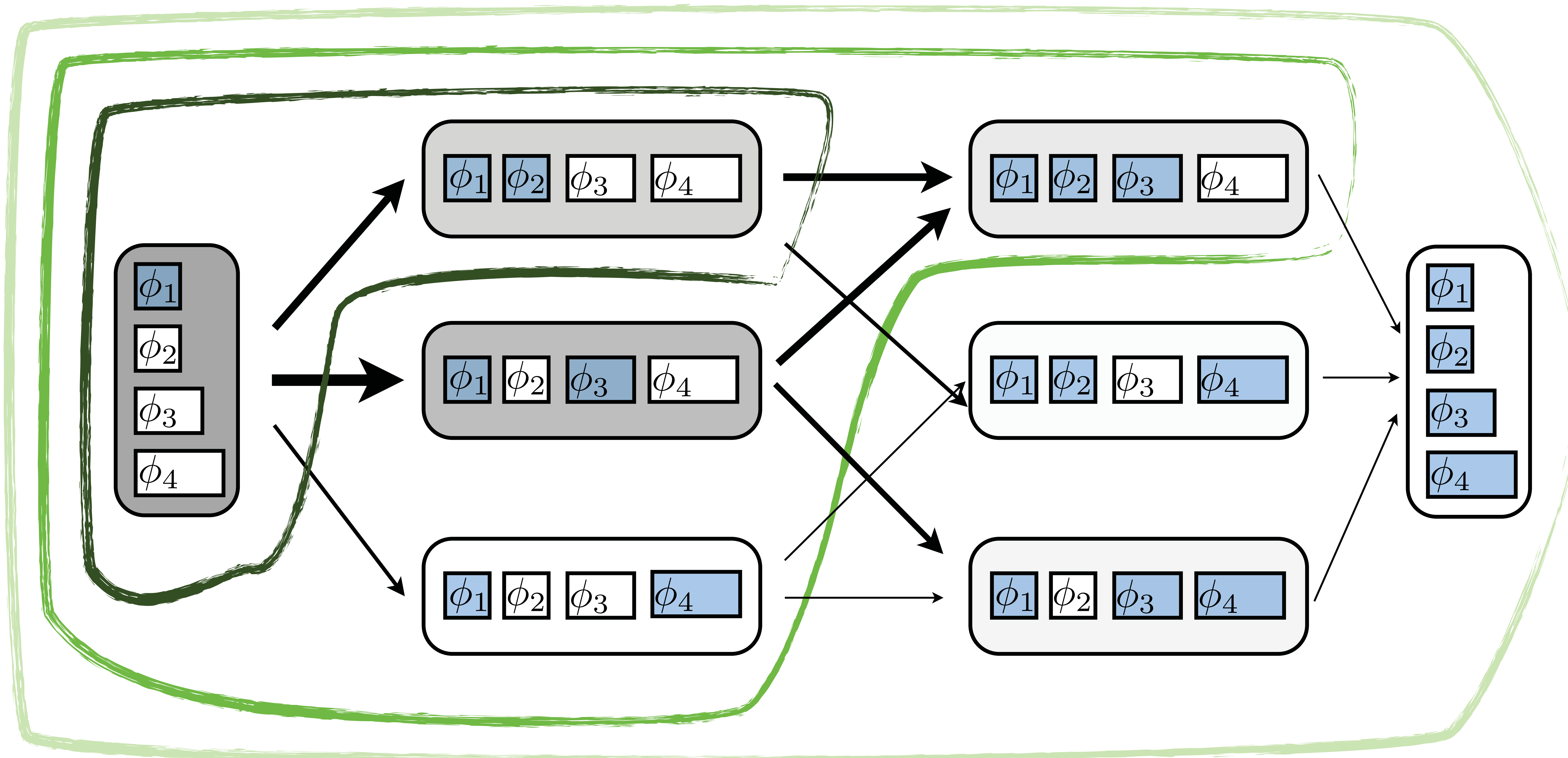
Dynamic Recognition on a Budget



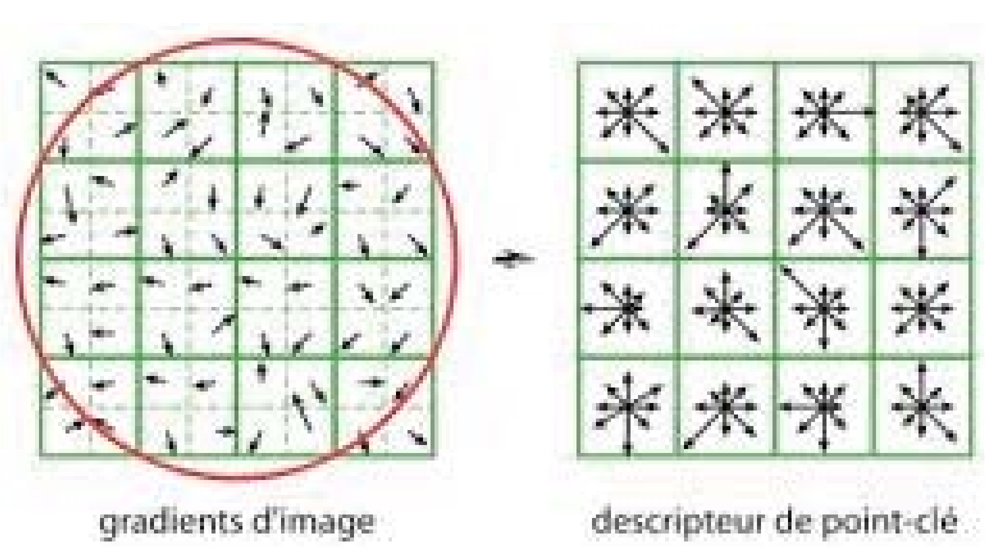
Sergey Karayev

Mario Fritz · Trevor Darrell

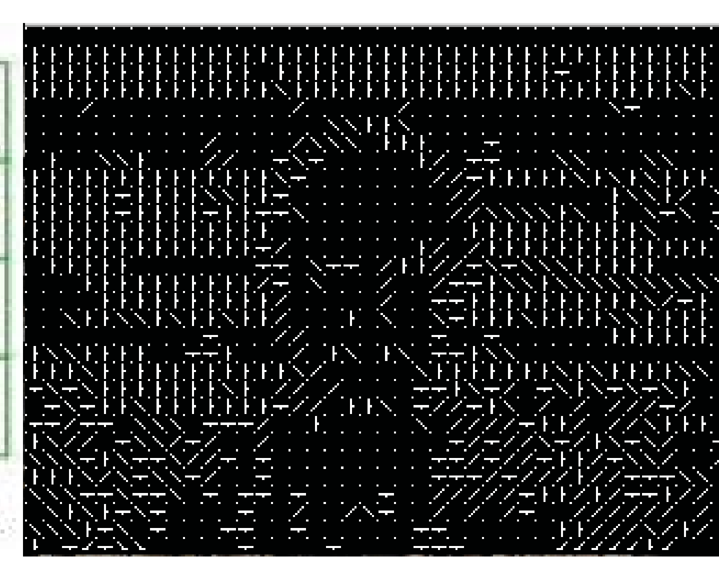
I work in computer vision.

Logo courtesy of Jon Barron.

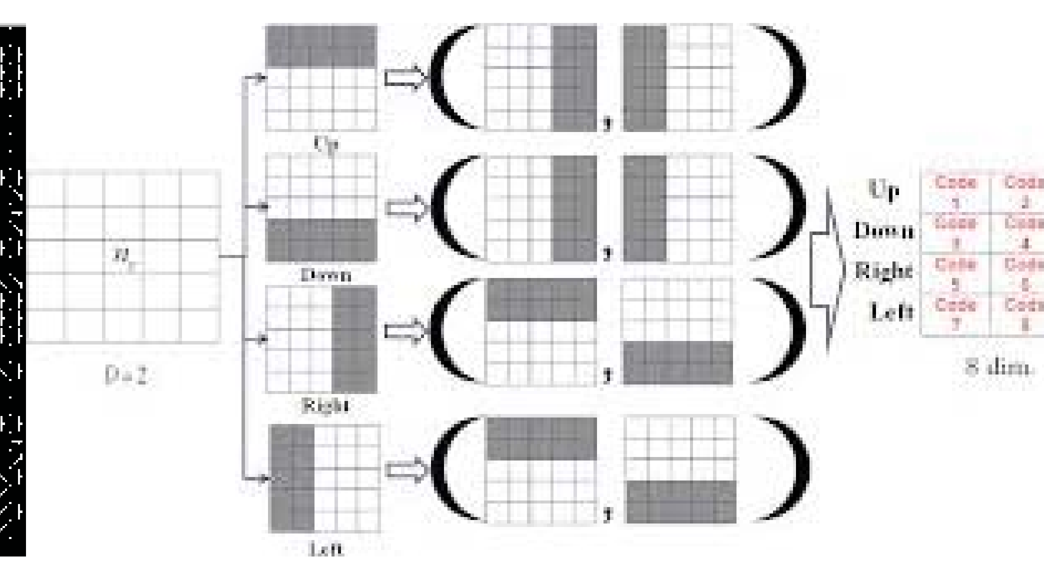
With a test time budget, cannot compute all features.



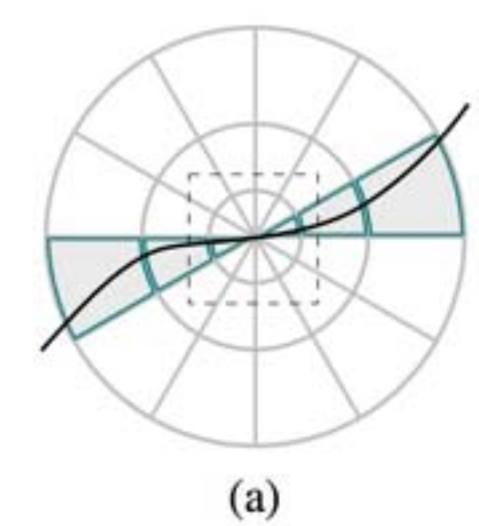
SIFT



HOG

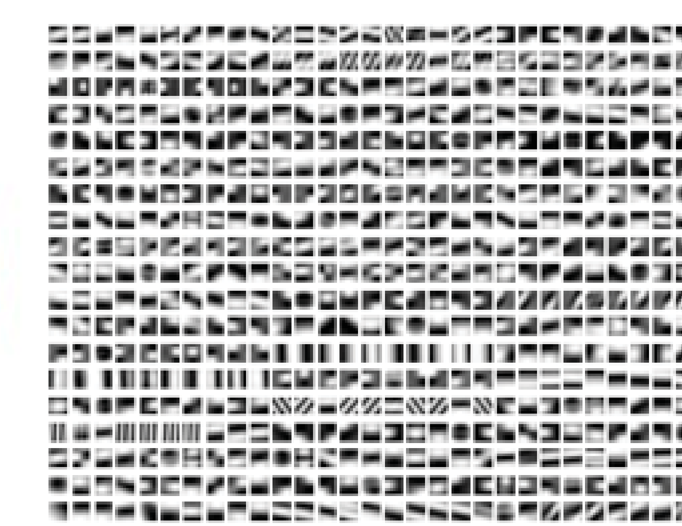


LBP



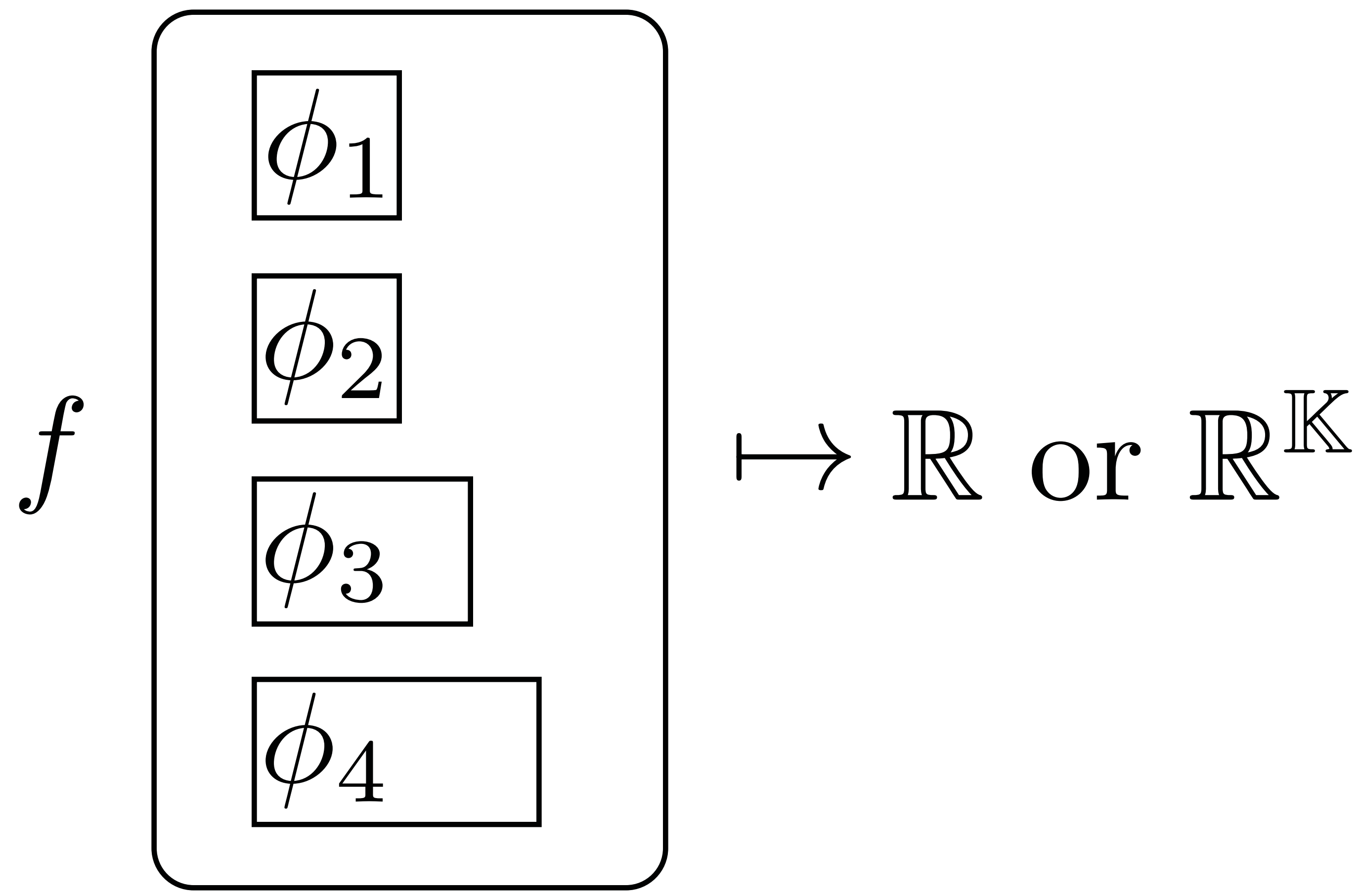
SSIM

color

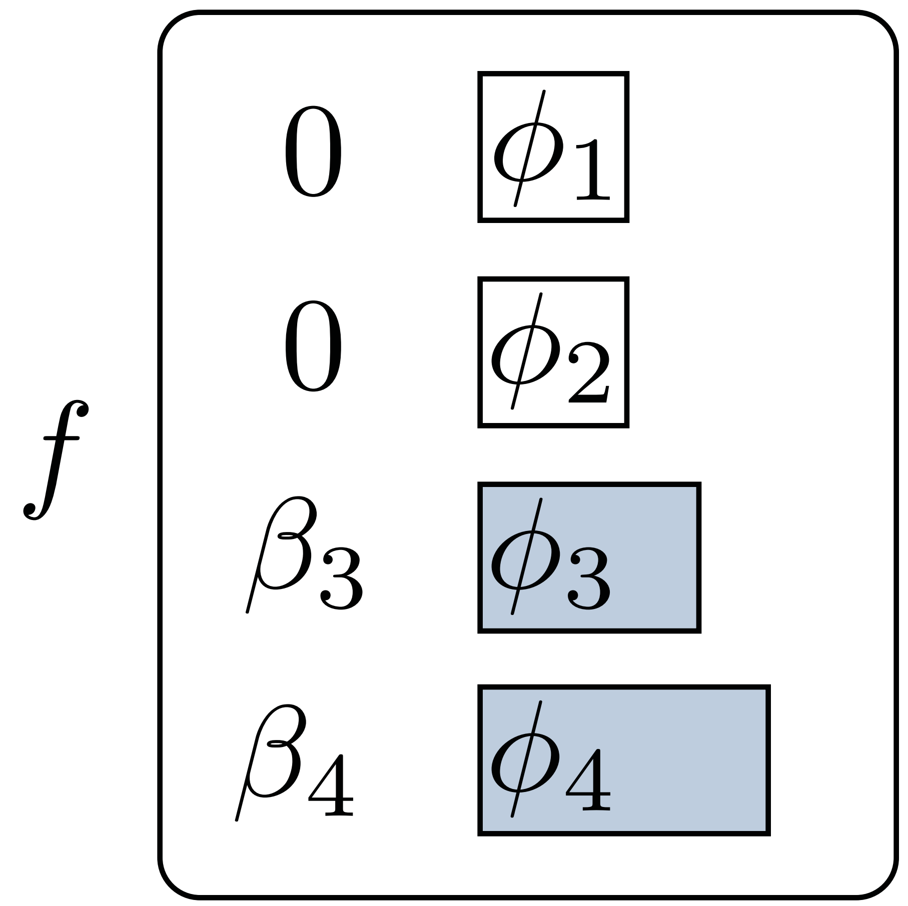


Textons

With different instances benefiting from different features, selection needs to be dynamic.


f
ϕ_1
ϕ_2
ϕ_3
ϕ_4
$\mapsto \mathbb{R}$ or $\mathbb{R}^{\mathbb{K}}$

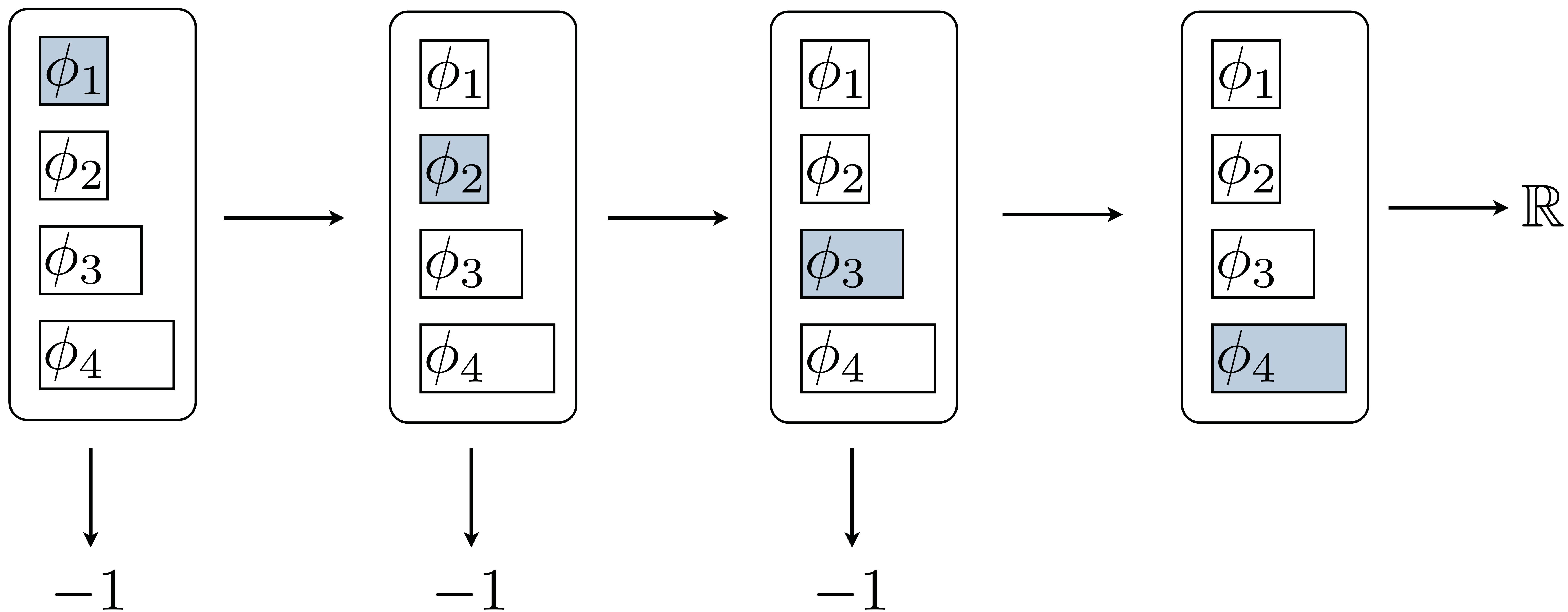
Feature selection



- Insufficient when budget is Anytime ,
  or when budgeted across multiple instances.

Cascade

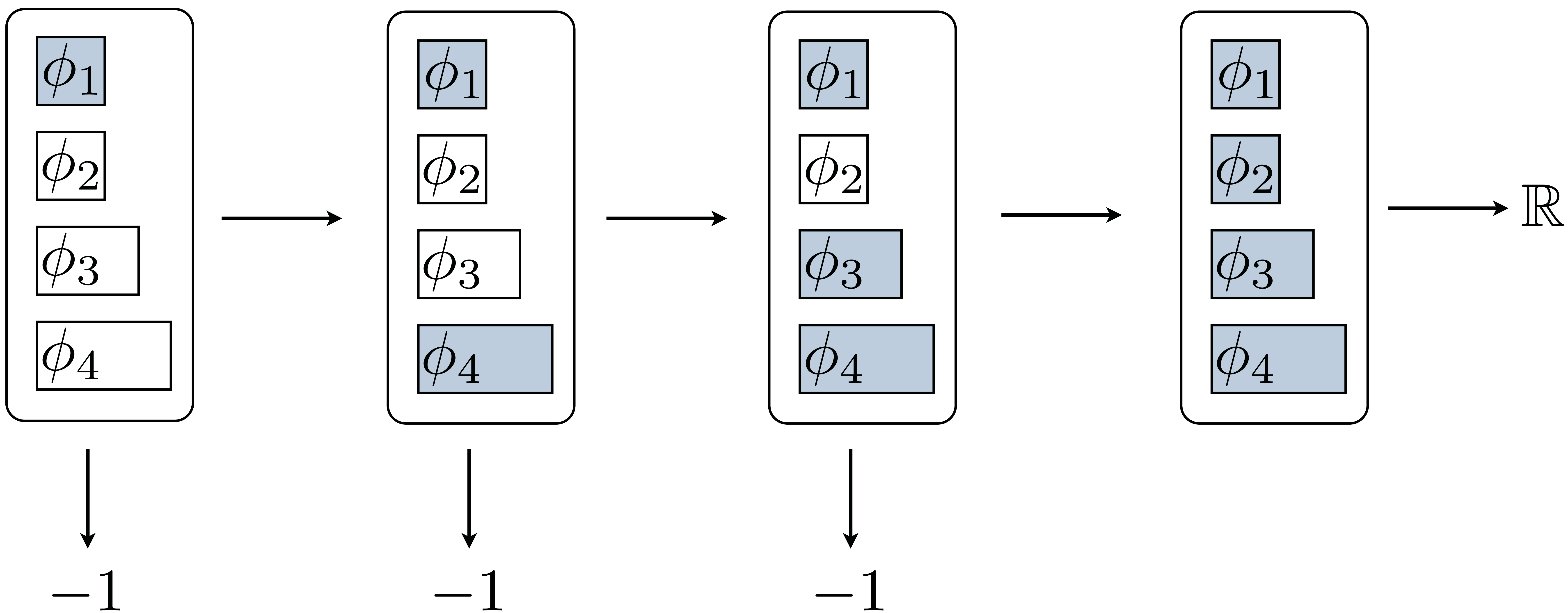
Viola & Jones (CVPR 2001)



- Two actions: Reject and Continue.

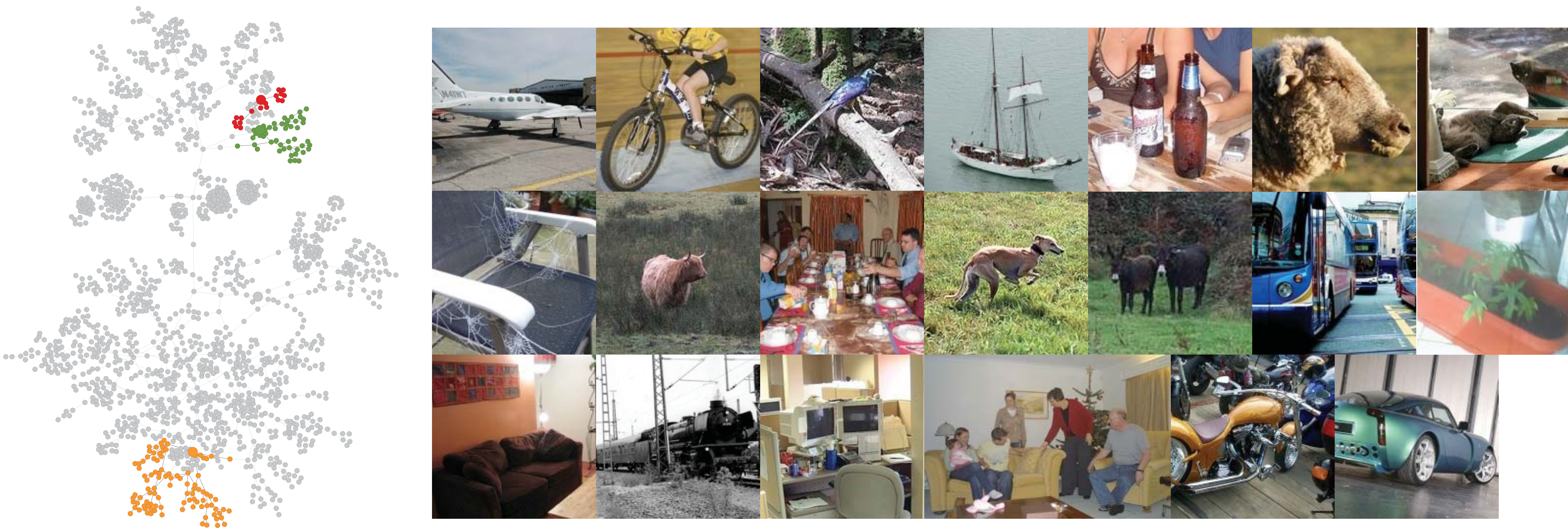
Minimizing feature cost

Chen et al. (ICML 2012), and others.

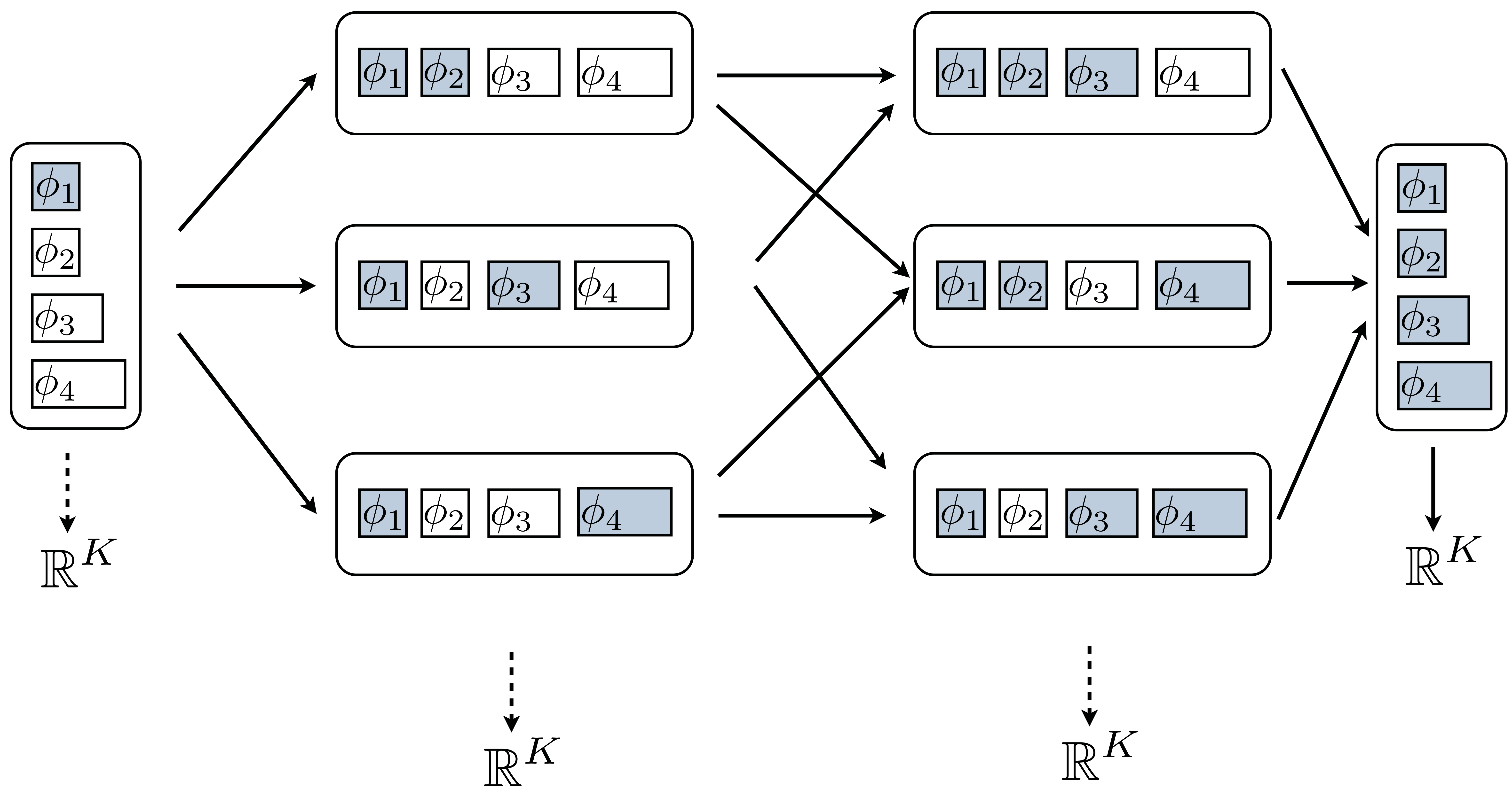


- Optimize both stage thresholds and order by considering feature computation cost.

$$\mathcal{L}(f) = \underbrace{\ell(f) + \rho r(f)}_{\text{regularized risk}} + \underbrace{\lambda c(f)}_{\text{test cost}}$$



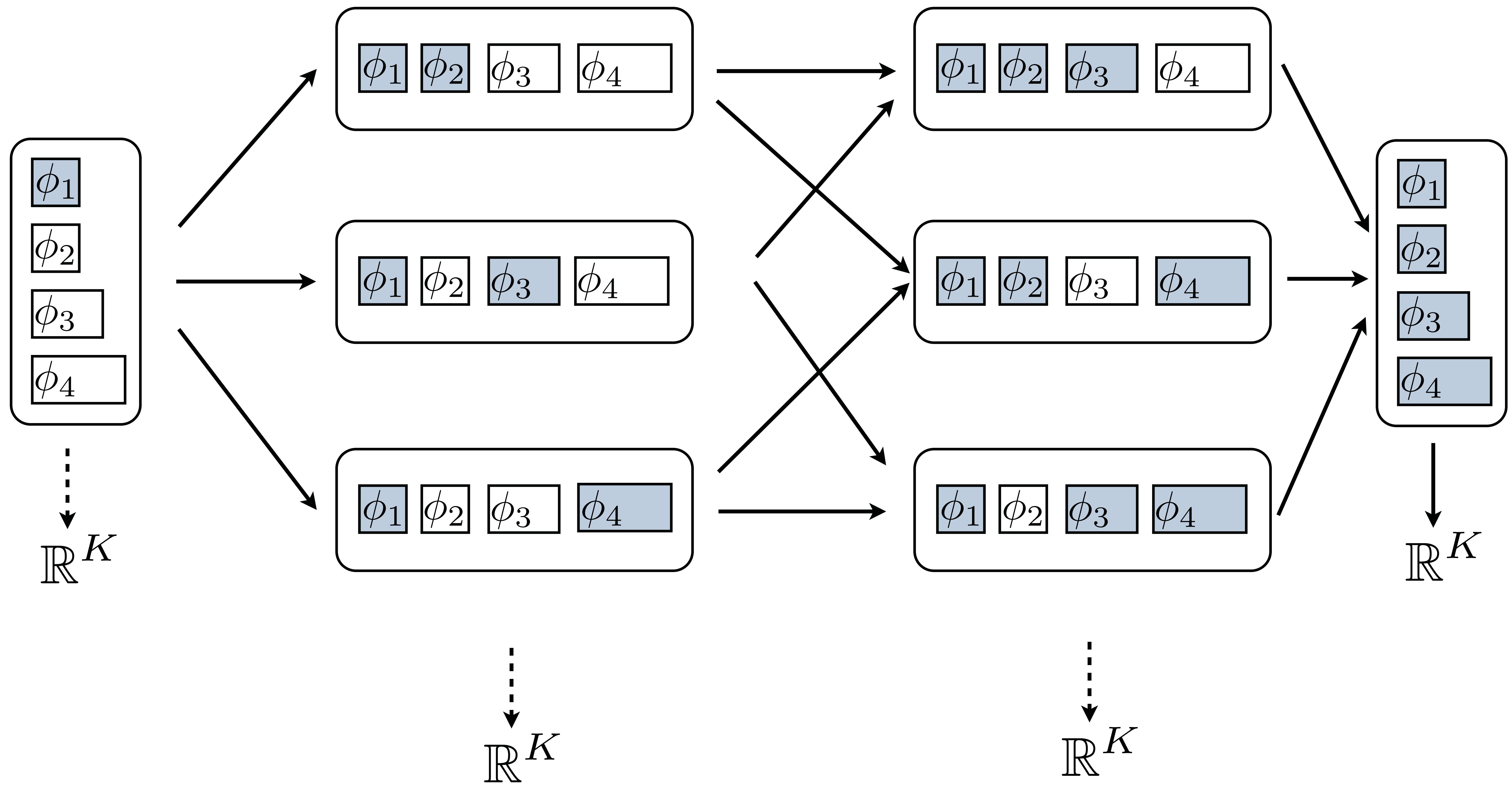
Dynamic Feature Selection



- Branch on selected feature values.
- Be able to classify from any state.

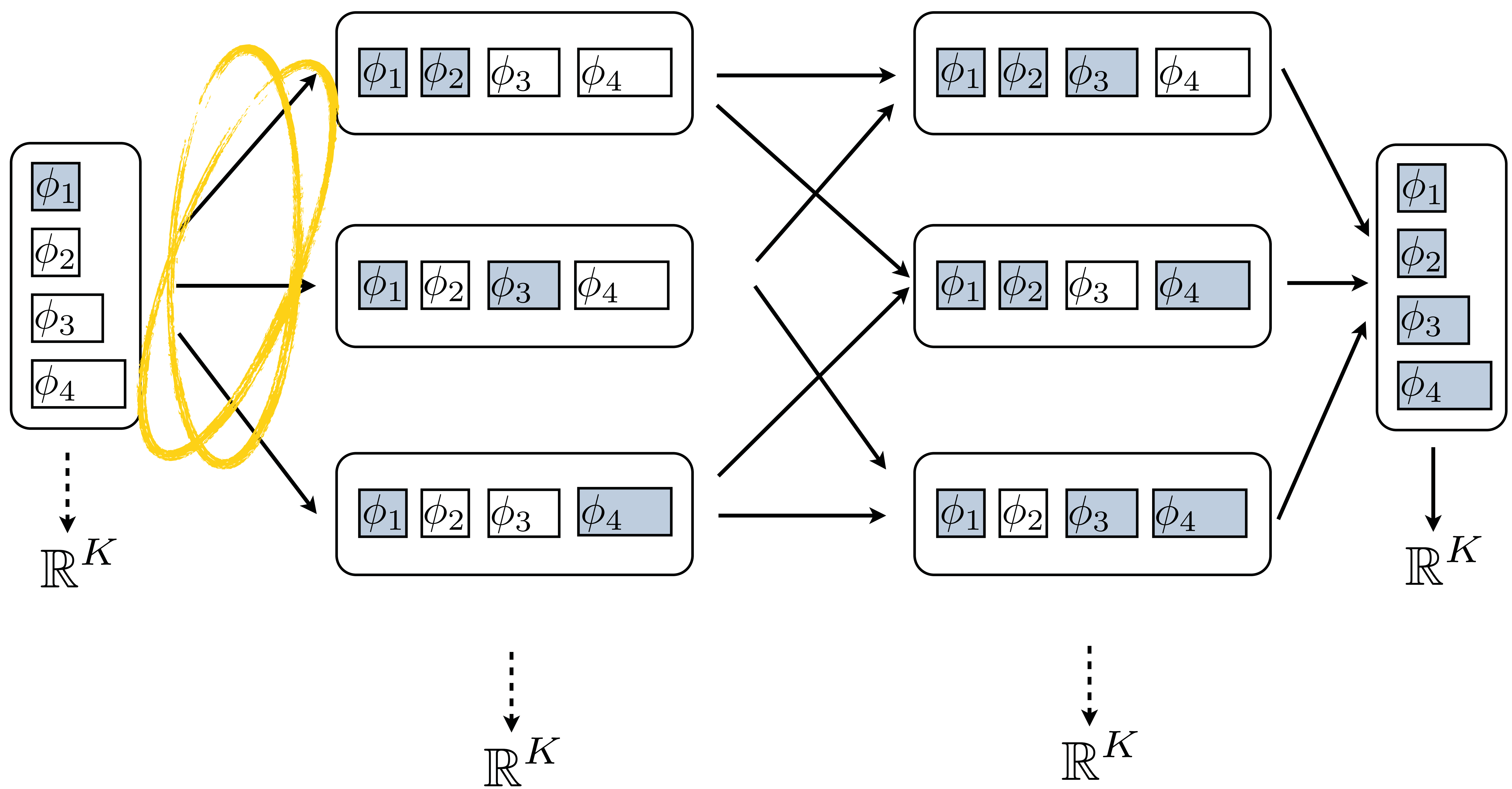
Active Classification

Gao & Koller (NIPS 2011)



Active Classification

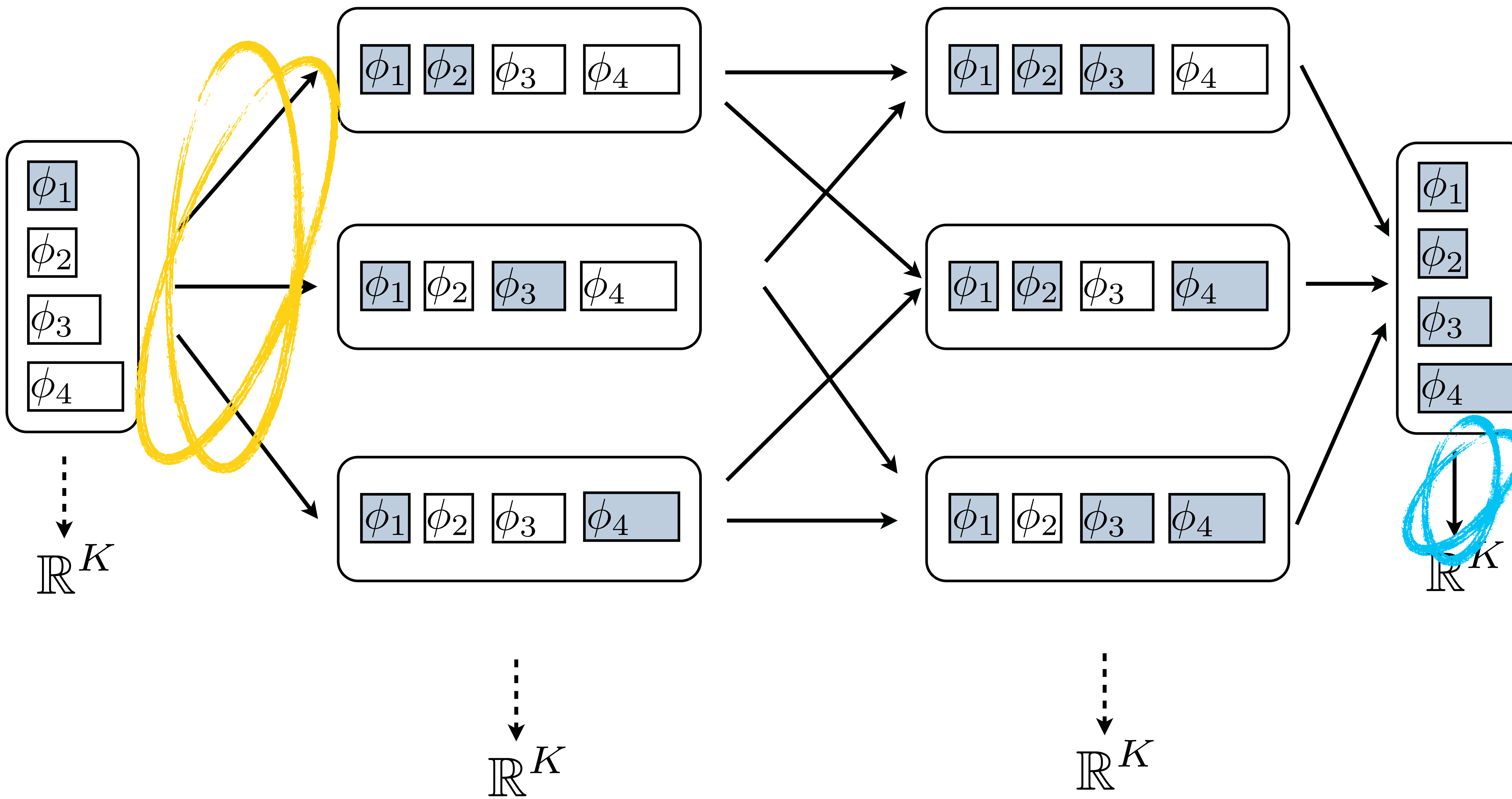
Gao & Koller (NIPS 2011)



- Action selection: greedy, based on expected information gain in mixture of Gaussians model (predicted by nearest neighbors in dataset).

Active Classification

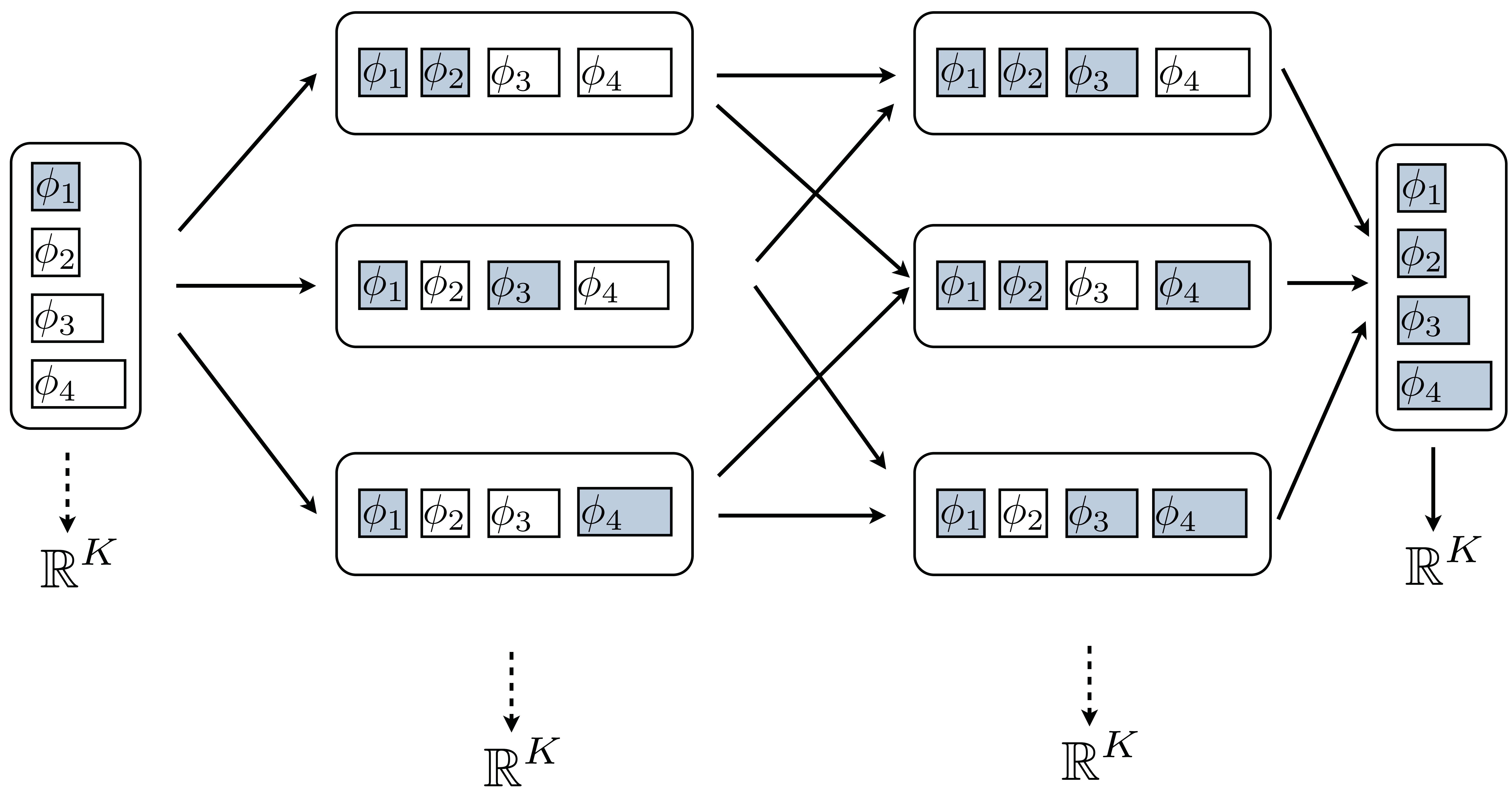
Gao & Koller (NIPS 2011)



- Action selection: greedy, based on expected information gain in mixture of Gaussians model (predicted by nearest neighbors in dataset).
- Feature combination: same mixture of Gaussians model.

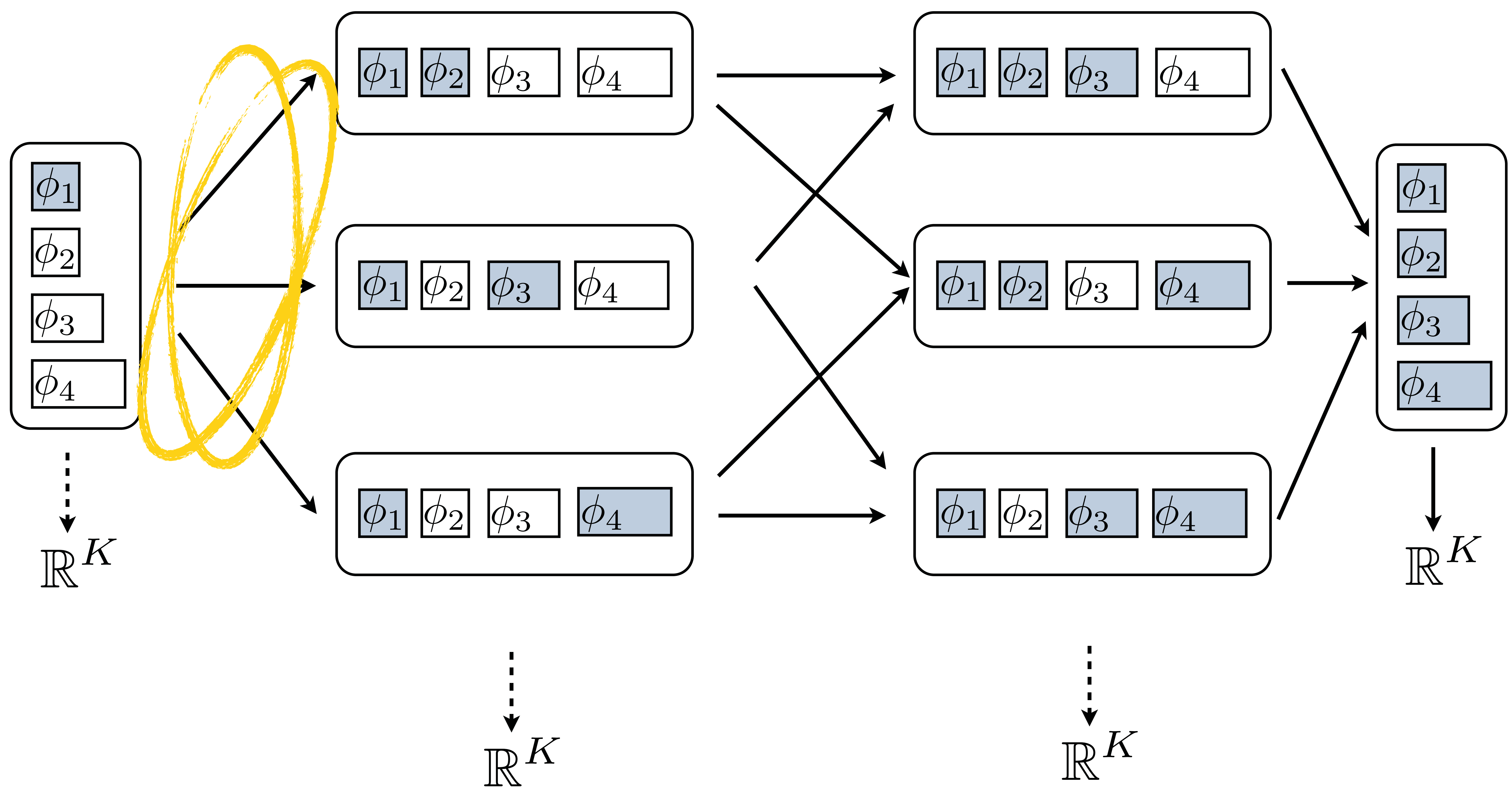
Timely Object Recognition

Karayev et al. (NIPS 2012)



Timely Object Recognition

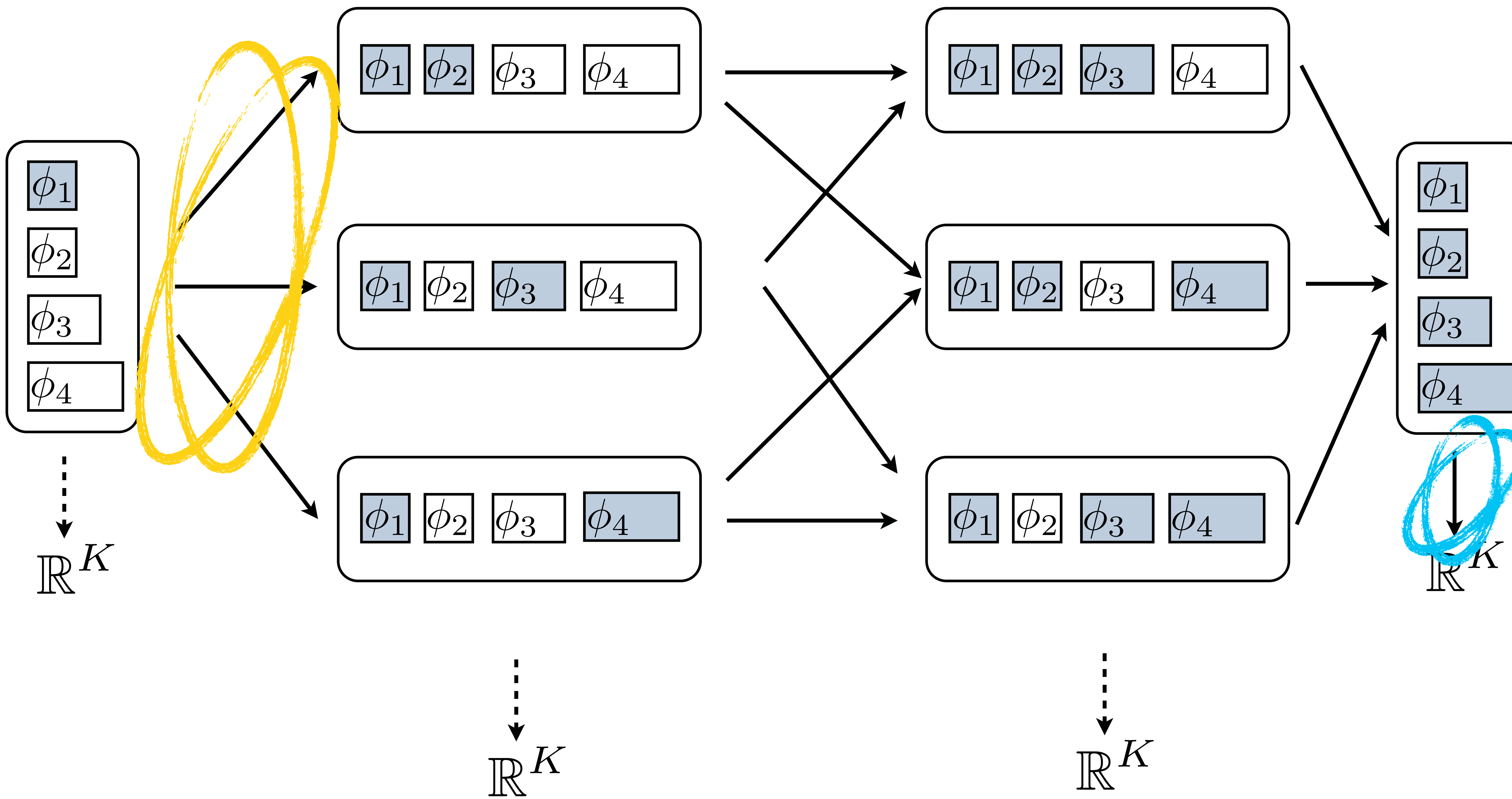
Karayev et al. (NIPS 2012)



- Action selection: non-myopic linear policy learned by MDP, with manually defined reward.

Timely Object Recognition

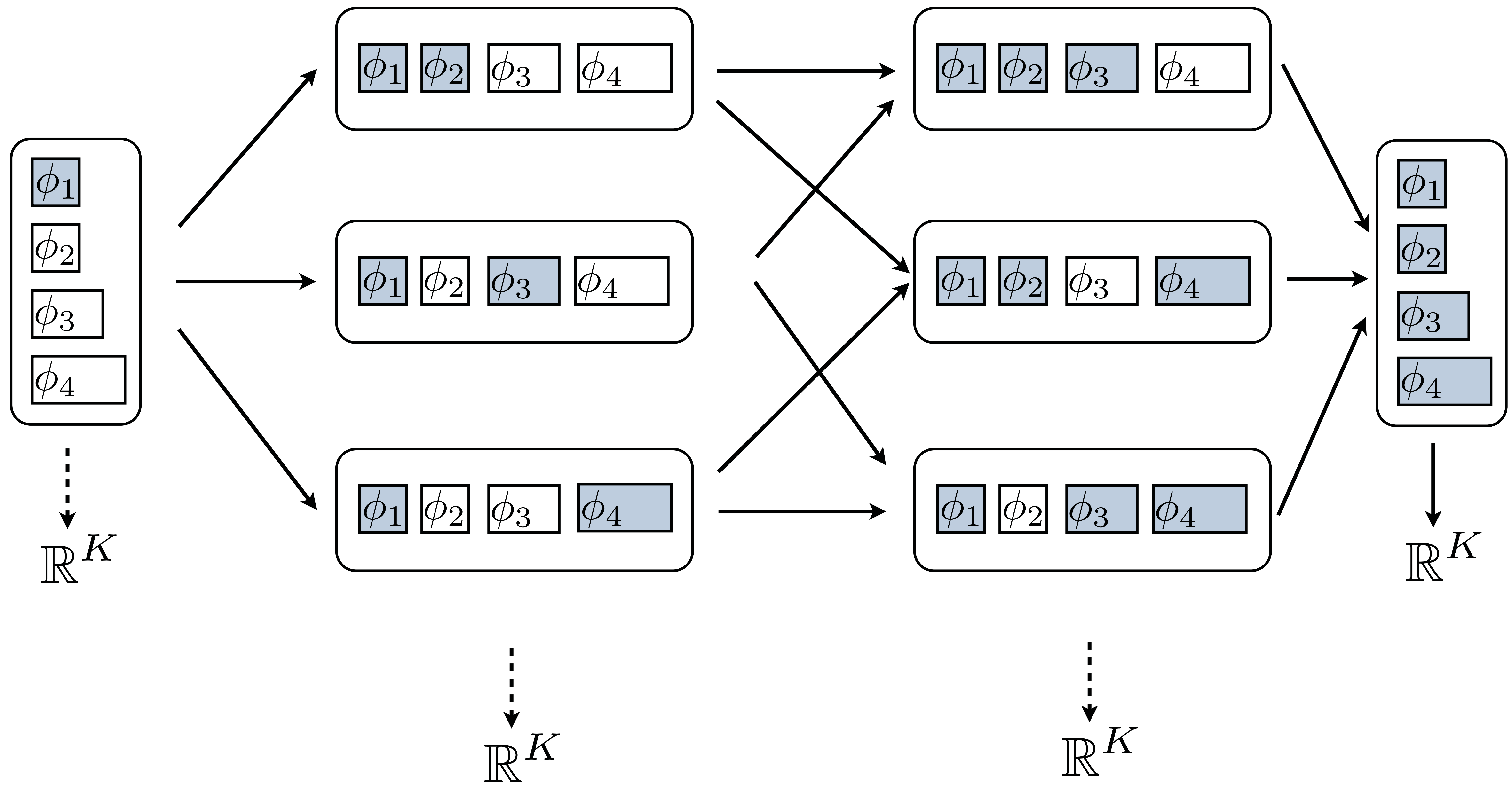
Karayev et al. (NIPS 2012)



- Action selection: non-myopic linear policy learned by MDP, with manually defined reward.
- Feature combination: inference in a graphical model (but isn't the point).

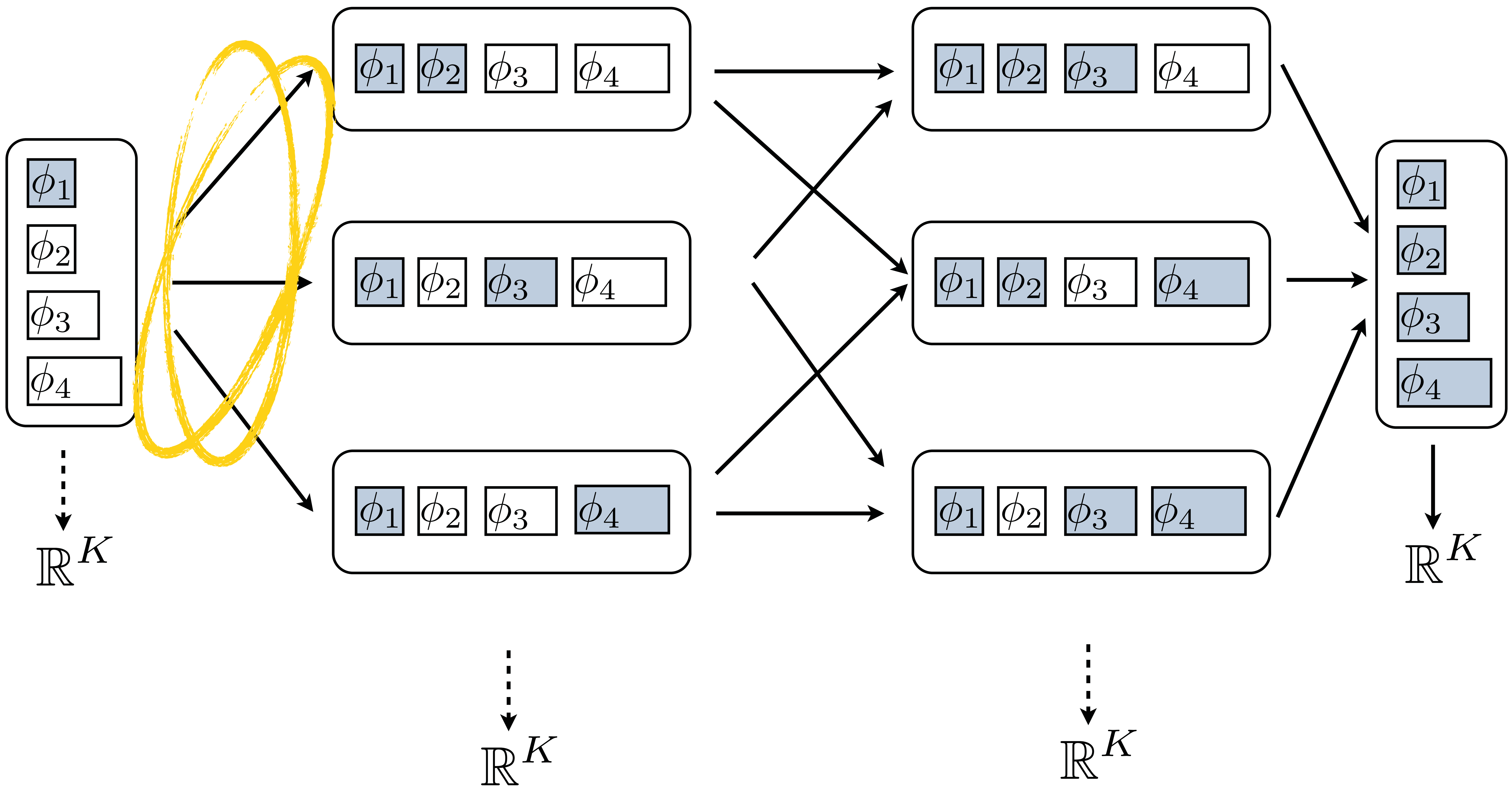
Dynamic Feature Selection for Classification

Karayev et al. (current)



Dynamic Feature Selection for Classification

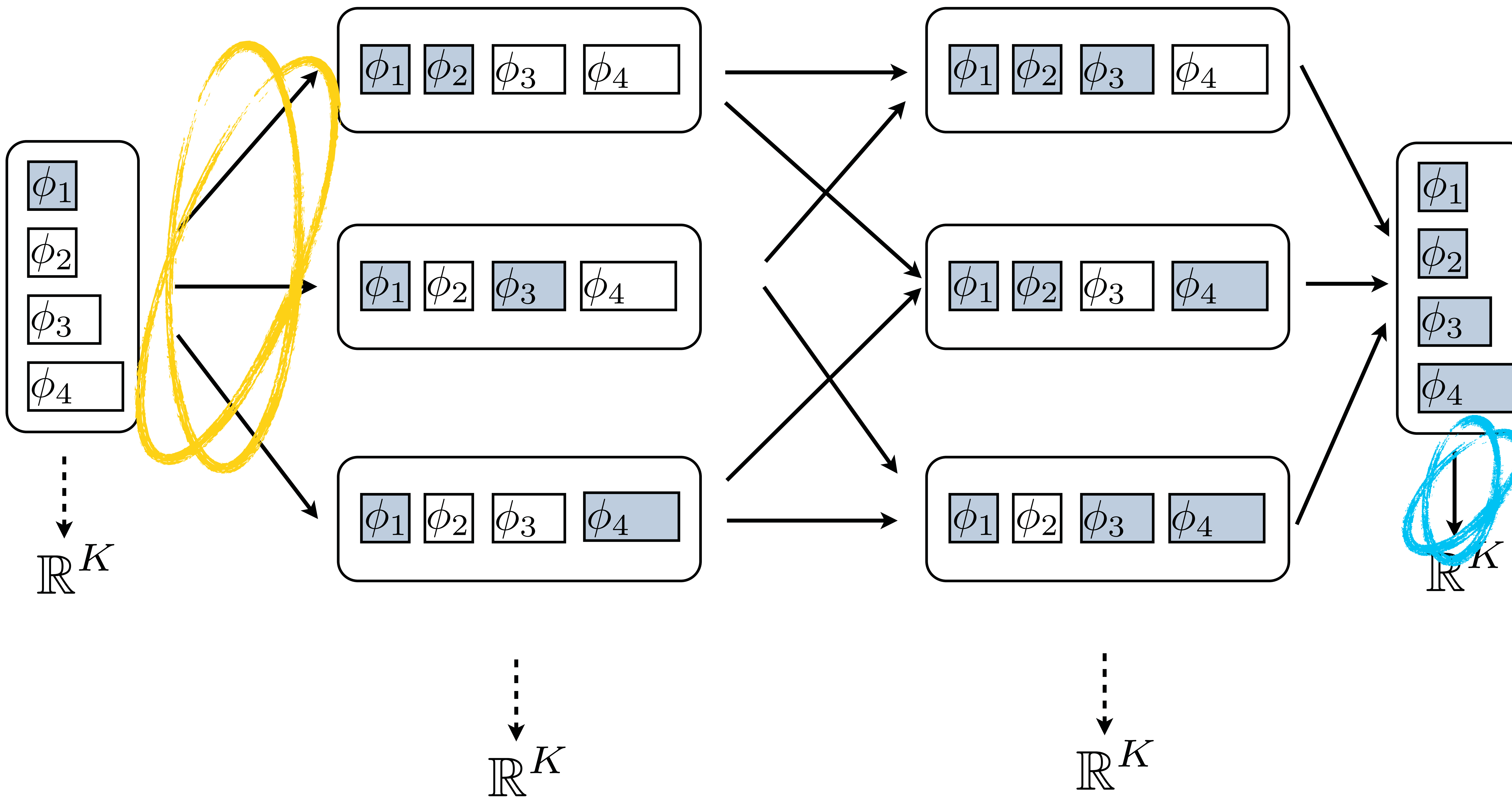
Karayev et al. (current)



- Action selection: non-myopic policy learned by MDP.

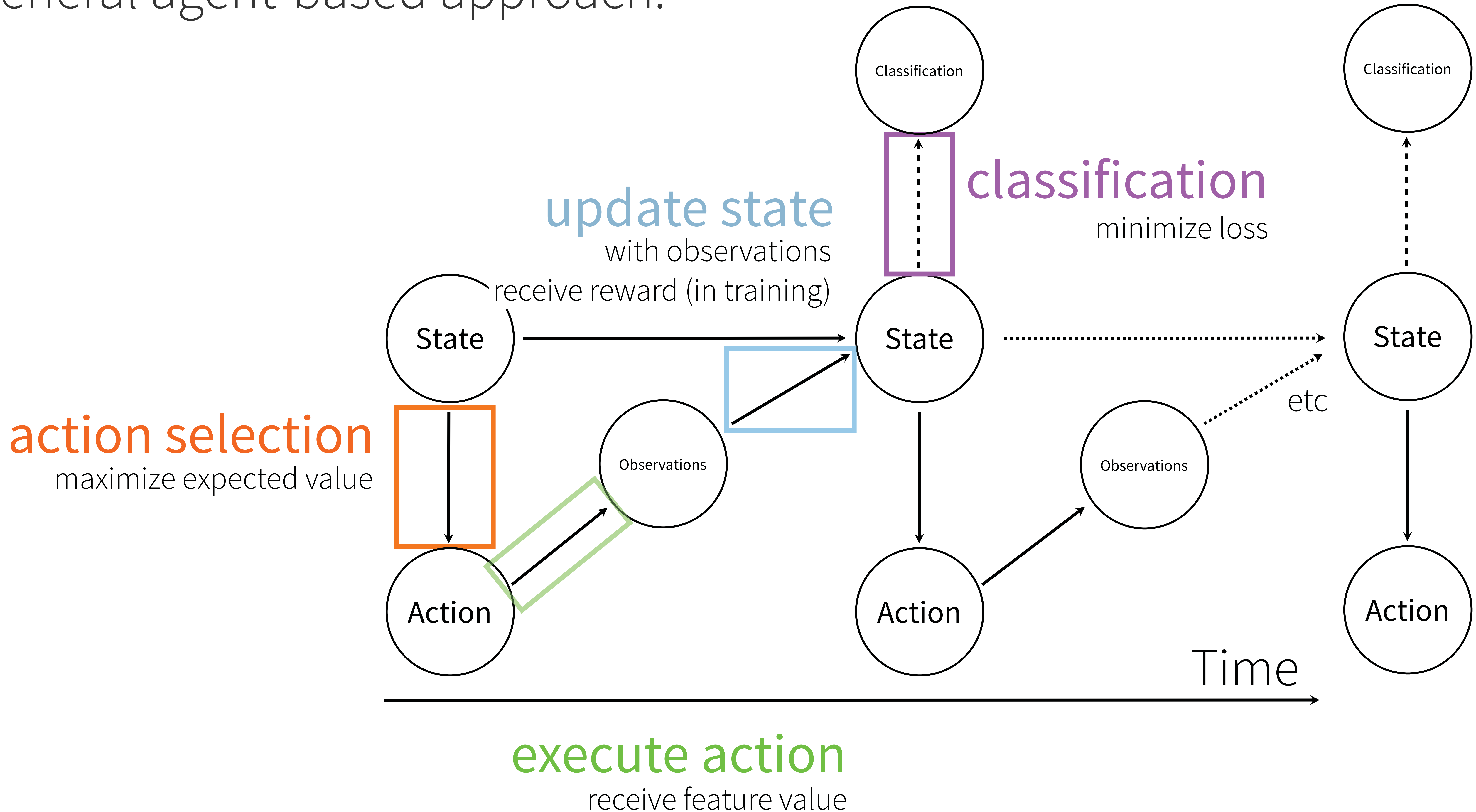
Dynamic Feature Selection for Classification

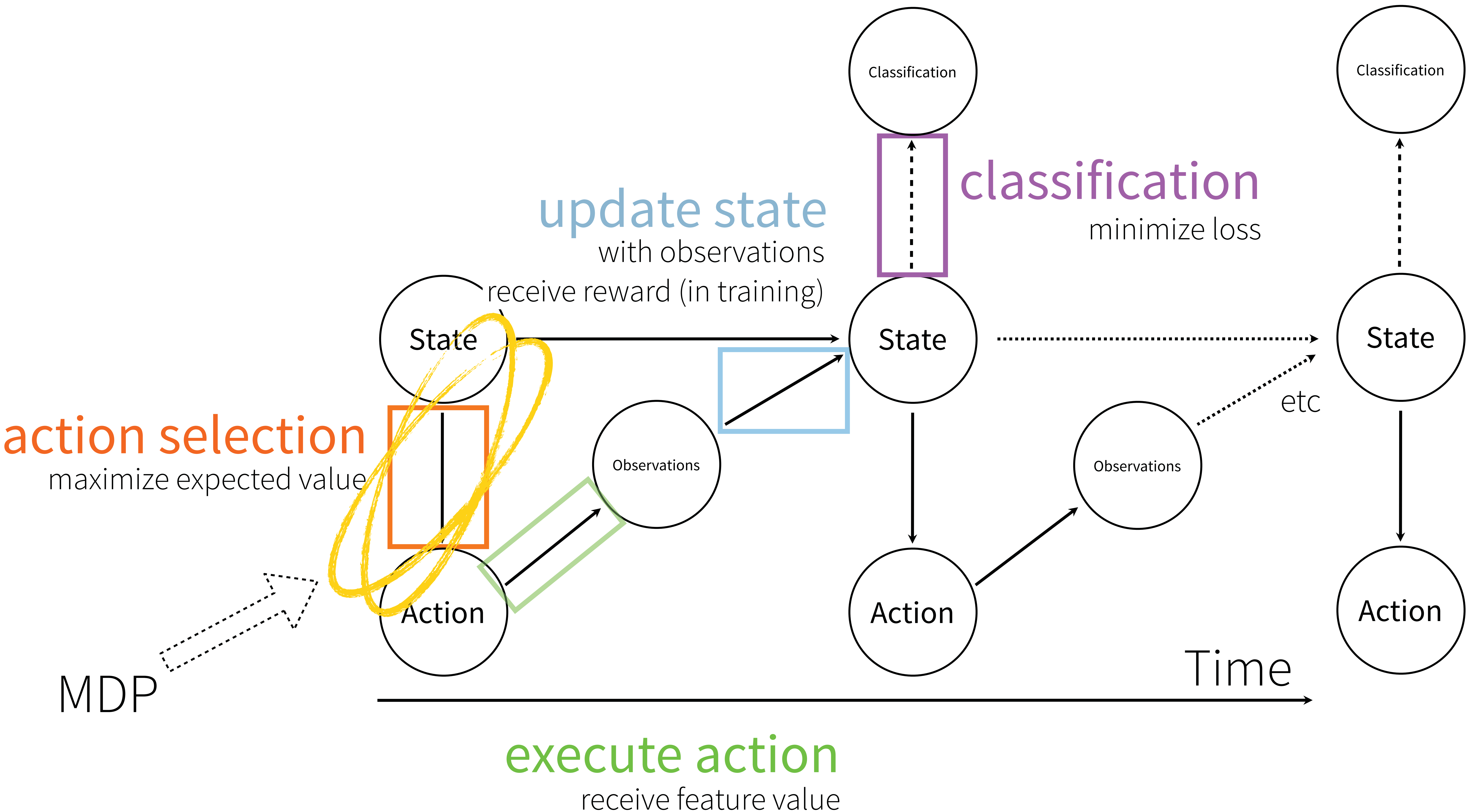
Karayev et al. (current)



- Action selection: non-myopic policy learned by MDP.
- Feature combination: linear.

General agent-based approach.





Classification
Classification
update state
with observations
receive reward (in training)
classification
minimize loss
State
State
State
action selection
maximize expected value
Observations
Observations
etc
Action
Action
Action
MDP
Time
execute action
receive feature value

Definition 2. *The **feature selection MDP** consists of the tuple $(\mathcal{S}, \mathcal{A}, T(\cdot), R(\cdot), \gamma)$:*

- ***State** $s \in \mathcal{S}$ stores the selected feature subset $\mathcal{H}_{\pi(x)}$ and their values and total cost $C_{\mathcal{H}_{\pi(x)}}$.*
- *The set of **actions** $\mathcal{A}$ is exactly the set of features $\mathcal{H}$.*
- *The (stochastic) **state transition** distribution $T(s' \mid s, a)$ can depend on the instance x.*
- *The **reward** function $R(s, a, s') \mapsto \mathbb{R}$ is manually specified, and depends on the classifier g and the instance x.*
- *The discount γ determines amount of **lookahead** in selecting actions: if 0, actions are selected greedily based on their immediate reward; if 1, the reward accrued by subsequent actions is given just as much weight as the reward of the current action.*

$$V_\pi(s_0) = \mathbb{E}_{\xi \sim \{\pi, x\}} r(\xi) = \mathbb{E}_{\xi \sim \{\pi, x\}} \left[\sum_{i=0}^{I} \gamma^i \, r_i \right]$$

action selection

maximize expected value

policy: $$\pi(s) = \arg\max_{a_i \in \mathcal{A} \setminus \mathcal{O}} Q(s, a_i)$$

action-value function: $$Q^\pi(s, a_i) = \mathbb{E}_{s'}[R(s', a_i) + \gamma Q^\pi(s', \pi(s'))]$$

reward definition

assume linearity: $$Q^\pi(s, a_i) = \theta_\pi^\top \phi(s, a_i)$$

learning the policy

action selection

maximize expected value

policy: $$\pi(s) = \arg\max_{a_i \in \mathcal{A} \backslash \mathcal{O}} Q(s, a_i)$$

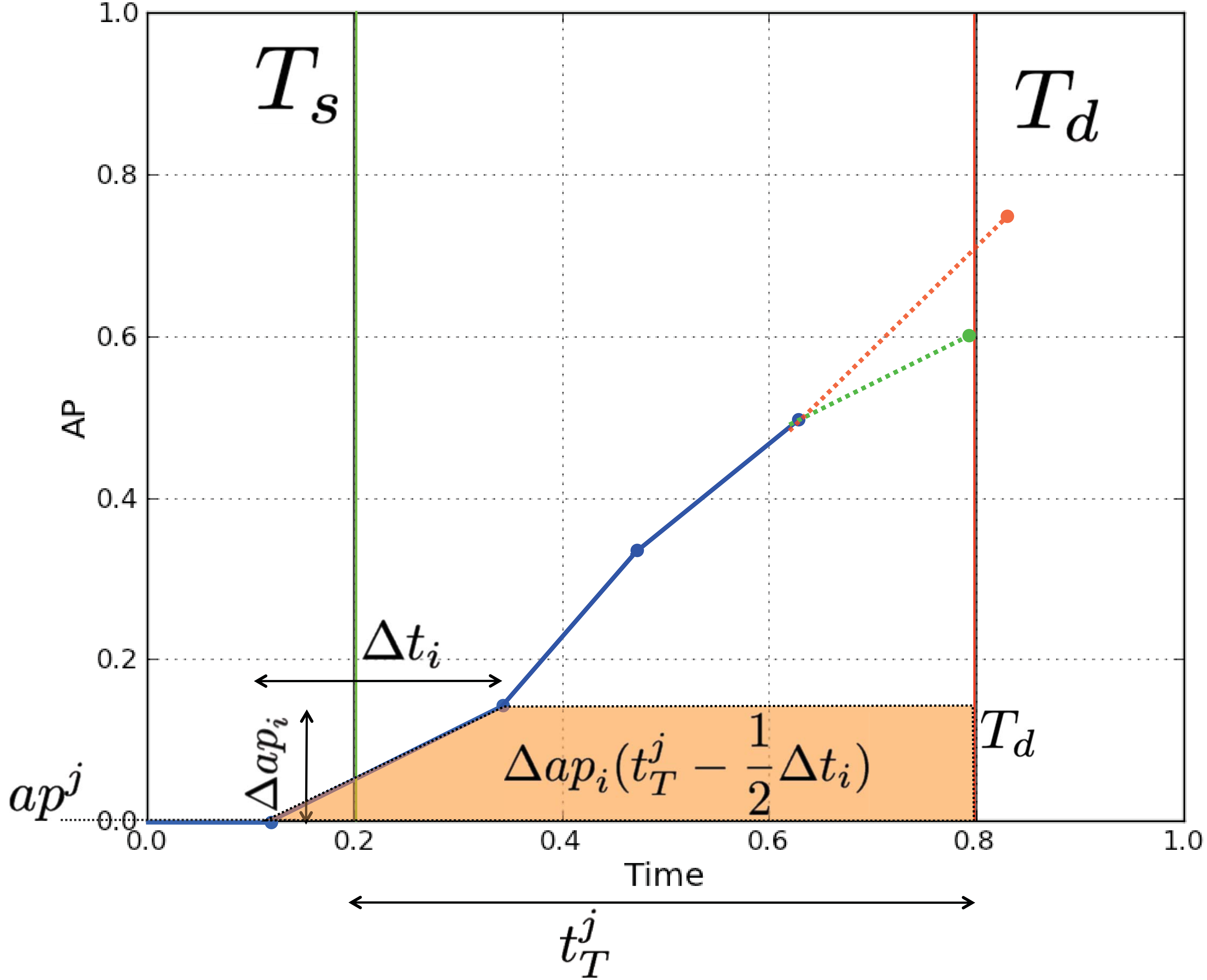
action-value function: $$Q^{\pi}(s, a_i) = \mathbb{E}_{s'}[R(s', a_i) + \gamma Q^{\pi}(s', \pi(s'))]$$

reward definition

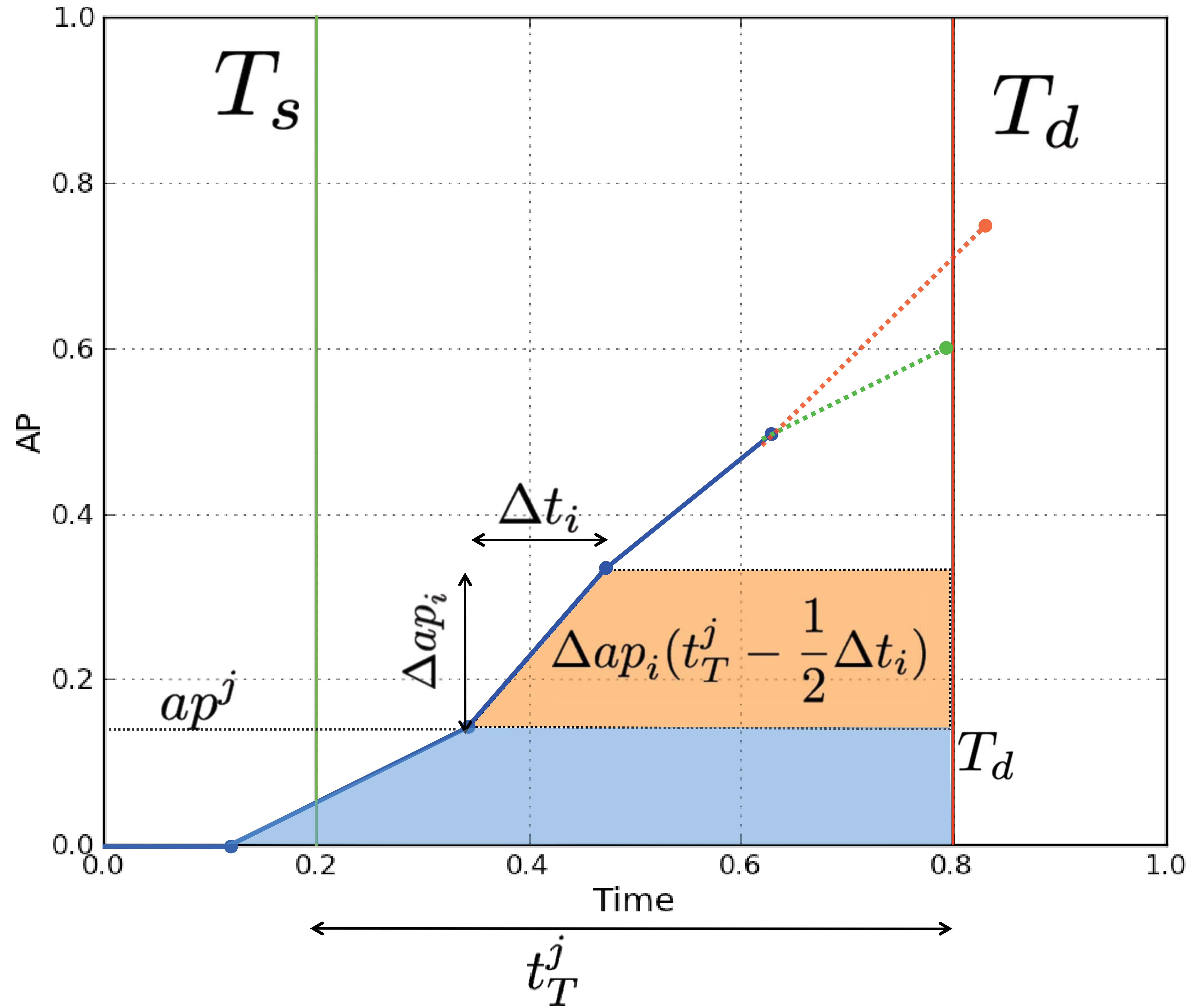
assume linearity: $$Q^{\pi}(s, a_i) = \theta_{\pi}^{\top} \phi(s, a_i)$$

learning the policy

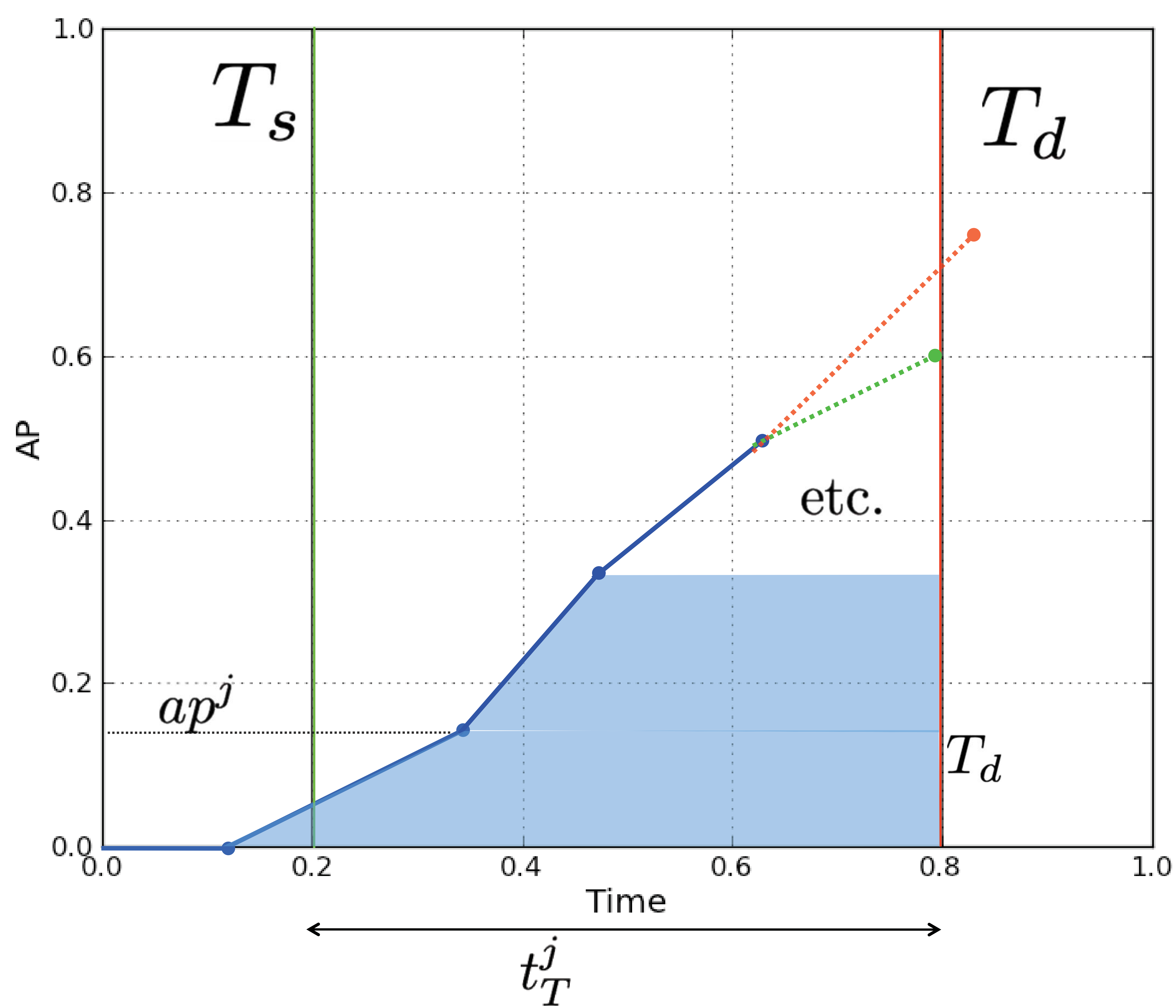
reward definition: Anytime performance



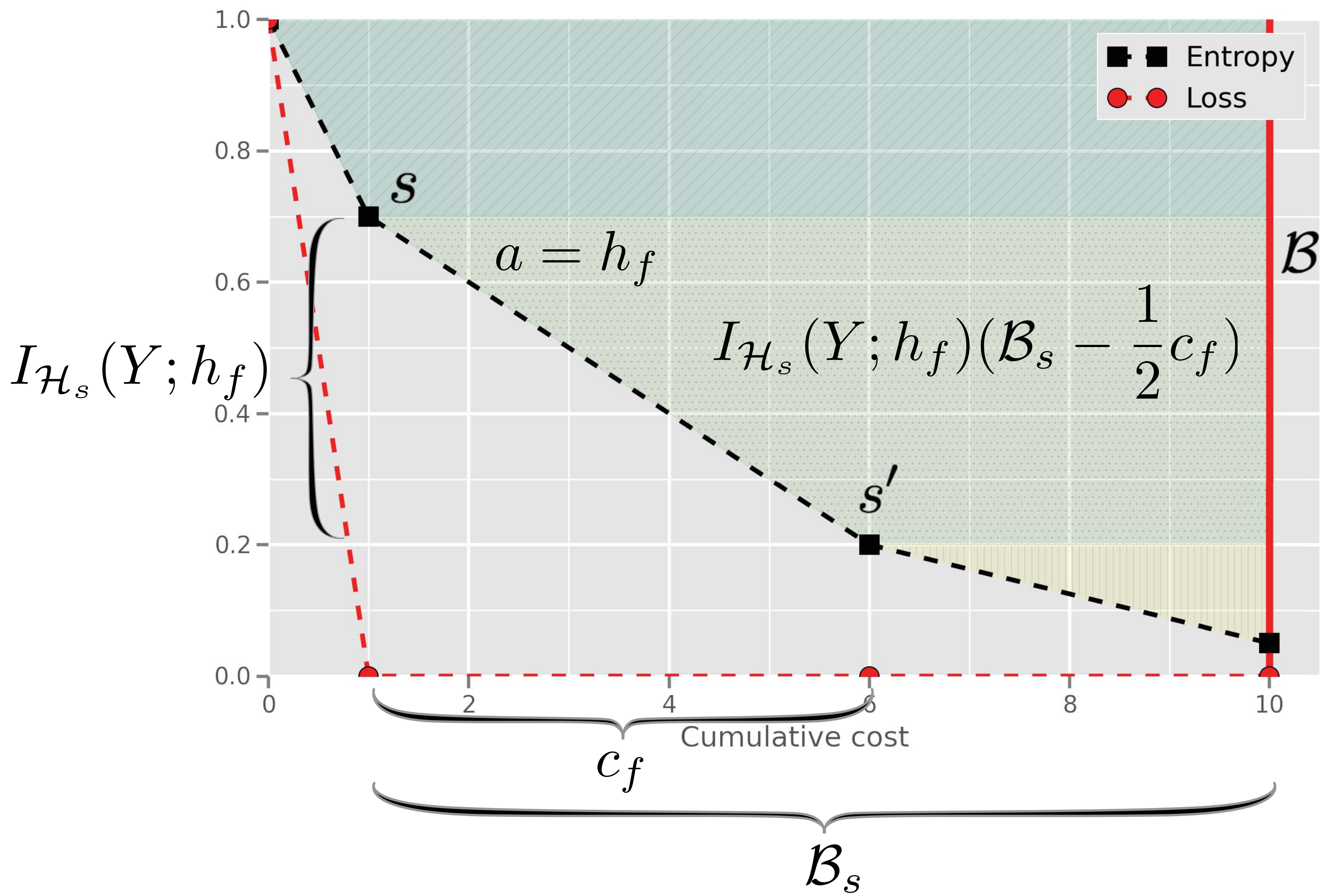
reward definition: Anytime performance



reward definition: Anytime performance



In practice, use infogain.



$$I(Y;\mathcal{H}_{\pi(x)}) = H(Y) - H(Y|\mathcal{H}_{\pi(x)}) = \sum_{y \in Y} P(y) \log P(y) - \sum_{y, \mathcal{H}_{\pi(x)}} P(y, \mathcal{H}_{\pi(x)}) \log P(y \mid \mathcal{H}_{\pi(x)})$$

action selection

maximize expected value

policy: $$\pi(s) = \arg\max_{a_i \in \mathcal{A} \setminus \mathcal{O}} Q(s, a_i)$$

action-value function: $$Q^\pi(s, a_i) = \mathbb{E}_{s'}\left[\underline{R(s', a_i)} + \gamma Q^\pi(s', \pi(s'))\right]$$

reward definition

assume linearity: $$Q^\pi(s, a_i) = \underline{\theta_\pi^\top} \phi(s, a_i)$$

learning the policy

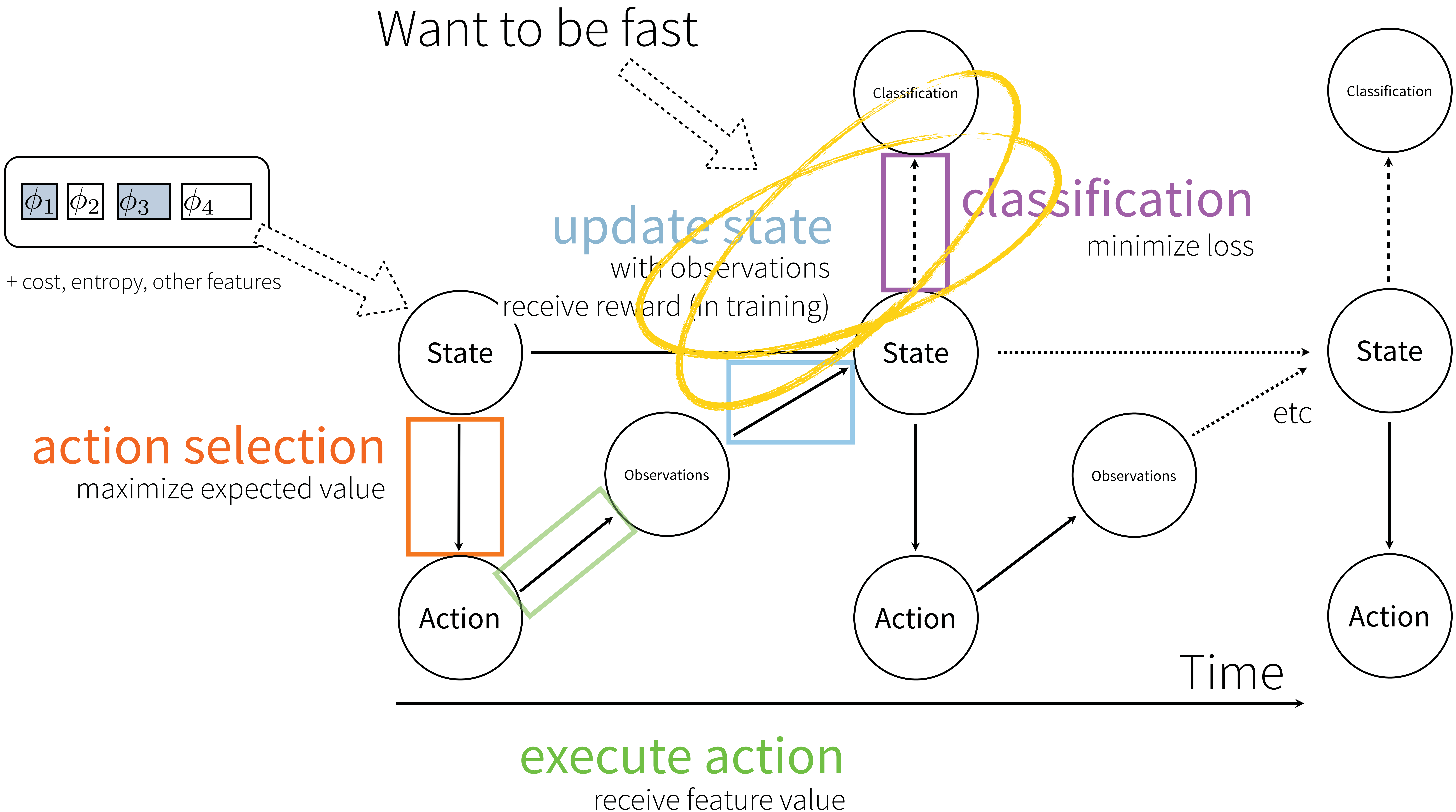
learning the policy

$$Q^\pi(s, a_i) = \mathbb{E}_{s'}[R(s', a_i) + \gamma Q^\pi(s', \pi(s'))]$$
$$Q^\pi(s, a_i) = \underline{\theta_\pi^\top} \phi(s, a_i)$$

- Sample the expectation:

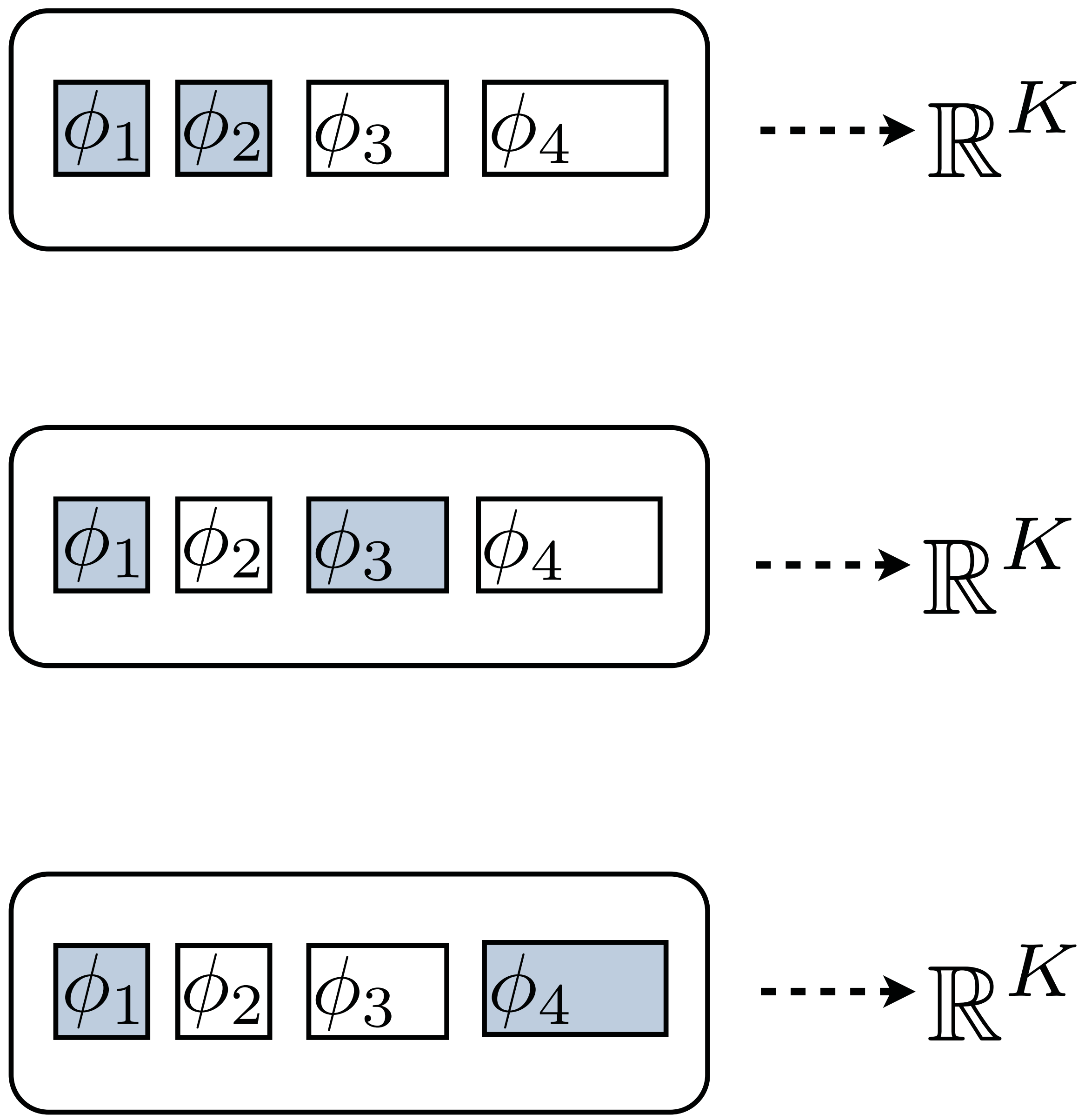
  collect (state, action, reward, state) tuples by

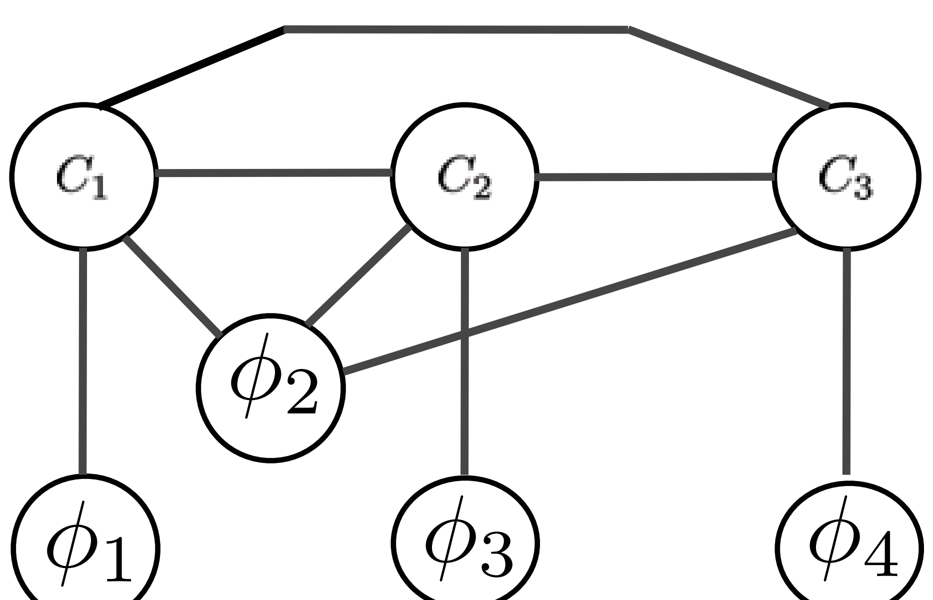
  executing current policy.
- Update the policy: solve for weights.
- Iterate.



Want to be fast
ϕ_1 ϕ_2 ϕ_3 ϕ_4
+ cost, entropy, other features
update state
with observations
receive reward (in training)
classification
minimize loss
Classification
Classification
State
State
State
action selection
maximize expected value
Observations
Observations
etc
Action
Action
Action
Time
execute action
receive feature value

How do we combine arbitrary subset of features?





Karayev et al. (NIPS 2012)

	Accuracy	Subset?	Cost
MRF model	✓	✓	✗
Naive Bayes	✗	✓	✓
Linear model	✓	?	✓

Two ideas:
Missing value imputation;
Learning multiple linear models.

Missing value imputation

- Mean

Missing value imputation

- Mean
- SVD

The rank-R truncated SVD of $N \times F$ matrix X^c can be written as

$$\hat{X}^c = U_R D_R V_R^T \tag{1}$$

The unobserved values $\mathbf{x}^{\mathrm{u}}$ are filled in by $V_R^{\mathrm{u}} \left(V_R^{\mathrm{o}T} V_R^{o}\right)^{-1} V_R^{\mathrm{o}T} \mathbf{x}^{\mathrm{o}}$.

Missing value imputation

- Mean

- SVD

The rank-R truncated SVD of $N \times F$ matrix X^c can be written as

$$\hat{X}^c = U_R D_R V_R^T \tag{1}$$

The unobserved values $\mathbf{x}^{\mathrm{u}}$ are filled in by $V_R^{\mathrm{u}} \left(V_R^{\mathrm{o}T} V_R^{o} \right)^{-1} V_R^{\mathrm{o}T} \mathbf{x}^{\mathrm{o}}$.

- Gaussian

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}^{\mathrm{o}} \\ \mathbf{x}^{\mathrm{u}} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^T & \mathbf{B} \end{bmatrix}\right) \tag{3}$$

$$\mathbf{x}^{\mathrm{u}} \mid \mathbf{x}^{\mathrm{o}} \sim \mathcal{N}\left(\mathbf{C}^T \mathbf{A}^{-1} \mathbf{x}^{\mathrm{o}}, \mathbf{B} - \mathbf{C}^T \mathbf{A}^{-1} \mathbf{C}\right) \tag{4}$$

Missing value imputation

- Mean
- SVD

The rank-R truncated SVD of $N \times F$ matrix X^c can be written as

$$\hat{X}^c = U_R D_R V_R^T \tag{1}$$

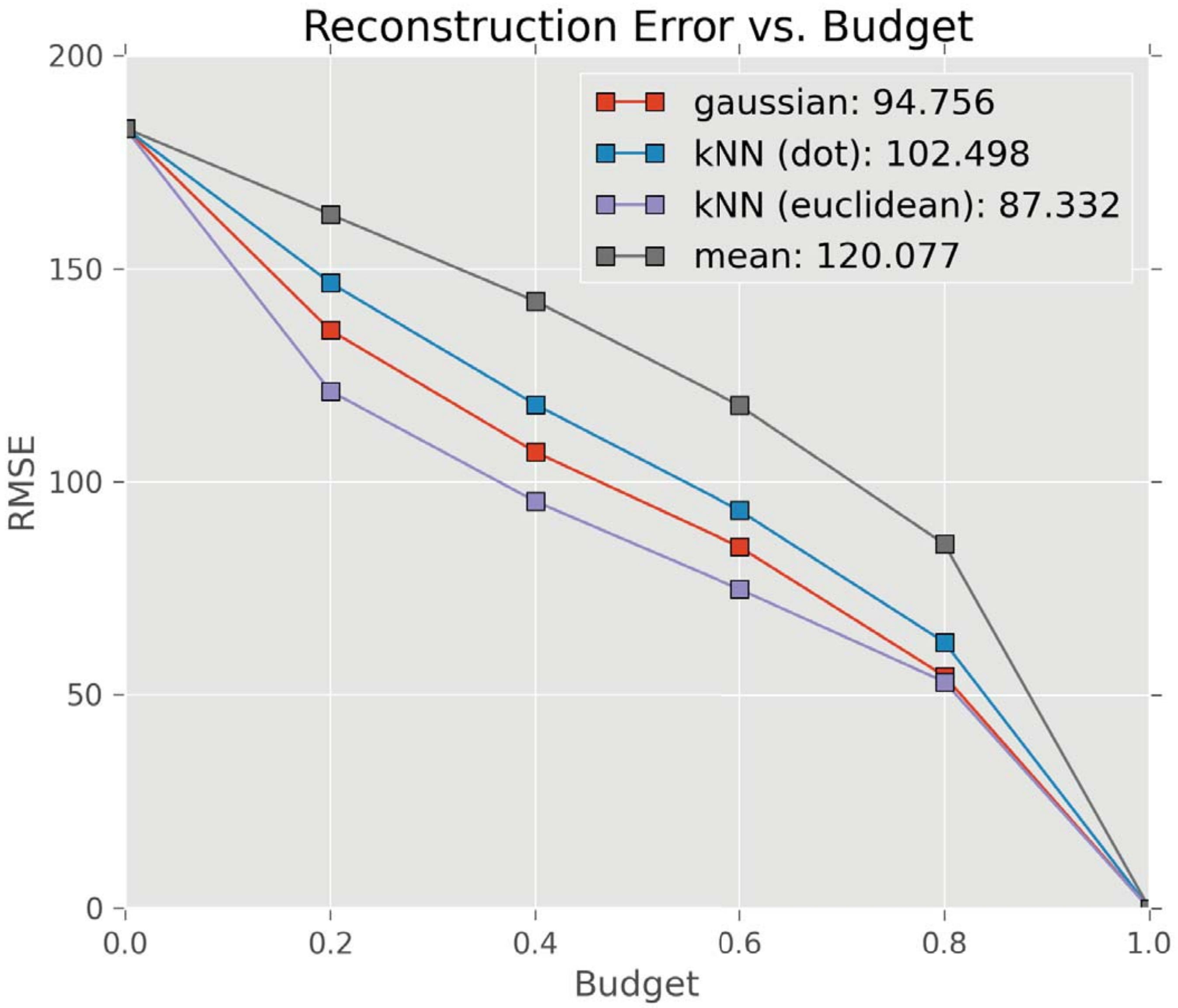
The unobserved values $\mathbf{x}^{\mathrm{u}}$ are filled in by $V_R^{\mathrm{u}} \left(V_R^{\mathrm{o}T} V_R^{o} \right)^{-1} V_R^{\mathrm{o}T} \mathbf{x}^{\mathrm{o}}$.

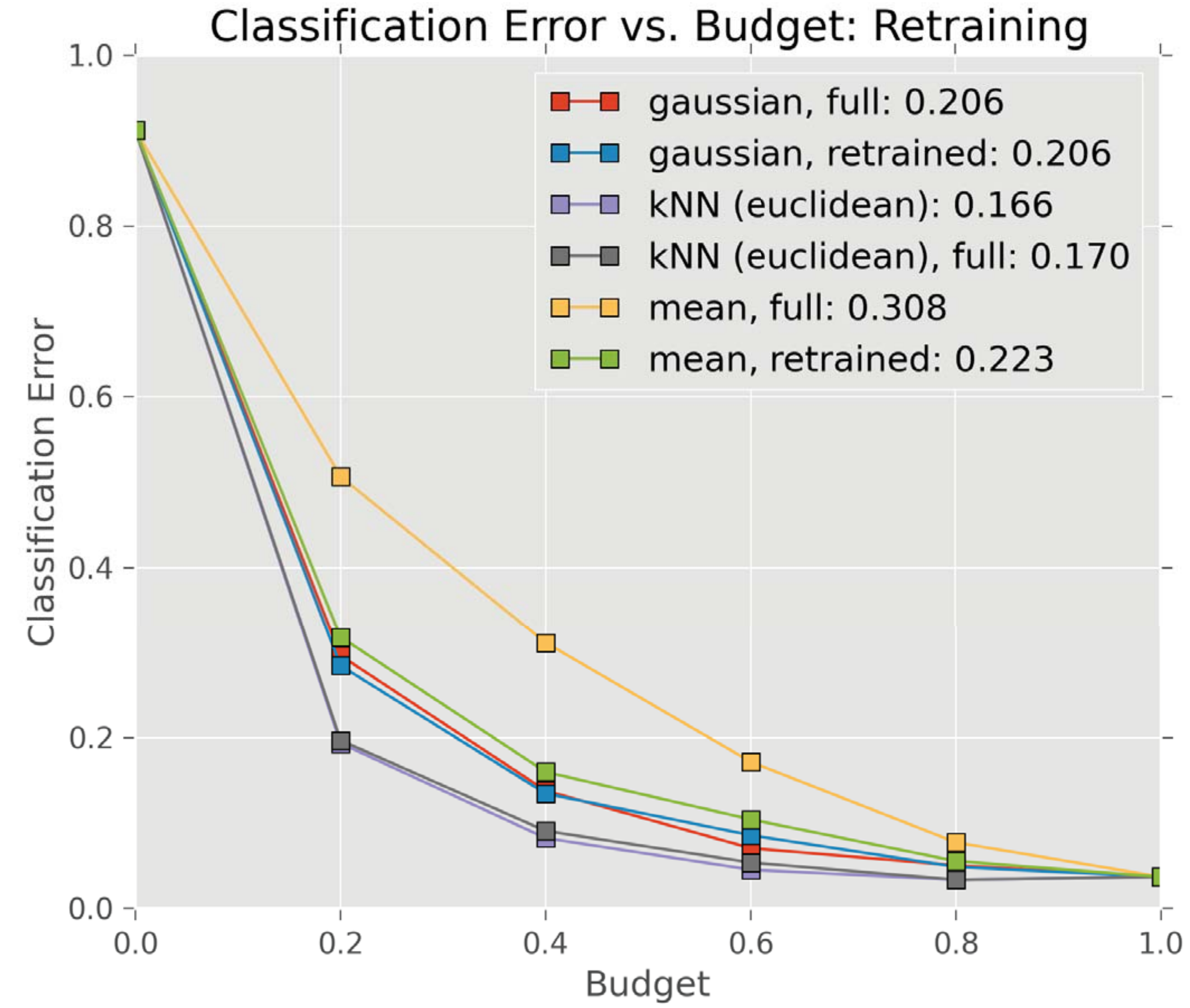
- Gaussian

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}^{\mathrm{o}} \\ \mathbf{x}^{\mathrm{u}} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^T & \mathbf{B} \end{bmatrix} \right) \tag{3}$$

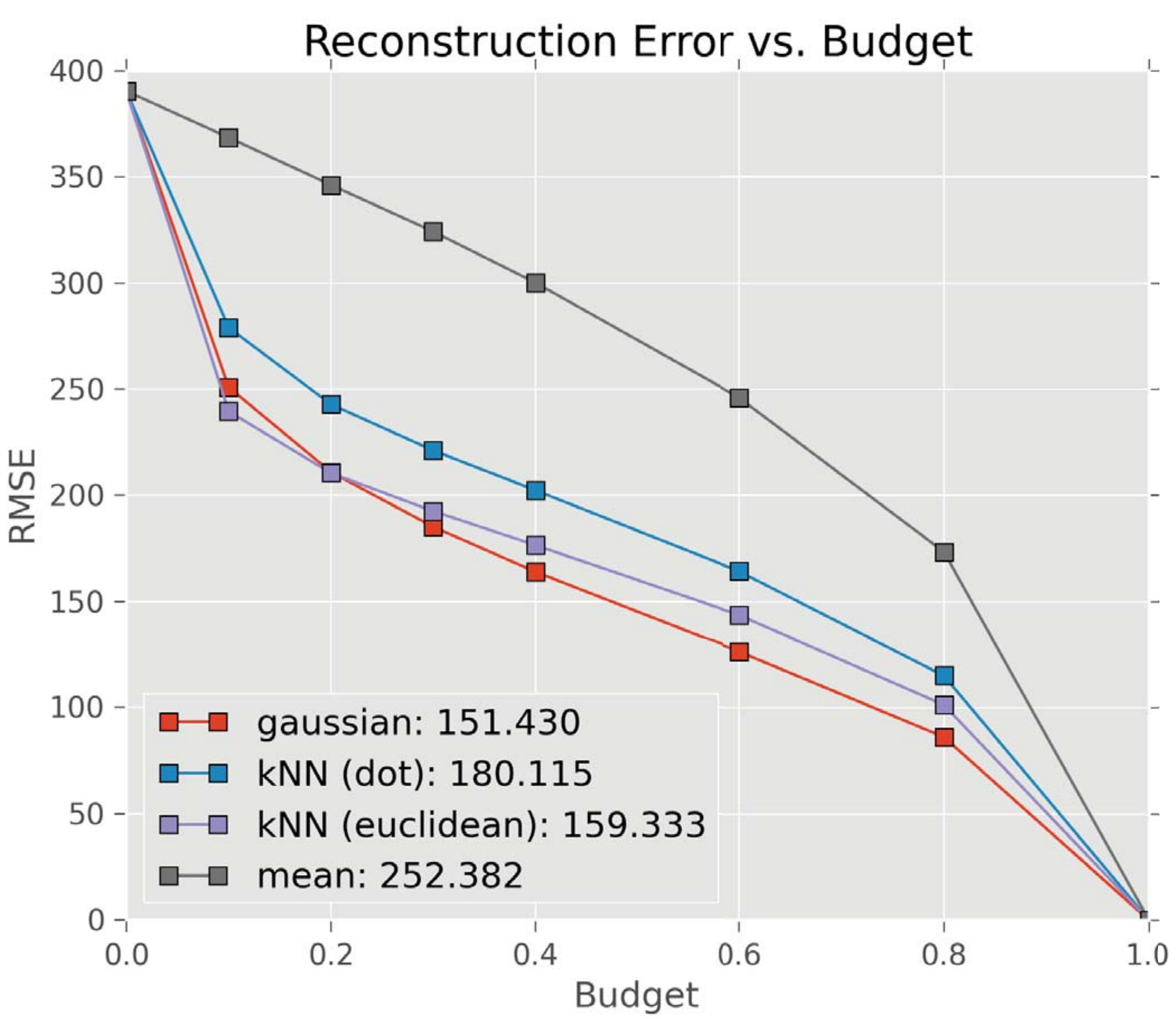
$$\mathbf{x}^{\mathrm{u}} \mid \mathbf{x}^{\mathrm{o}} \sim \mathcal{N}\left(\mathbf{C}^T \mathbf{A}^{-1} \mathbf{x}^{\mathrm{o}}, \mathbf{B} - \mathbf{C}^T \mathbf{A}^{-1} \mathbf{C} \right) \tag{4}$$

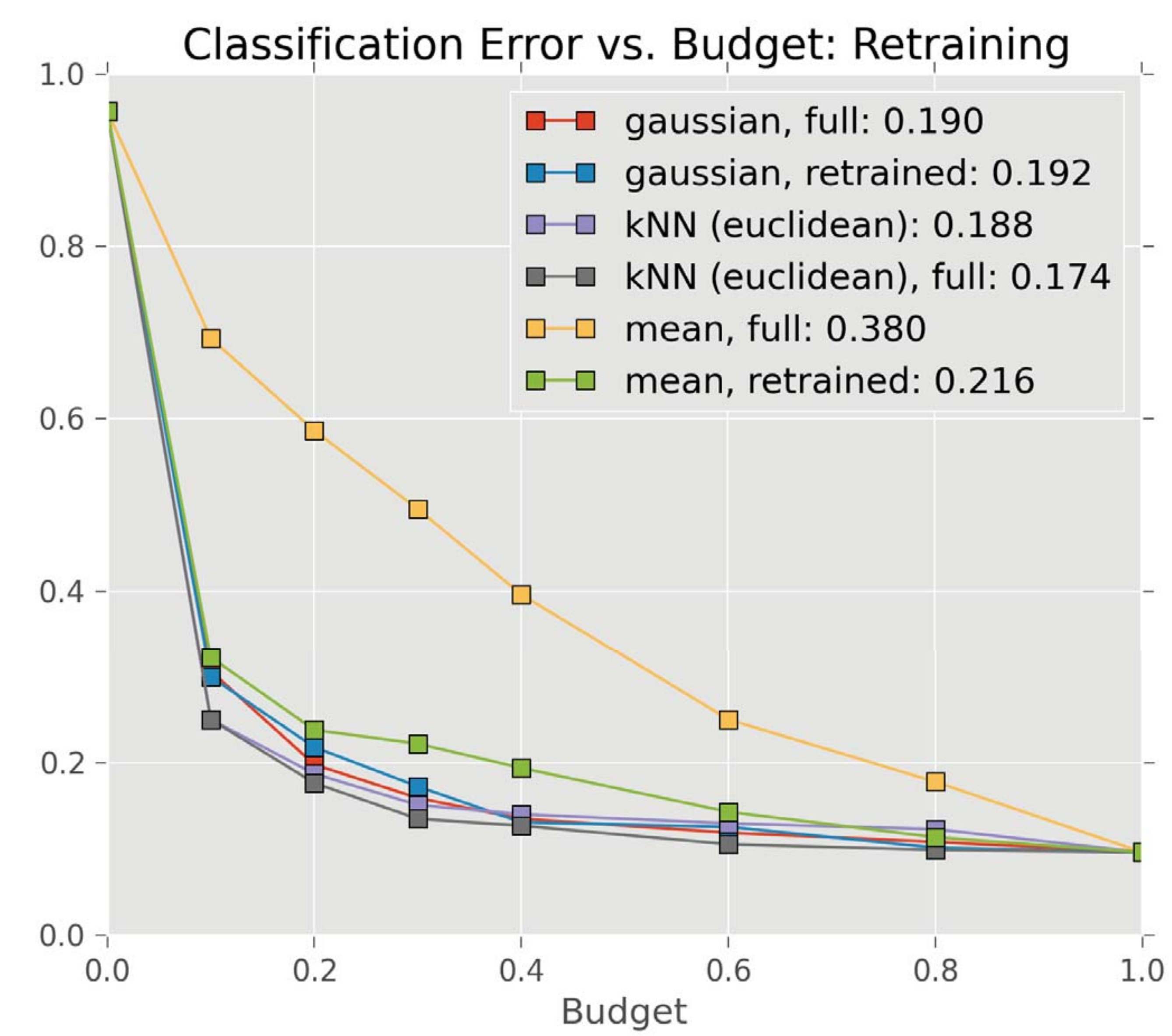
- kNN





Digits





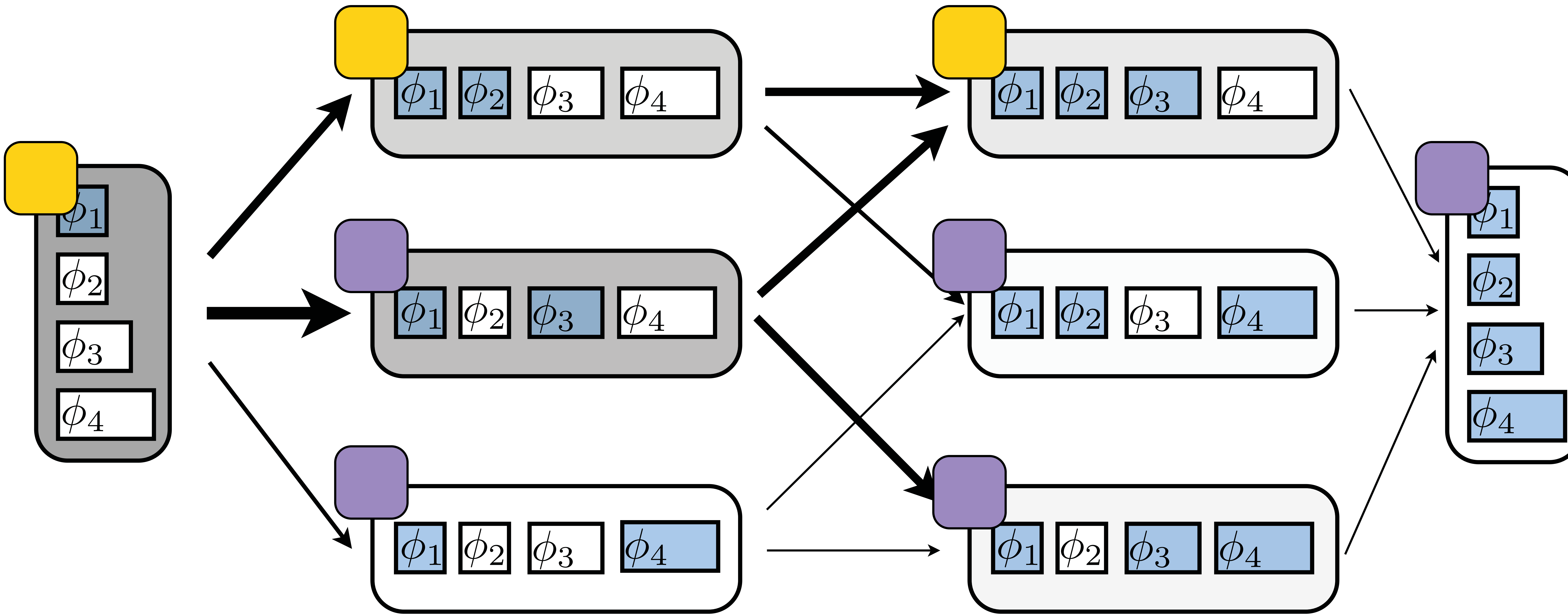
Scenes

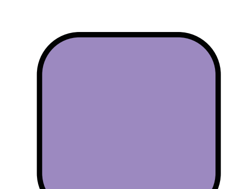
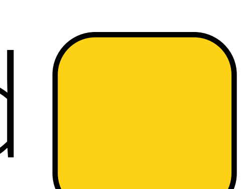
Training on feature vectors with correctly missing value distribution.

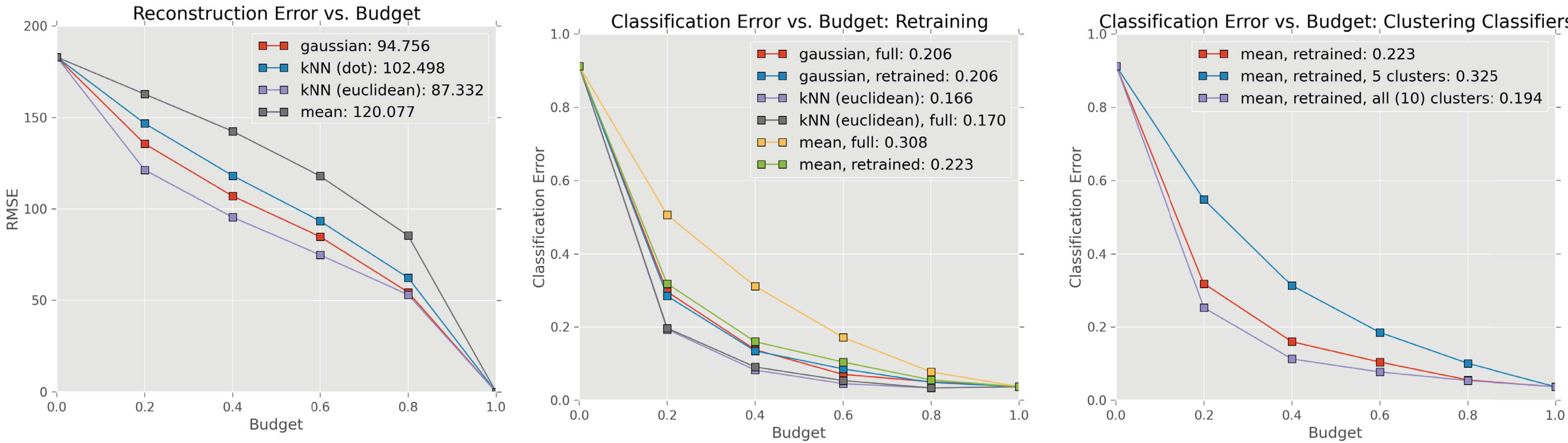
Input: $\mathcal{D} = \{x_n, y_n\}_{n=1}^{N}$; $\mathcal{L}_{\mathcal{B}}$
Result: Trained π, g

$\pi_0 \leftarrow$ random;
for $i \leftarrow 1$ **to** *max_iterations* **do**
  States, Actions, Costs, Labels $\leftarrow$ `GatherSamples(`$\mathcal{D}$, π_{i-1}`)`;
  $g_i \leftarrow$ `UpdateClassifier(`*States, Labels*`)`;
  Rewards $\leftarrow$ `ComputeRewards(`*States, Costs, Labels,* $g_i, \mathcal{L}_{\mathcal{B}}, \gamma$`)`;
  $\pi_i \leftarrow$ `UpdatePolicy(`*States, Actions, Rewards*`)`;
end

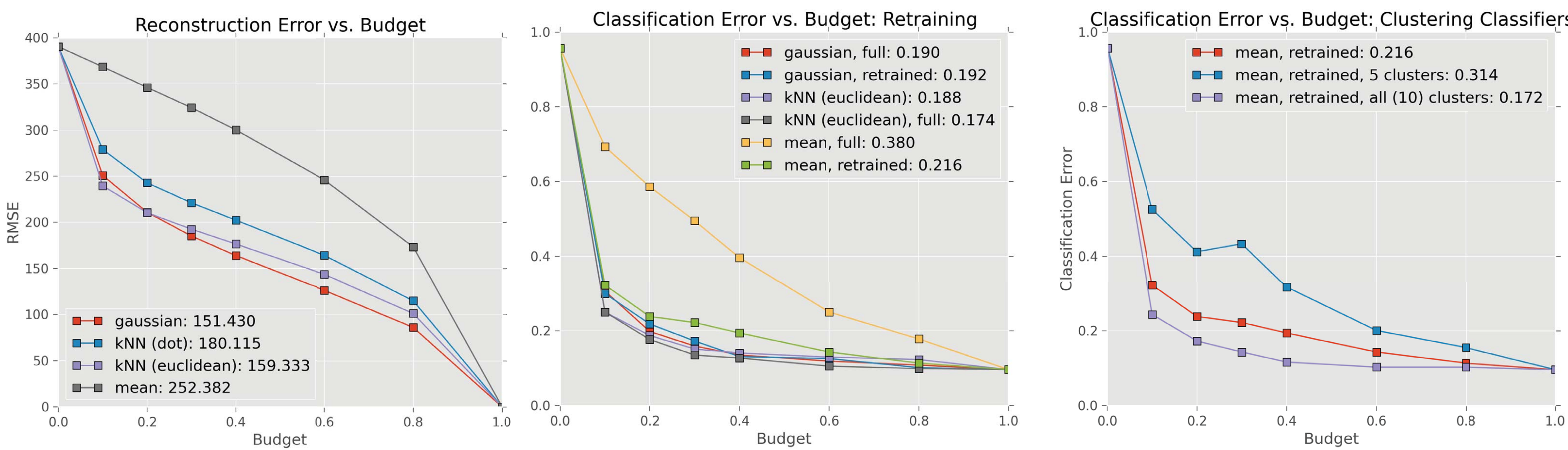
Learning multiple classifiers.



In this case, 2 separate models:  and 



Digits



Scenes

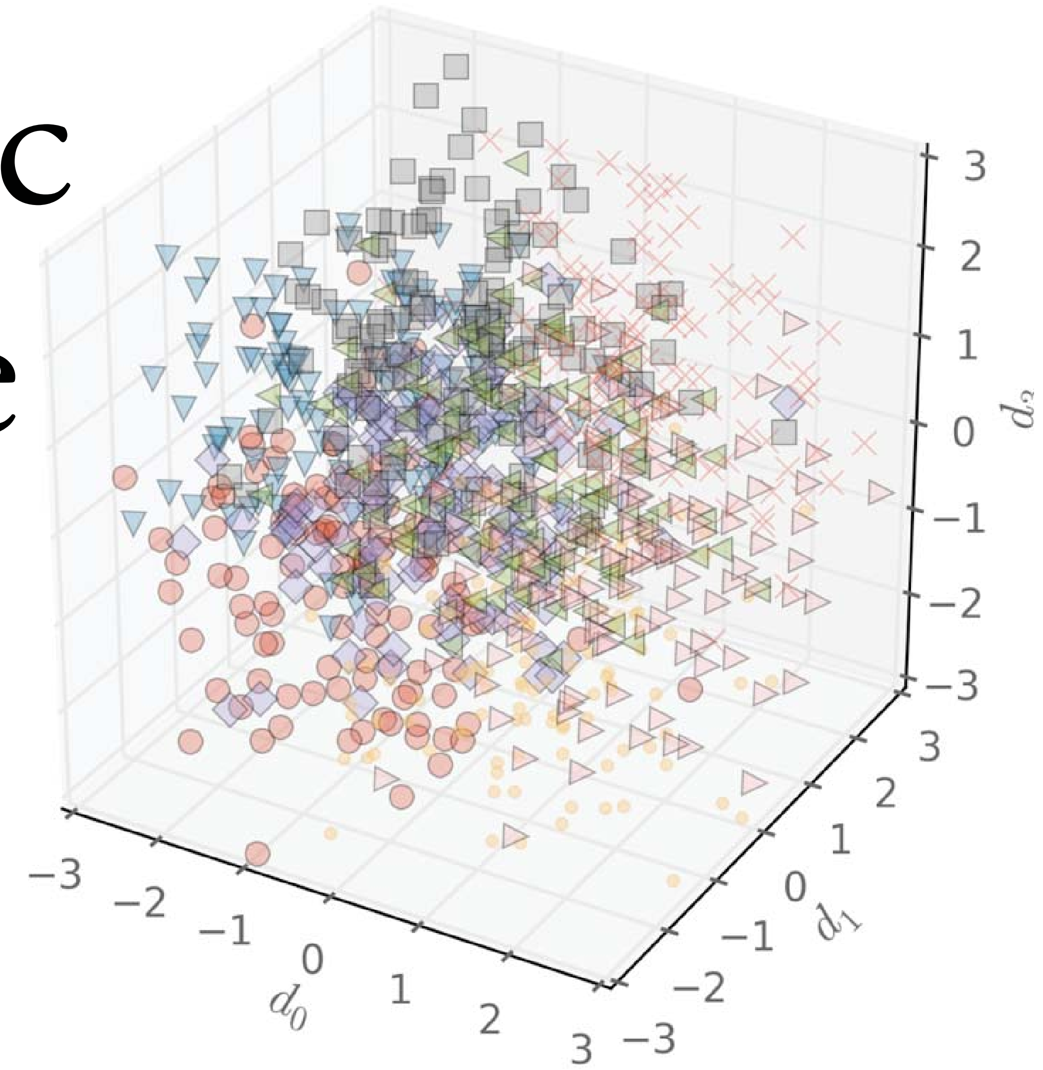
Dynamic results.

We evaluate the following baselines:

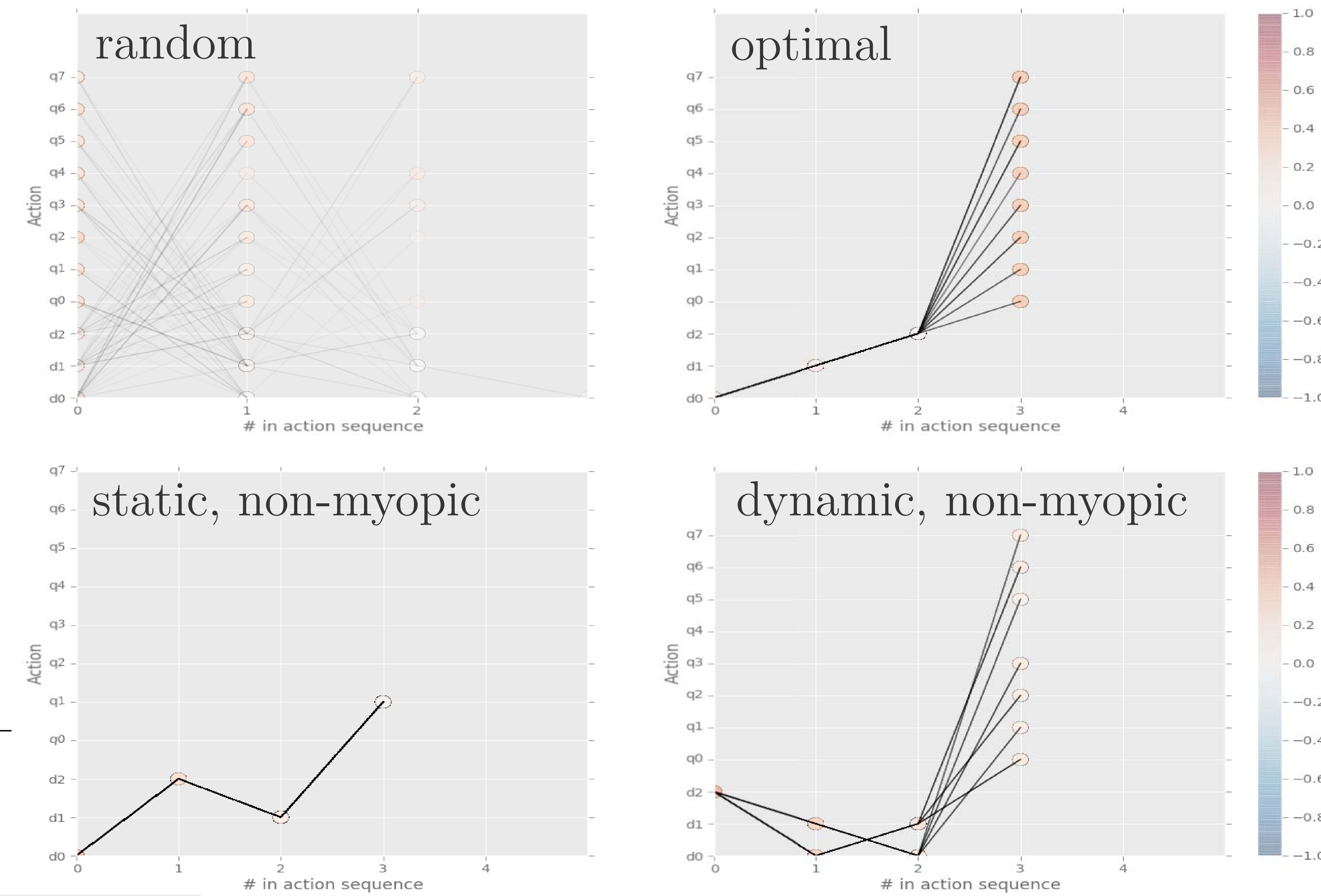
- **Static, greedy**: corresponds to best performance of a policy that does not observe feature values and selects actions greedily ($\gamma = 0$).
- **Static, non-myopic**: policy that does not observe values but considers future action rewards ($\gamma = 1$).
- **Dynamic, greedy**: policy that observes feature values, but selects actions greedily.

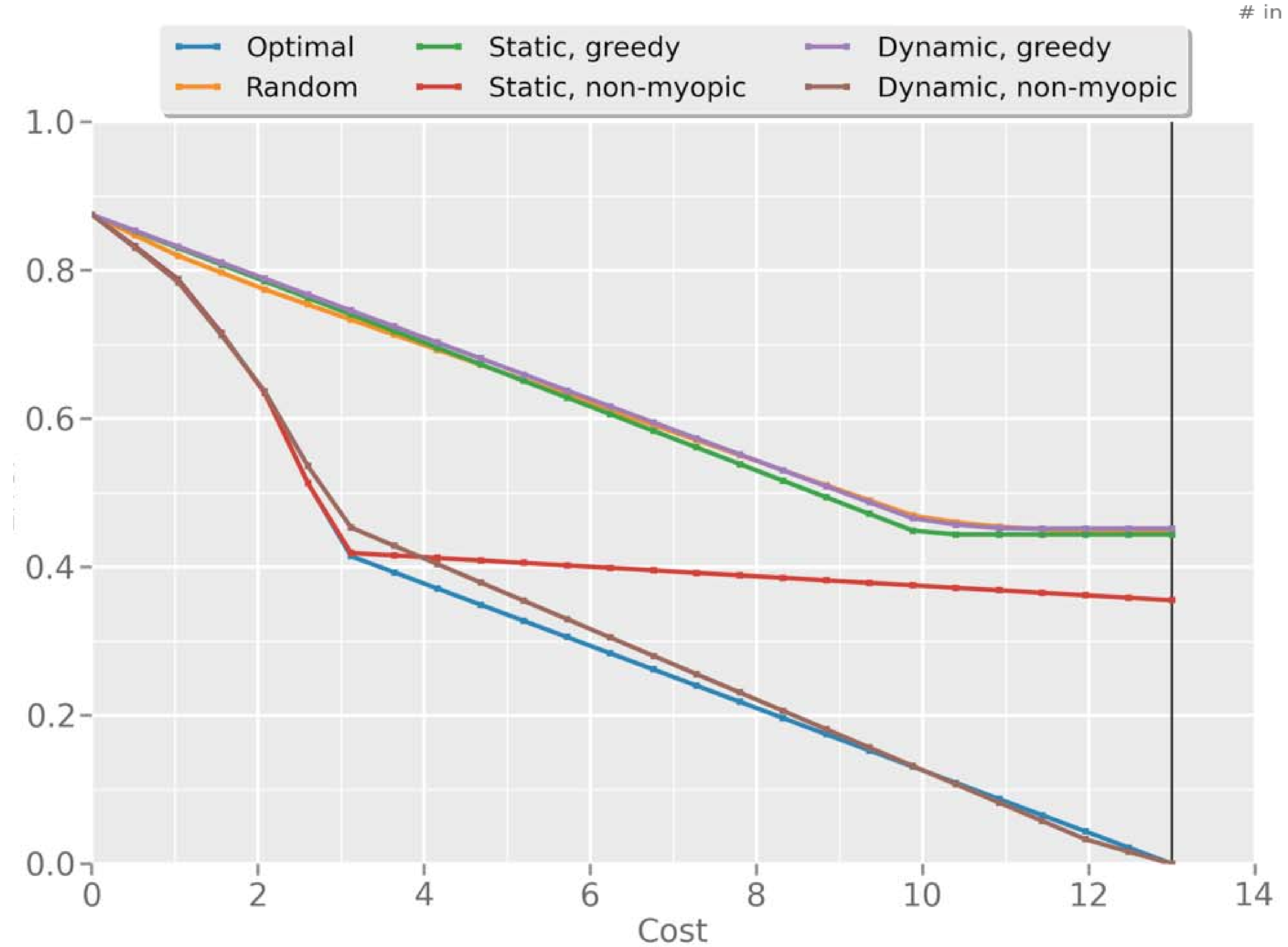
Our method is the **Dynamic, non-myopic** policy: feature values are observed, with full lookahead.

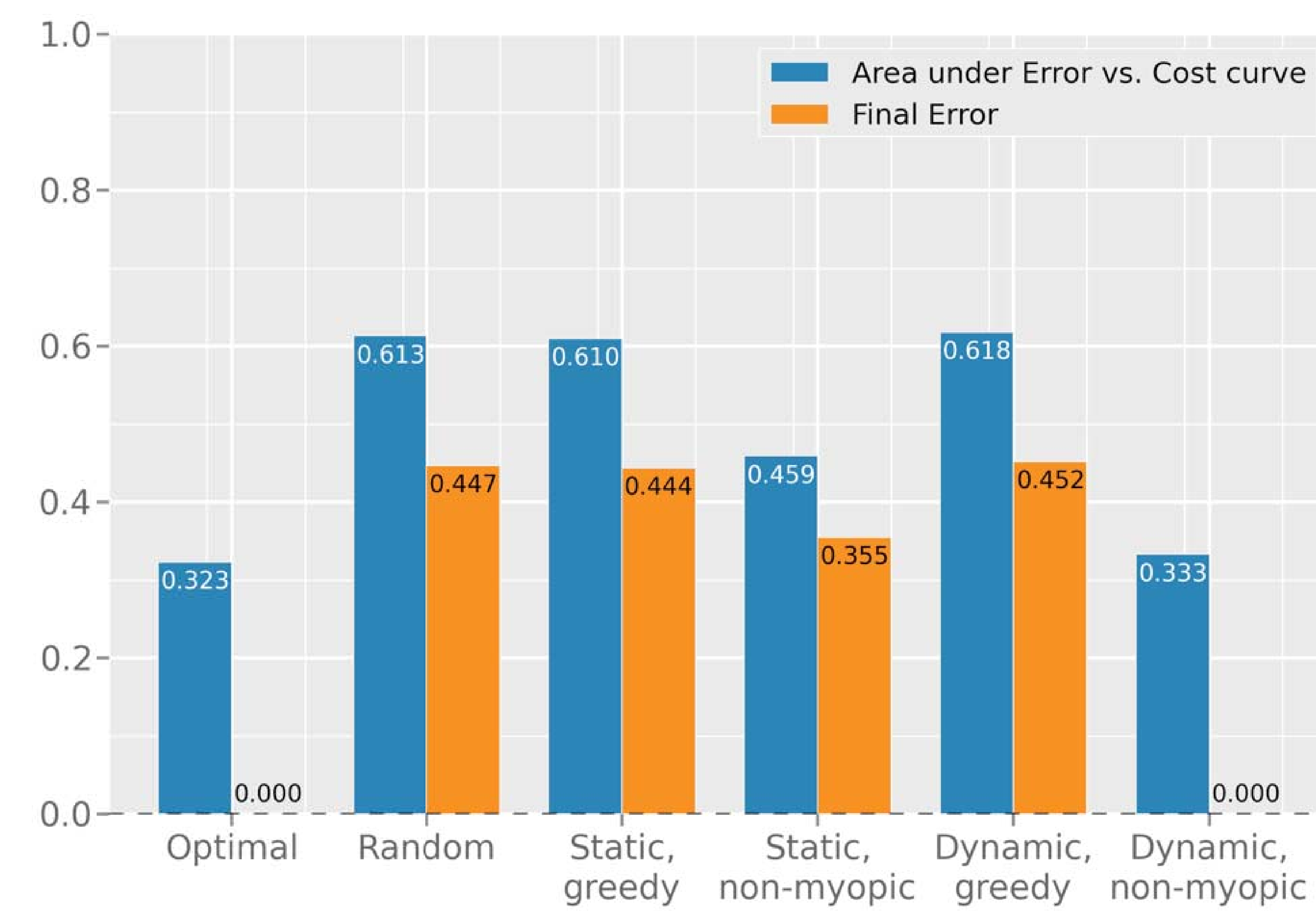
Synthetic Example



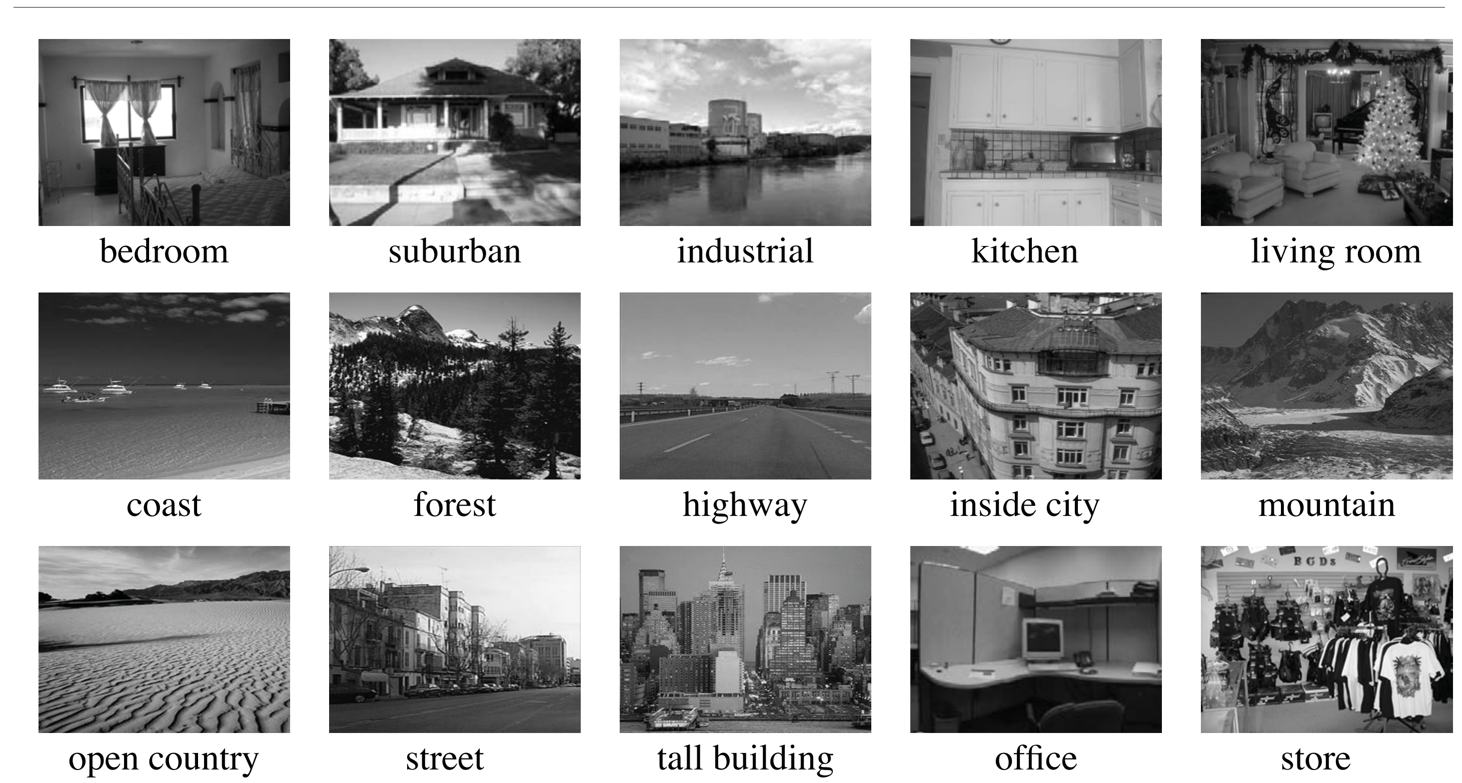
Feature	Number	Cost
d_i: sign of dimension i	D	1
q_o: label of datapoint, if in quadrant o	2^D	10

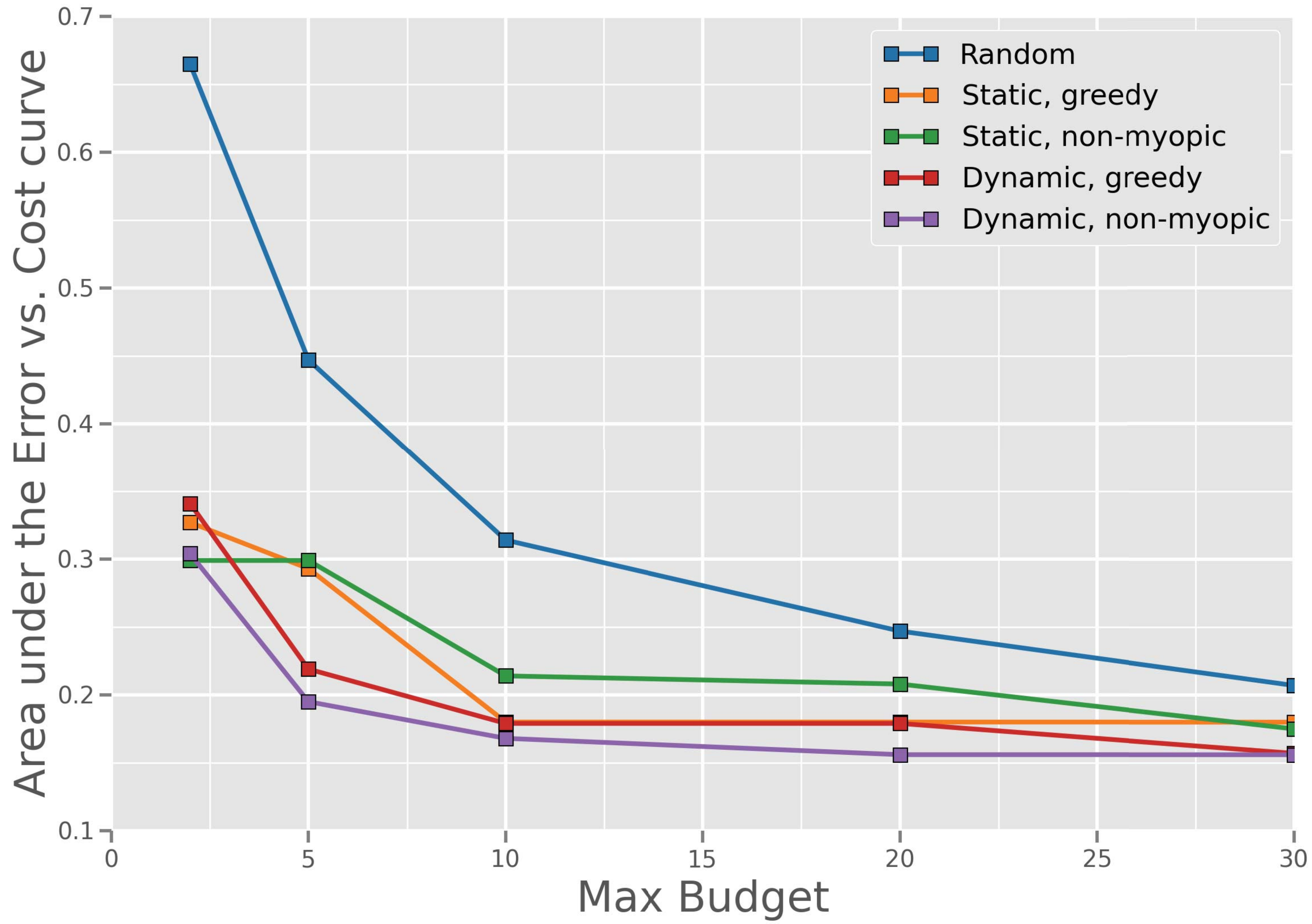


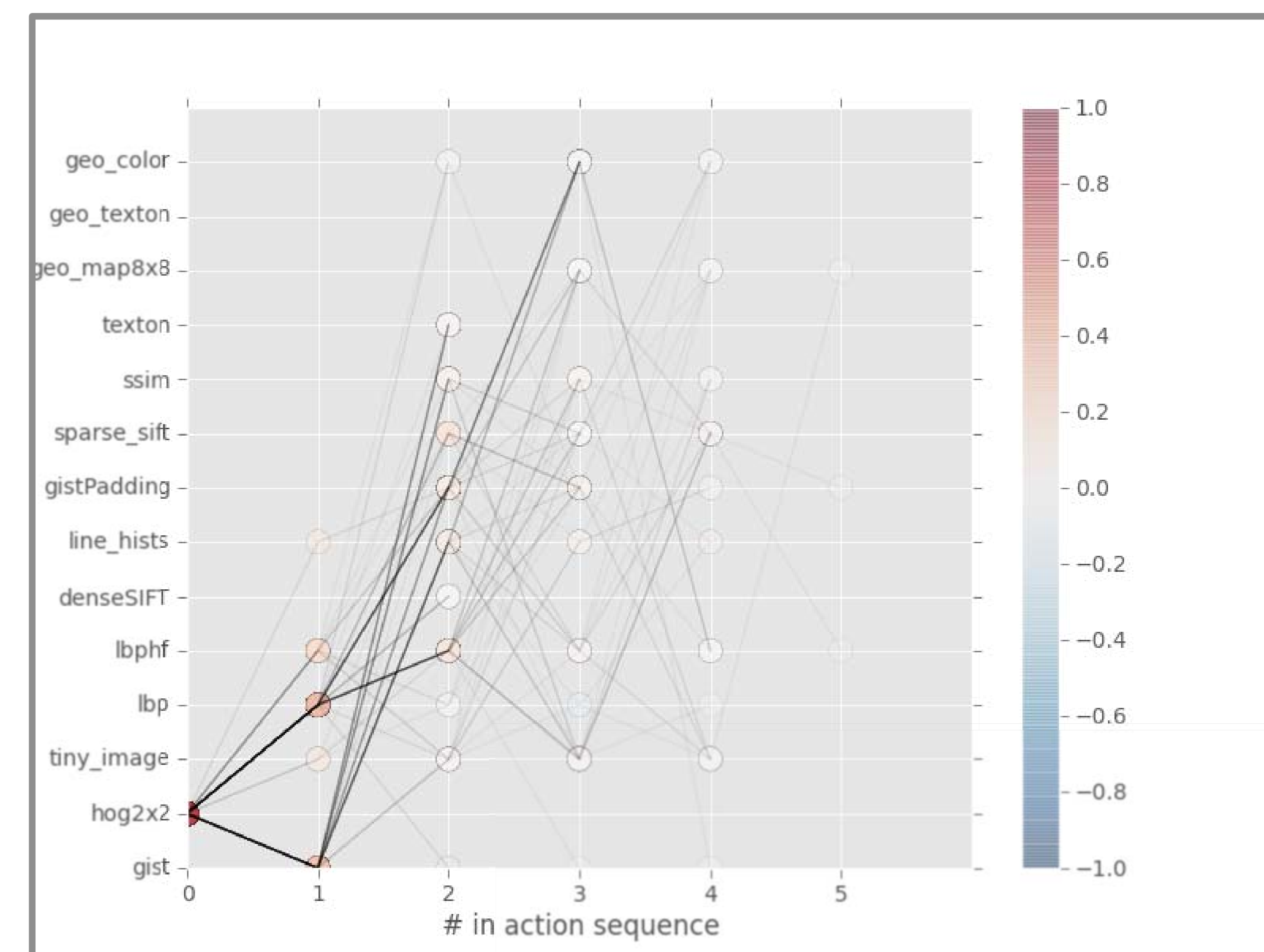




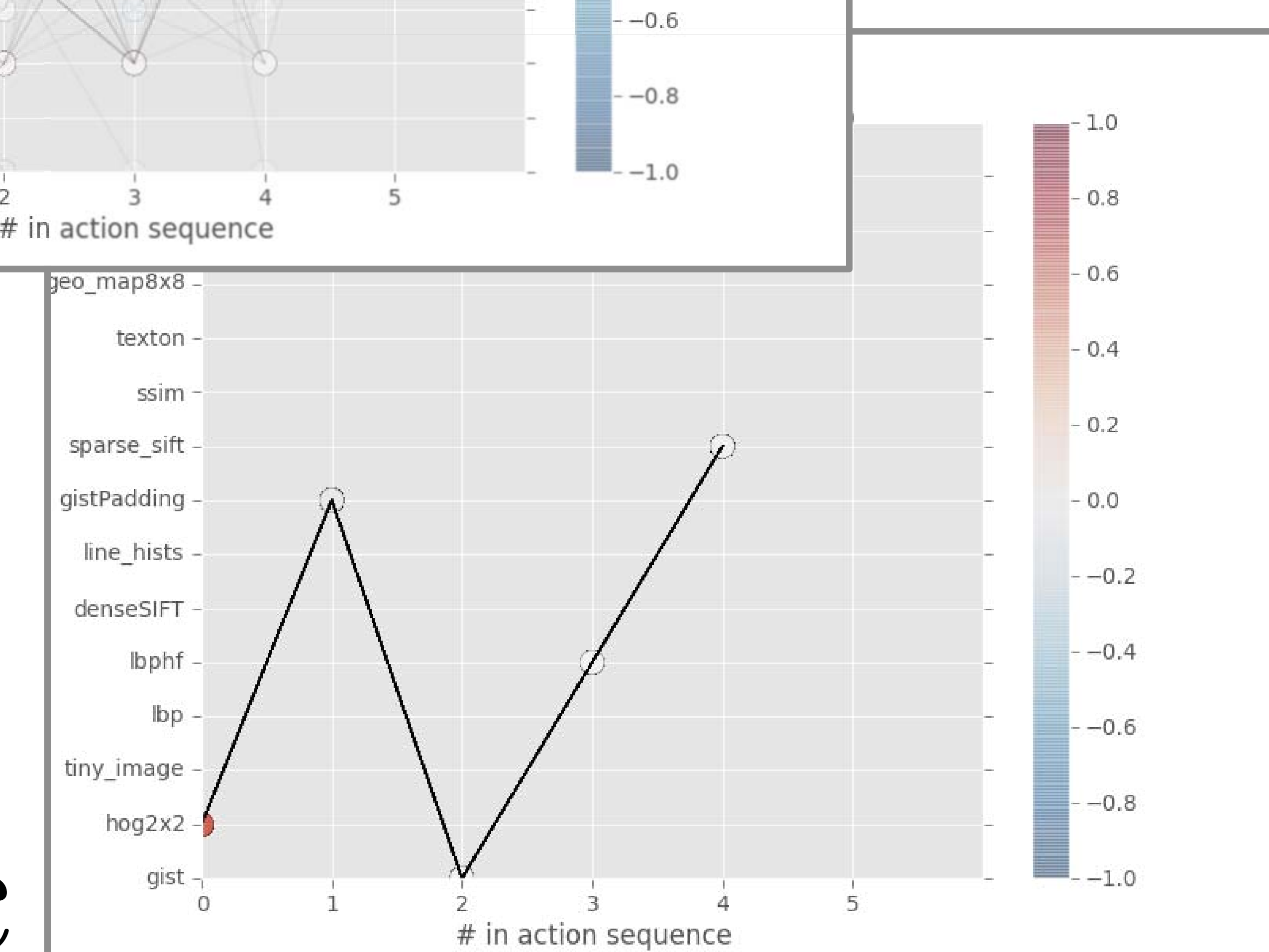
Scenes-15





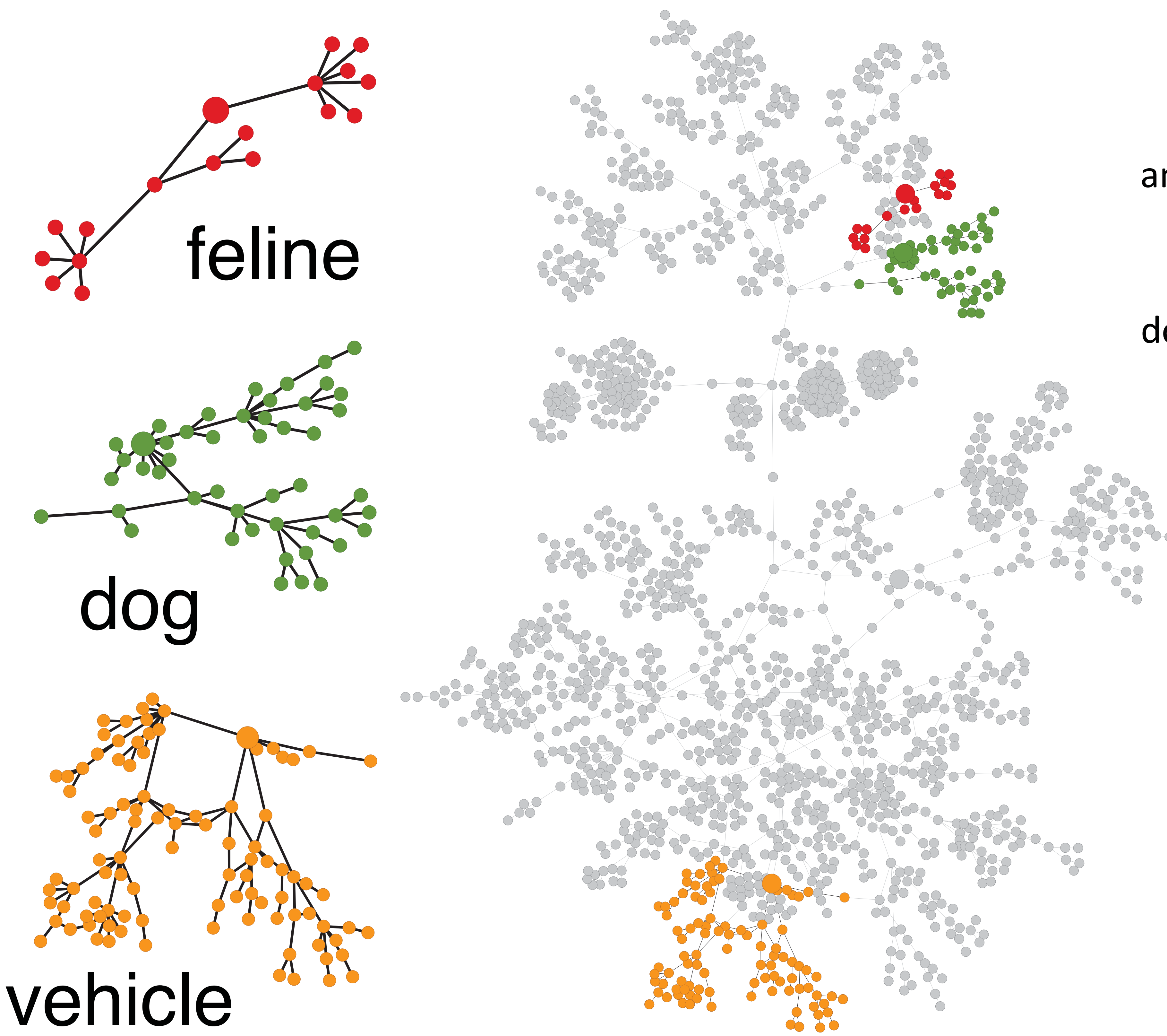


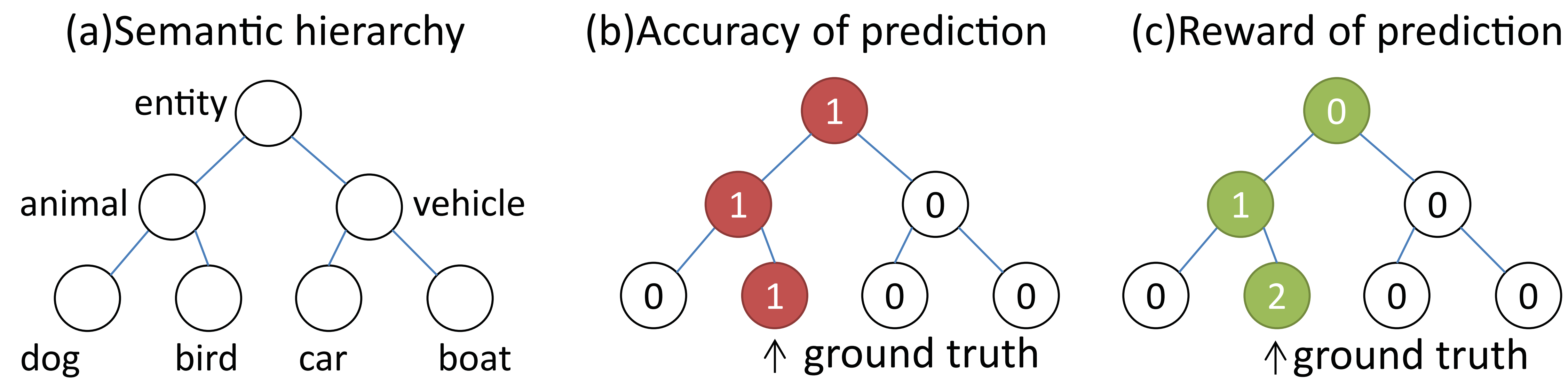
Dynamic

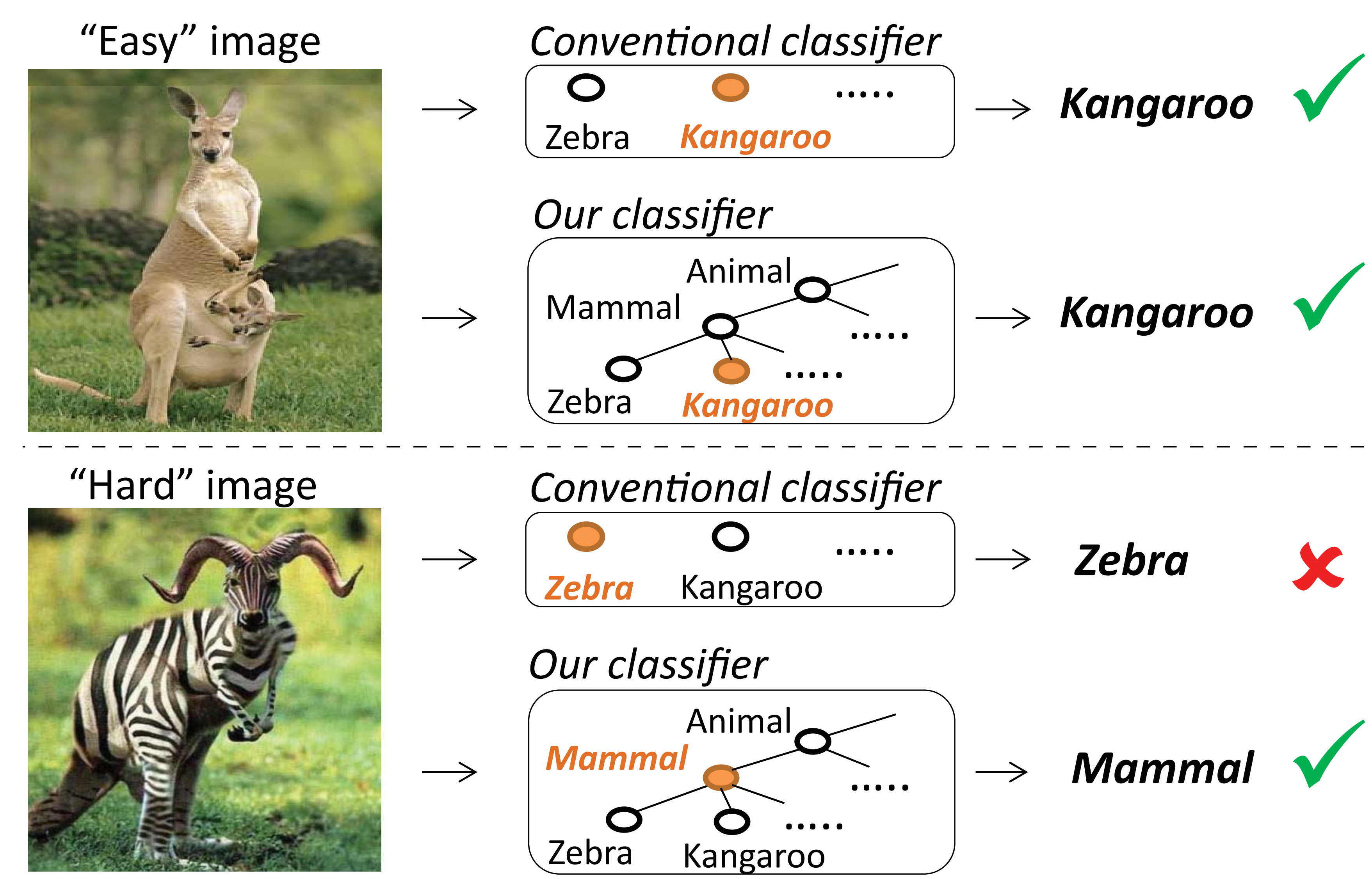


Static

IMAGENET

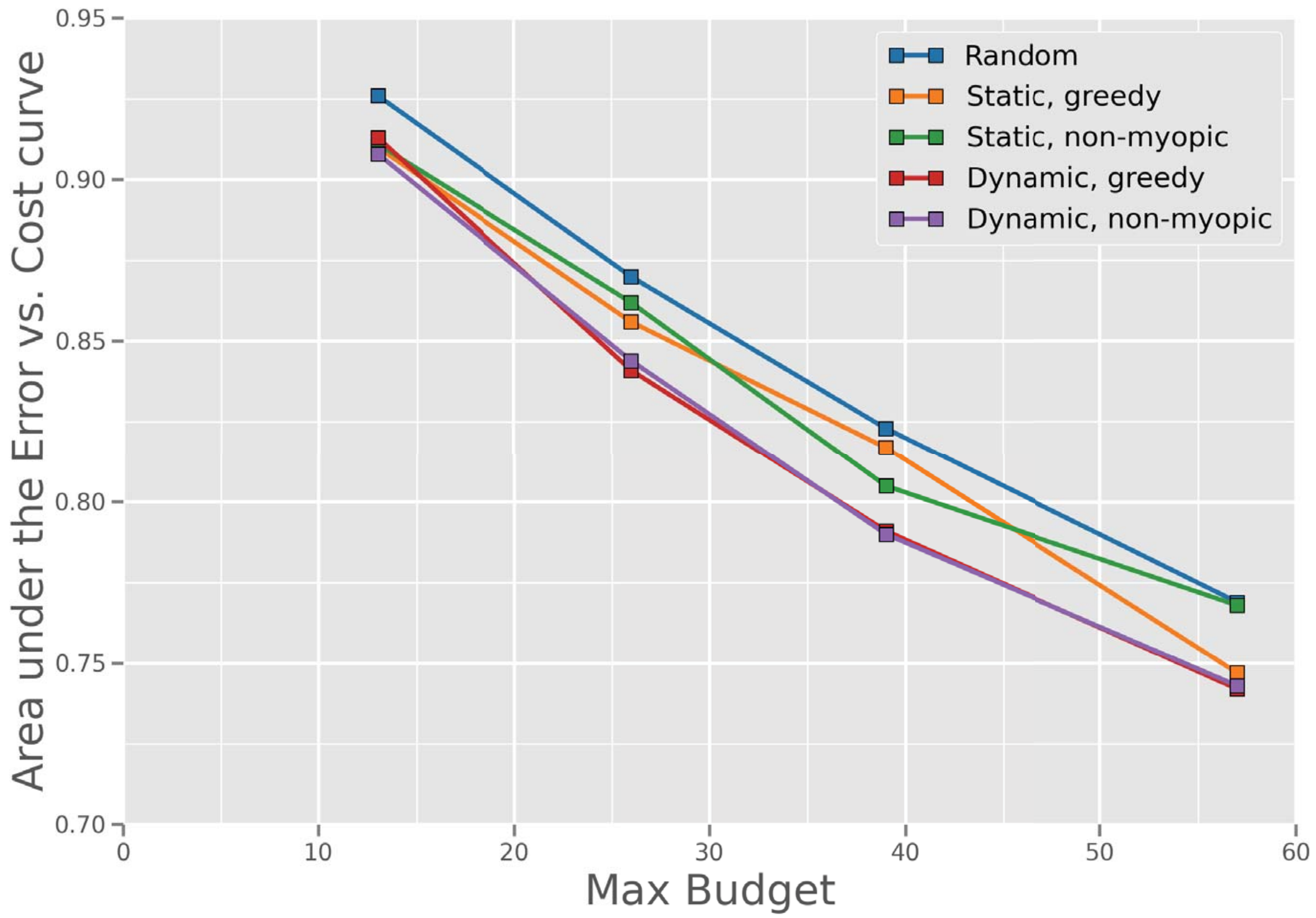


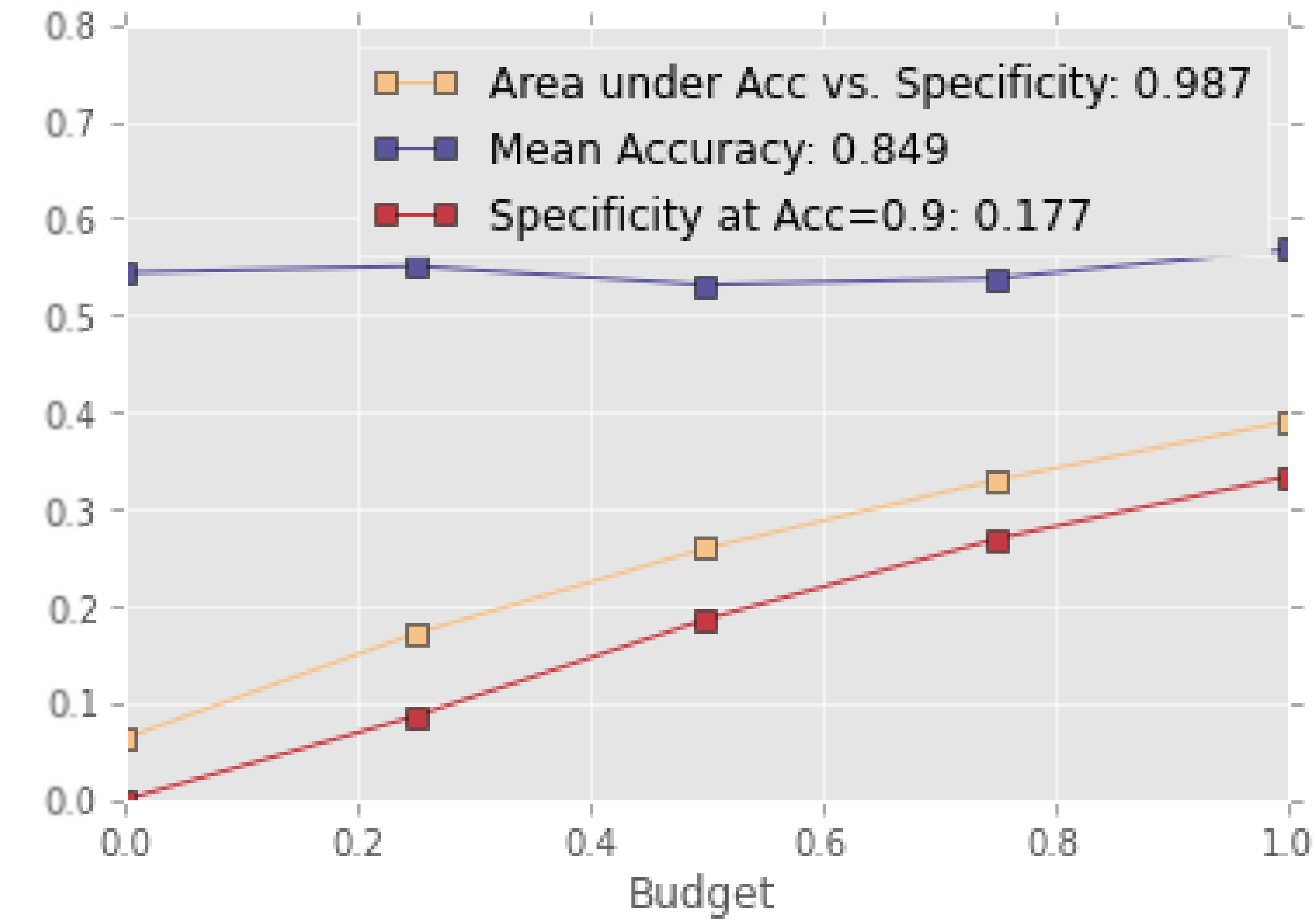


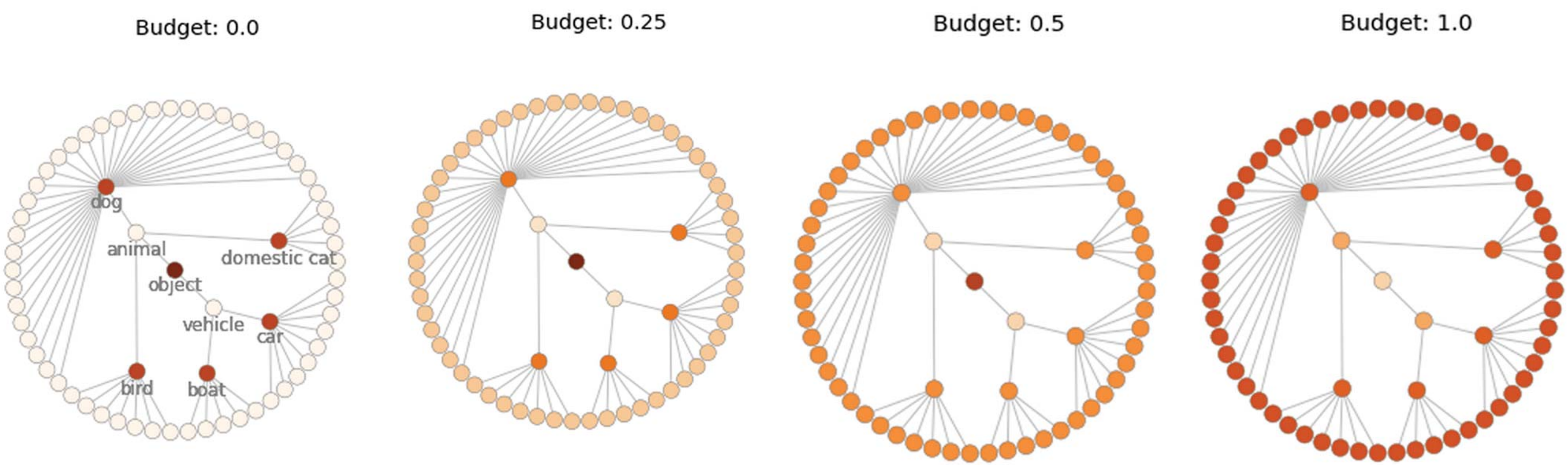


(Figures from Jia Deng at Stanford and Yangqing Jia at Berkeley).

ILSVRC-65


Random
Static, greedy
Static, non-myopic
Dynamic, greedy
Dynamic, non-myopic
Area under the Error vs. Cost curve
Max Budget
0.95
0.90
0.85
0.80
0.75
0.70
0
10
20
30
40
50
60


Area under Acc vs. Specificity: 0.987
Mean Accuracy: 0.849
Specificity at Acc=0.9: 0.177
Budget
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.0
0.2
0.4
0.6
0.8
1.0


Budget: 0.0
Budget: 0.25
Budget: 0.5
Budget: 1.0
dog
animal
domestic cat
object
vehicle
car
bird
boat

Thank you.