)
16. Grubb, A., Bagnell, D.: SpeedBoost: anytime prediction with uniform near-optimality. In: AISTATS (2012)
17. Grubb, A., Munoz, D., Bagnell, J.A., Hebert, M.: SpeedMachines: anytime structured prediction. In: Learning with Test-time Budgets Workshop on ICML (2013)
18. Lagoudakis, M.G., Parr, R.: Least-squares policy iteration. In: JMLR (2003)
19. Brostow, G.J., Shotton, J., Fauqueur, J., Cipolla, R.: Segmentation and recognition using structure from motion point clouds. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5302, pp. 44–57. Springer, Heidelberg (2008). doi:10.1007/978-3-540-88682-2_5
20. Gould, S., Fulton, R., Koller, D.: Decomposing a scene into geometric and semantically consistent regions. In: ICCV (2009)
21. Liu, C., Yuen, J., Torralba, A.: Nonparametric scene parsing via label transfer. PAMI (2011)
22. Slorkey, A., Williams, C.K.: Image modeling with position-encoding dynamic trees. PAMI (2003)
23. Socher, R., Lin, C.C., Manning, C., Ng, A.Y.: Parsing natural scenes and natural language with recursive neural networks. In: ICML (2011)
24. Lempitsky, V., Vedaldi, A., Zisserman, A.: Pylon model for semantic segmentation. In: NIPS (2011)
25. Russell, C., Kohli, P., Torr, P.H., et al.: Associative hierarchical crfs for object class image segmentation. In: ICCV (2009)
26. Munoz, D., Bagnell, J.A., Hebert, M.: Stacked hierarchical labeling. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6316, pp. 57–70. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15567-3_5
27. Zhu, S.C., Mumford, D.: A Stochastic Grammar of Images. Now Publishers Inc., Hanover (2007)
28. Liu, B., He, X.: Multiclass semantic video segmentation with object-level active inference. In: CVPR (2015)
29. Riemenschneider, H., Bódis-Szomorú, A., Weissenberg, J., Gool, L.: Learning where to classify in multi-view semantic segmentation. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 516–532. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10602-1_34
30. He, H., Daumé III., H., Eisner, J.: Dynamic feature selection for dependency parsing. In: EMNLP (2013)
31. Wang, J., Bolukbasi, T., Trapeznikov, K., Saligrama, V.: Model selection by linear programming. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8690, pp. 647–662. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10605-2_42
32. Trapeznikov, K., Saligrama, V.: Supervised sequential classification under budget constraints. In: AISTATS (2013)
33. Sochman, J., Matas, J.: Waldboost-learning for time constrained sequential detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005). IEEE (2005)

34. Weiss, D., Sapp, B., Taskar, B.: Dynamic structured model selection. In: ICCV (2013)
35. Benbouzid, D., Busa-Fekete, R., Kégl, B.: Fast classification using sparse decision DAGs. In: ICML (2012)
36. Jiang, J., Moon, T., Daumé III., H., Eisner, J.: Prioritized asynchronous belief propagation. In: Inferning Workshop on ICML (2013)
37. Wu, T., Zhu, S.C.: Learning near-optimal cost-sensitive decision policy for object detection. In: ICCV (2013)
38. Amer, M.R., Xie, D., Zhao, M., Todorovic, S., Zhu, S.-C.: Cost-sensitive top-down/bottom-up inference for multiscale activity recognition. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7575, pp. 187–200. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33765-9_14
39. Ross, S., Munoz, D., Hebert, M., Bagnell, J.A.: Learning message-passing inference machines for structured prediction. In: CVPR (2011)
40. Denoyer, L., Gallinari, P.: Deep sequential neural network. In: Workshop Deep Learning NIPS (2014)
41. Doppa, J.R., Fern, A., Tadepalli, P.: Structured prediction via output space search. JMLR (2014)
42. Zhang, Y., Lei, T., Barzilay, R., Jaakkola, T.: Greed is good if randomized: new inference for dependency parsing. In: EMNLP (2014)
43. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV (2004)
44. Grundmann, M., Kwatra, V., Han, M., Essa, I.: Efficient hierarchical graph-based video segmentation. In: CVPR (2010)
45. Xu, Z., Weinberger, K., Chapelle, O.: The greedy miser: learning under test-time budgets. In: ICML (2012)
46. Gould, S.: DARWIN: A framework for machine learning and computer vision research and development. JMLR (2012)
47. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected CRFs with gaussian edge potentials. In: NIPS (2011)
48. Vineet, V., Warrell, J., Torr, P.H.: Filter-based mean-field inference for random fields with higher-order terms and product label-spaces. Int. J. Comput. Vis. (2014)
49. Tighe, J., Lazebnik, S.: Superparsing - scalable nonparametric image parsing with superpixels. IJCV (2013)
50. Sturgess, P., Alahari, K.: Combining appearance and structure from motion features for road scene understanding. In: BMVC (2009)
51. Floros, G., Rematas, K., Leibe, B.: Multi-class image labeling with top-down segmentation and generalized robust p̂n potentials. In: BMVC (2011)
52. Sharma, A., Tuzel, O., Jacobs, D.W.: Deep hierarchical parsing for semantic segmentation. In: CVPR (2015)
53. Sharma, A., Tuzel, O., Liu, M.Y.: Recursive context propagation network for semantic scene labeling. In: NIPS (2014)
54. Pinheiro, P., Collobert, R.: Recurrent convolutional neural networks for scene labeling. In: ICML (2014)
55. Yang, J., Price, B., Cohen, S., Yang, M.H.: Context driven scene parsing with attention to rare classes. In: CVPR (2014)