Paper Reading

2022.03.30

Guan Yunyi

VERAM: View-Enhanced Recurrent Attention Model for 3D Shape Classification

Songle Chen , Lintao Zheng , Yan Zhang , Zhixin Sun , and Kai Xu

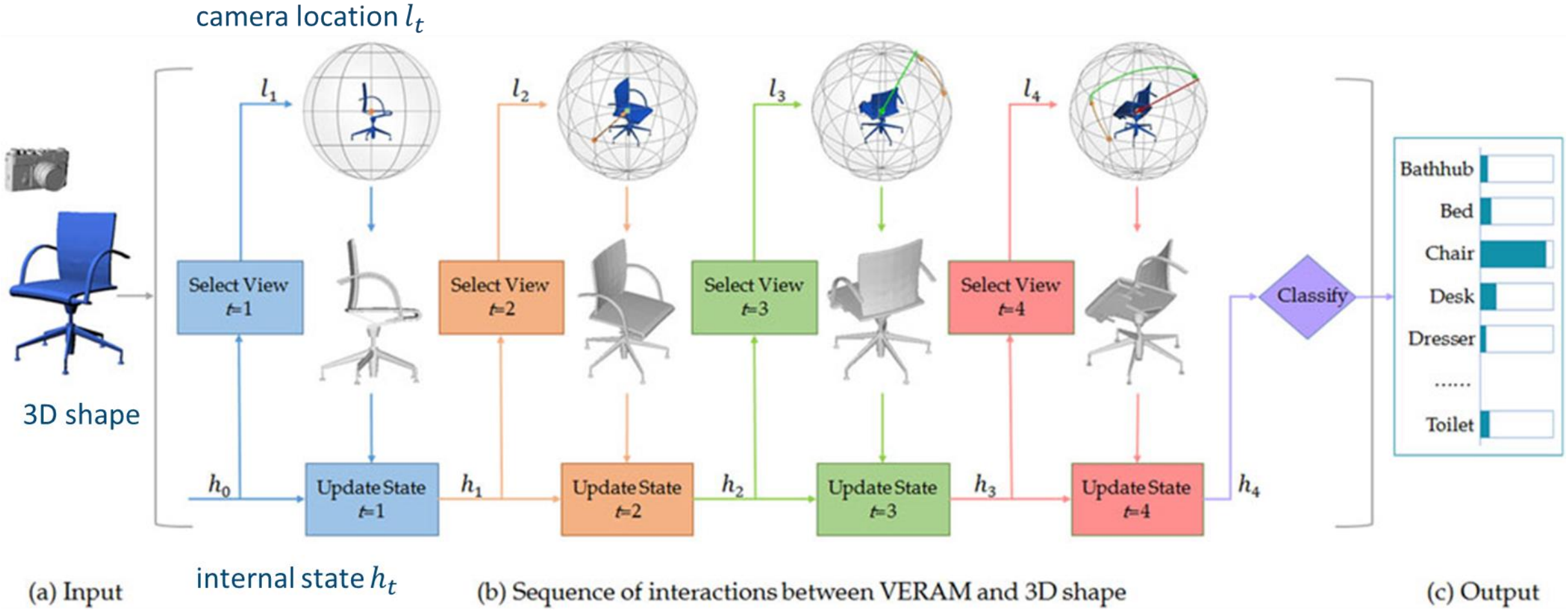
Key words: multi-view 3D shape recognition, visual attention model, RNN, RL

01

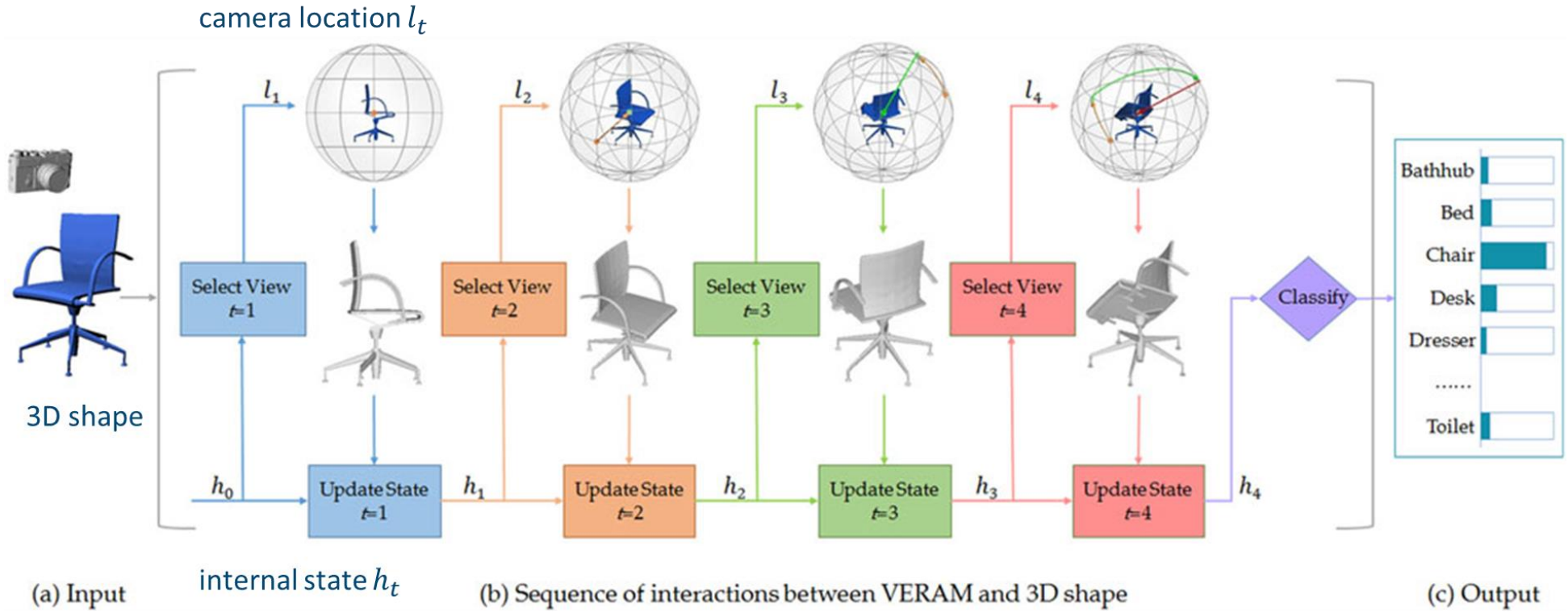
Introduction

1. VERAM

- **Actively** selecting views with **RL and RNN**
- Training: **SGD + REINFORCE** with **3 view-enhanced strategies**



1. VERAM

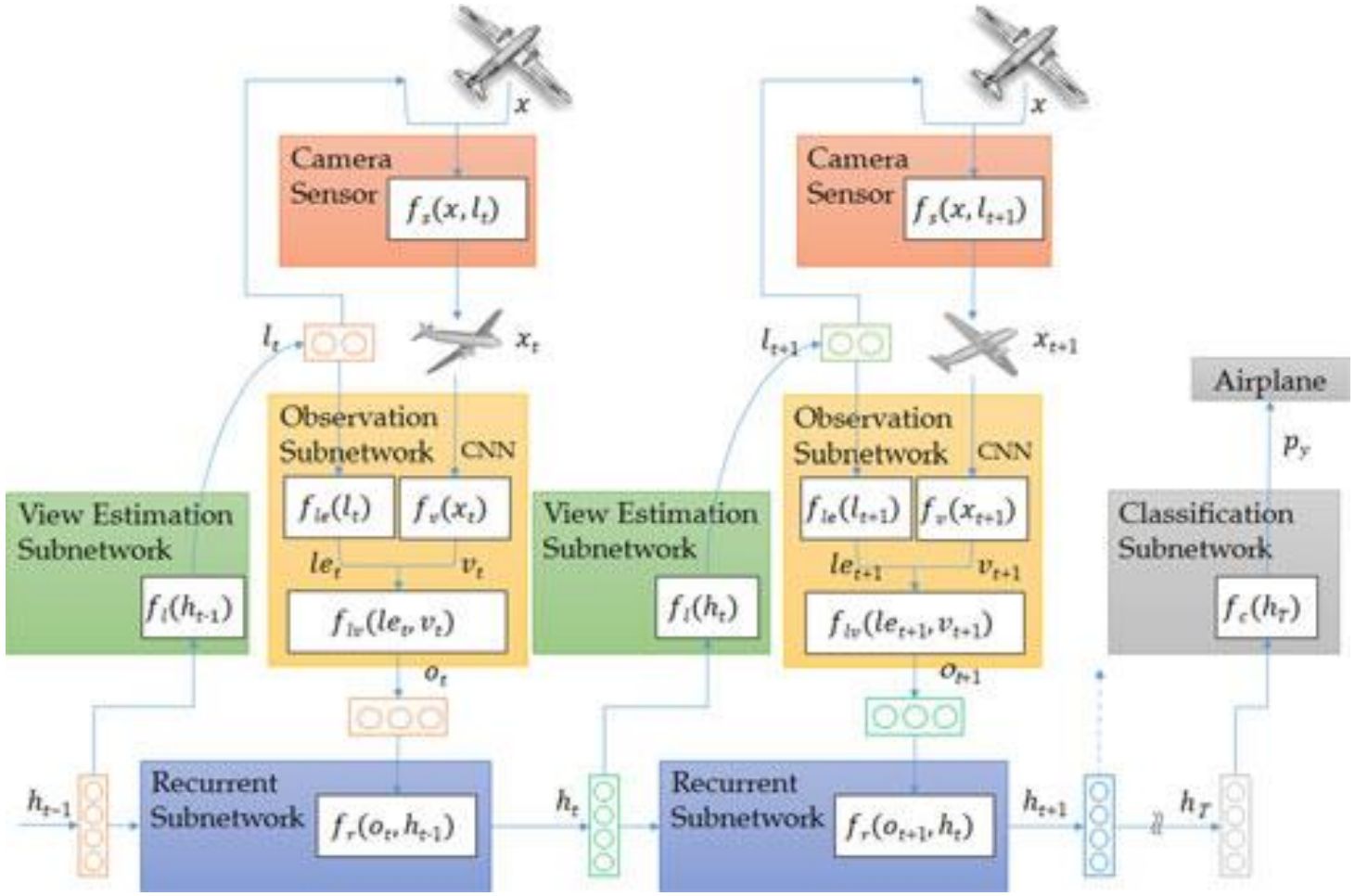


- **View estimation**: select current viewpoint l_t according to h_{t-1}
- **Observation**: obtain new 2D image x_t from l_t
- **Recurrent**: updates h_t with x_t and h_{t-1}
- **Classification**: predict category with final h_T

02

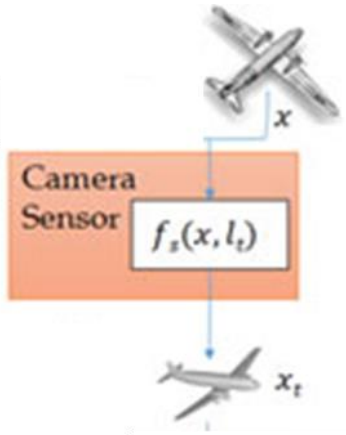
Methods

2.1 Architecture – 5 sub components



- Camera Sensor
- Observation Subnetwork
- Recurrent Subnetwork
- View Estimation Subnetwork
- Classification Subnetwork

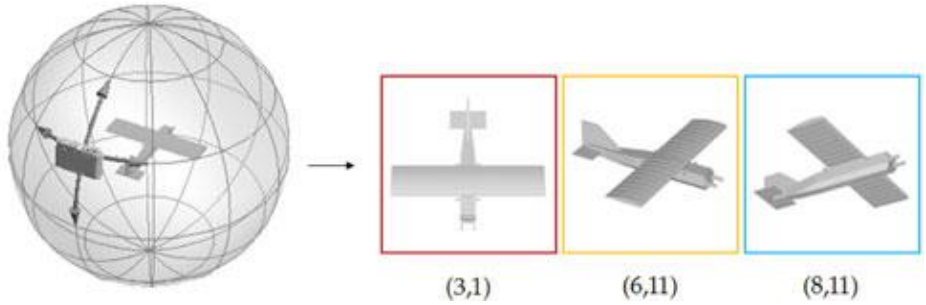
2.1.1 Camera Sensor: 3D to 2D



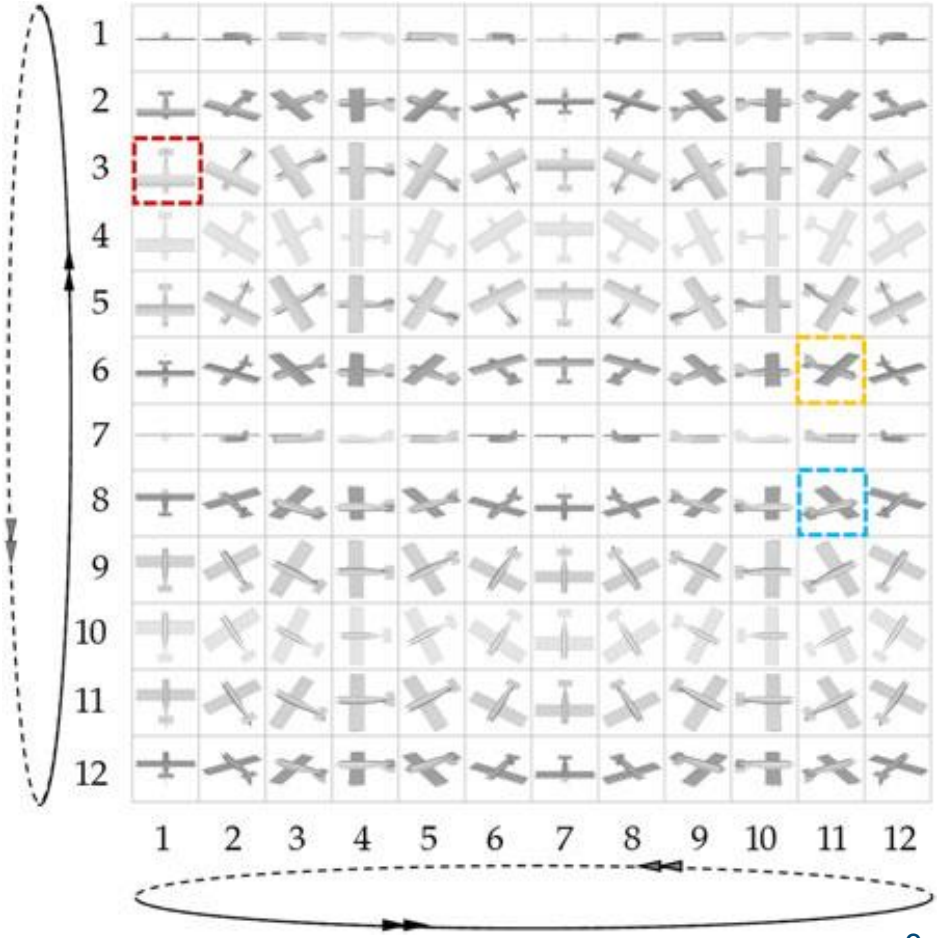
$$x_t = f_s(x, l_t)$$

x: 3D shape
f_s : renderer
l_t : camera location

non-differentiable process



view space

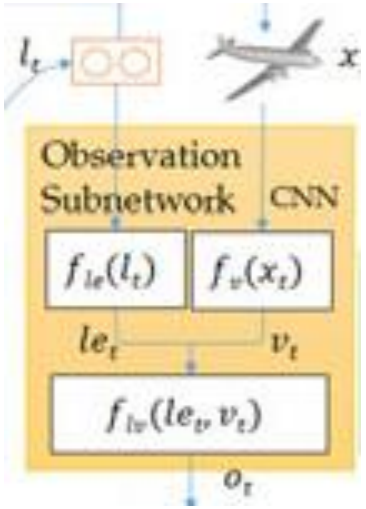


- sample views at every 30° both in <u>latitude and longitude</u>
- all views of a shape can be arranged in 12 × 12 grid

-> location l_t can be represented as

$l_t = (r_t, c_t)$, where $r_t, c_t \in [1, 12]$

2.1.2 Observation Subnetwork: encode location and visual information



$$o_t = f_o(l_t, x_t; \theta_o) = f_{lv}([f_{le}(l_t; \theta_{le}) \; f_v(x_t; \theta_v)]; \theta_{lv})$$

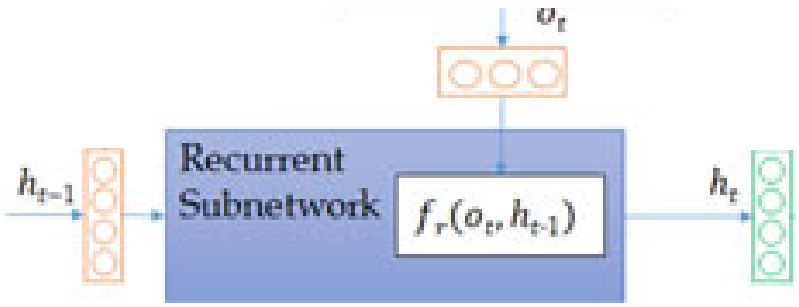
trained by **SGD**

x_t: 2D image
l_t: camera location
θ: parameters

- 3 parts of f_o :

- f_{le} : **fully connected layer**, embedding l_t to low-level features le_t
- f_v : **CNN**, extract high-level visual features v_t
- f_{lv} : **fully connected layer**, concatenate le_t with v_t

2.1.3 Recurrent Subnetwork: maintain internal state



$$h_t = f_r(o_t, h_{t-1}; \theta_r)$$

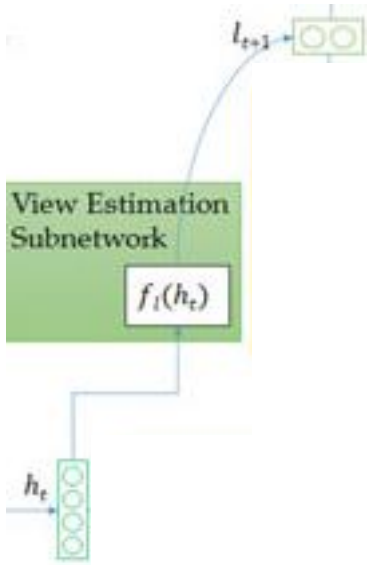
trained by **SGD**

o_t: extracted features
h_t: internal state
θ: parameters

- 2 kinds of f_r (both are **RNN**):

- Linear mapping: efficiency
- LSTM: able to learn long-range dependencies and stable dynamics

2.1.4 View Estimation Subnetwork: attention based view estimation

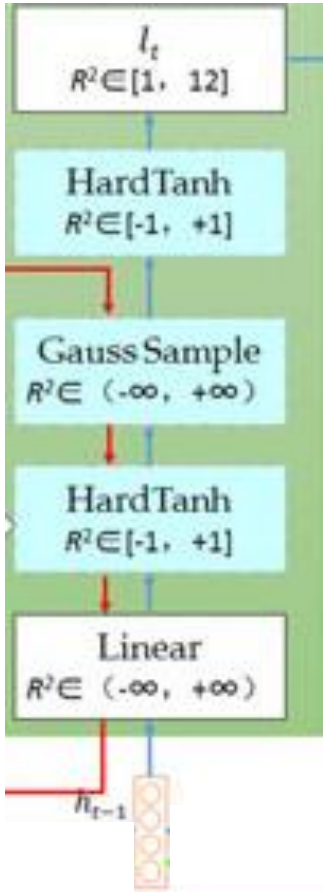


$$u_{t+1} = f_l(h_t; \theta_l)$$

$$l_{t+1} = f_g(u_t; \theta_g) = N(u_t, \sigma^2)$$

trained by **REINFORCE**

h_t: internal state
u_t: 2D coordinate tuple
θ: parameters

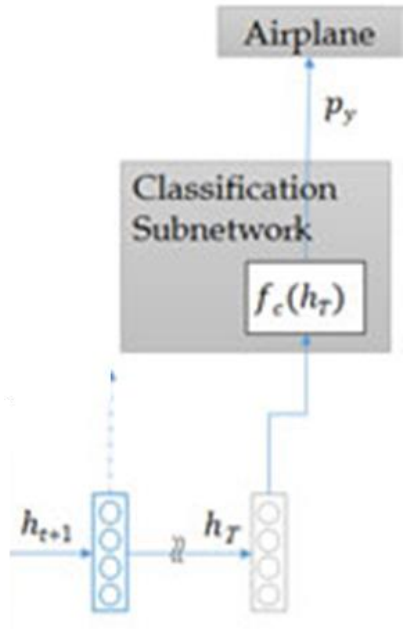


- 2 parts:

- f_l : **Linear layer + transfer function**(usually $HardTanh$ or $Eula\ldots$)
  maps internal state h_t into 2D coordinate tuple u_{t+1}

- f_g : **stochastic module**, samples l_{t+1} from Gaussian distribution,
  only used for RL in training (in testing: u_{t+1} is directly used as l_{t+1})

2.1.5 Classification Subnet



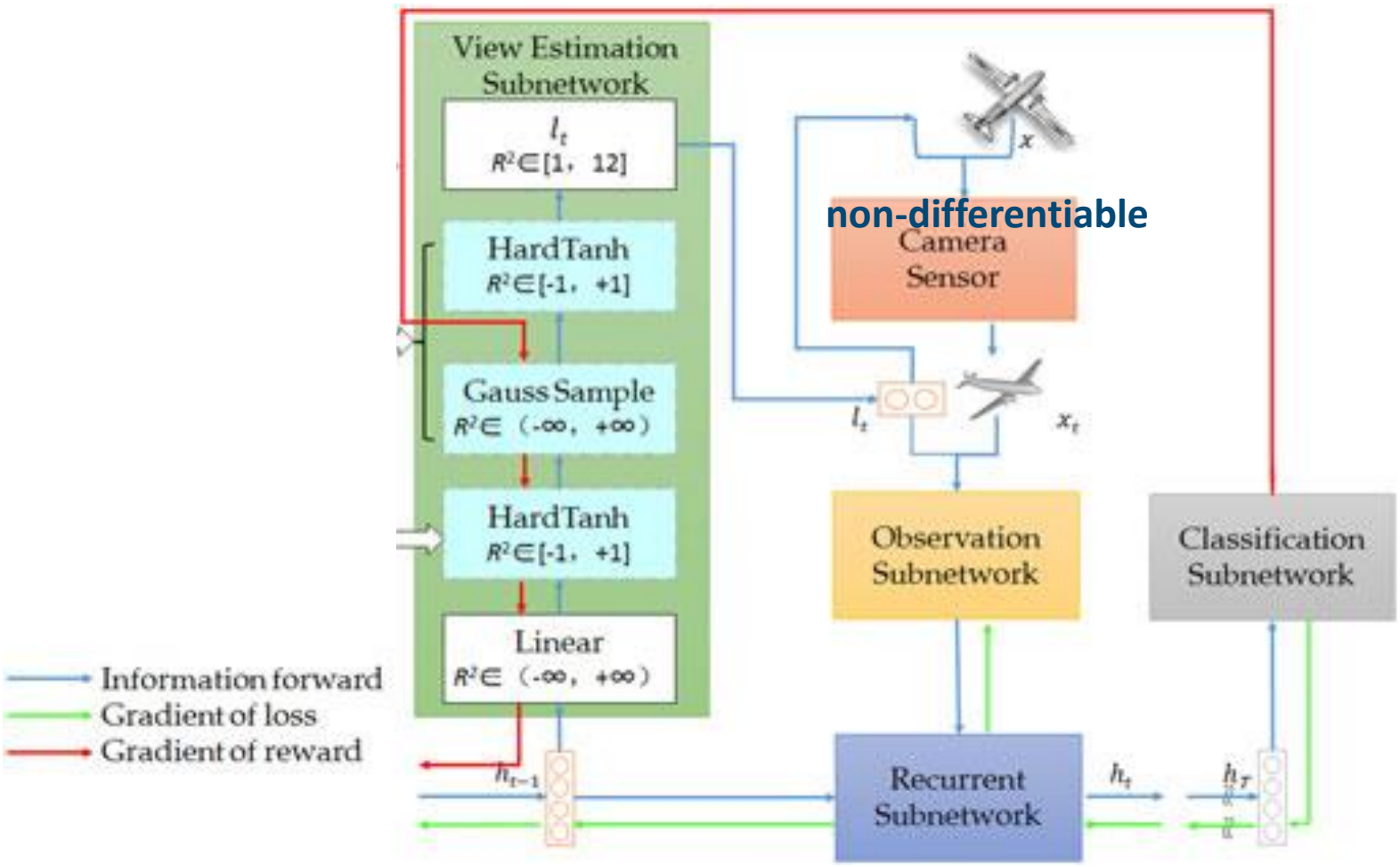
$$P(y|x) = f_c(h_T; \theta_c).$$

trained by **SGD**

h_T: final internal state
θ: parameters

- f_l : **fully connected layer** followed by $LogSoftMax$ layer

2.2 Overview of learning methods



- **SGD:**
- Classification Subnetwork
- Recurrent Subnetwork
- Observation Subnetwork
- **Non-differentiable**
- Camera Sensor $x_t = f_s(x, l_t)$

->

- **REINFORENCE:**
- View Estimation Subnetwork

2.2.1 REINFORCE based learning

View Estimation Subnetwork

$$u_{t+1} = f_l(h_t; \theta_l)$$

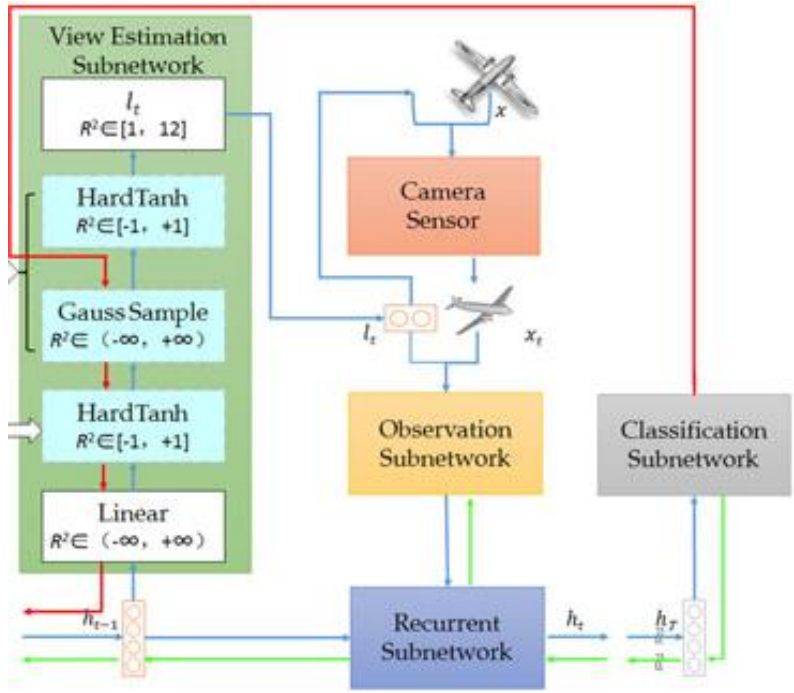
$$l_{t+1} = f_g(u_t; \theta_g) = N(u_t, \sigma^2)$$

h_t: internal state
u_t: 2D coordinate tuple
θ: parameters

- Approximate gradient using Monte Carlo sampling with $N(u_t, \sigma^2)$
- Policy $\pi_\theta = f_l(h_t; \theta_l)$

$$R = \begin{cases} 1, & classified\ correctly\ at\ T \\ 0, & otherwise \end{cases}$$

$$\frac{dR}{du_t} = (R - b) \times \frac{d\ ln(f(l_t, u_t))}{du_t}$$



- If classified correctly, encourage u_{t+1} to move to l_t
- otherwise, encourage u_{t+1} to move away from l_t

03

3 View-enhancement Strategies & Experiments

3.1 Enhancing the Information Flow of Gradient

- Common issue with these RNN-based attention models:

  unbalanced training of view estimation and classification subnetwork

- Classification Subnetwork is overfitted
- View Estimation Subnetwork poorly trained:

  estimated view l_t tend to get stuck at the boundary of view space

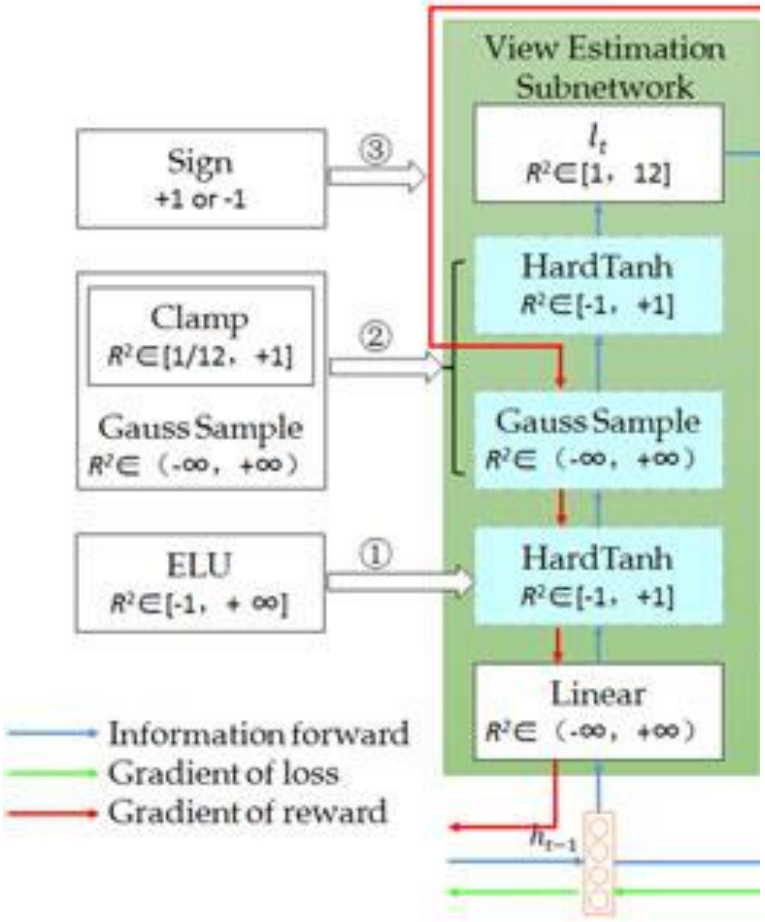
-> **3 critical schemes for enhancing the information flow of gradient**

to solve boundary issue

3.1.1 Enhancing the Information Flow of Gradient to solve boundary issue

$$u_{t+1} = f_l(h_t; \theta_l)$$

$$l_{t+1} = f_g(u_t; \theta_g) = N(u_t, \sigma^2)$$



- f_l is a Linear layer followed by **transfer function** ELU

  -> force u_{t+1} fall into the target range

- performs **Clamp** in Gaussian sample

  -> simulate the output of Gauss sample is always in the range of view space

- Add a **multiplication term** ***sign*** to the gradient of reward

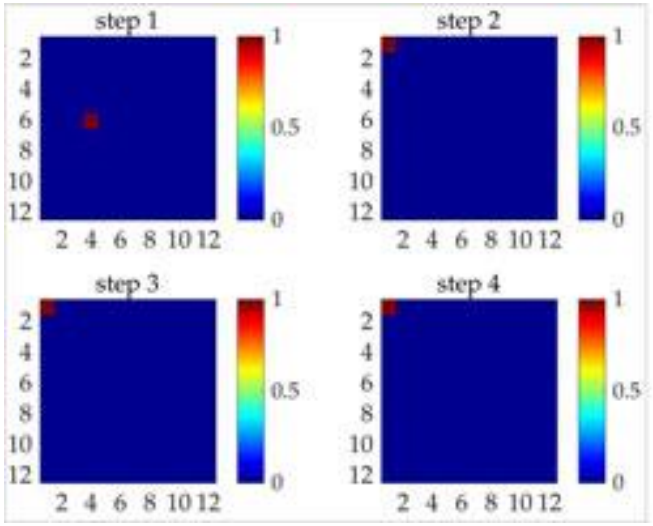
  -> avoid views always appearing at boundaries

$$sign = \begin{cases} +1, if\ l_t\ is\ not\ boundary \\ -1, if\ u_t < l_t, l_t = 1/12, left\ boundary \\ -1, if\ u_t > l_t, l_t = 1, \quad right\ boundary. \end{cases}$$

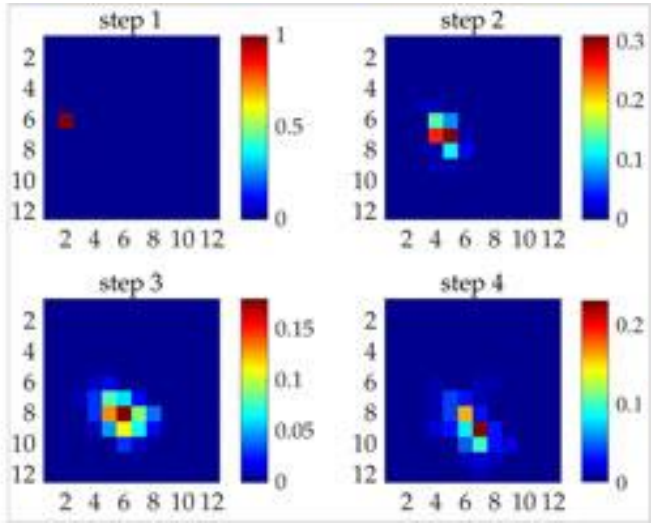
$$\Rightarrow \quad \frac{dR}{du_t} = sign \cdot (R - b) \times \frac{d\ ln(f(l_t, u_t))}{du_t}$$

3.1.2 Experiments of three critical schemes - BoundaryRAM

ClassicalRAM

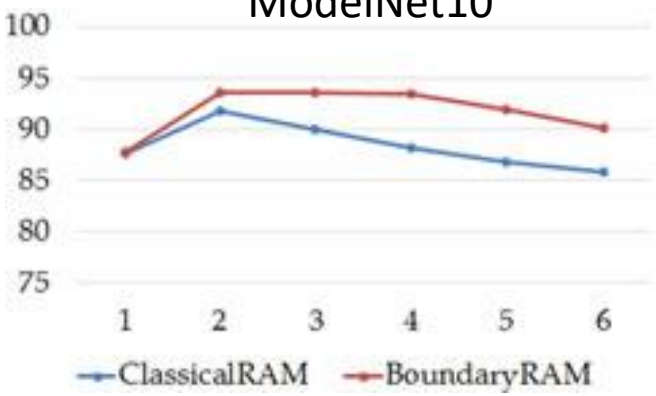


BoundaryRAM

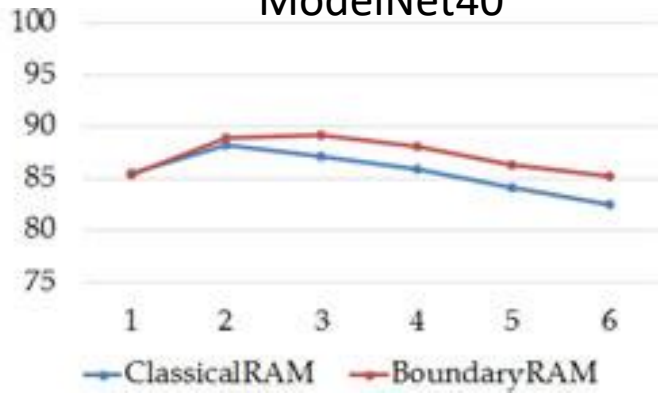


Heat maps of view frequency of predicating chair in the testing set of ModelNet10, time steps T = 4.

ModelNet10



ModelNet40



average class-level accuracy

- Views almost all locate at the boundary without the 3 schemes
- BoundaryRAM can effectively overcome the boundary issue and achieve a apparent performance enhancement.

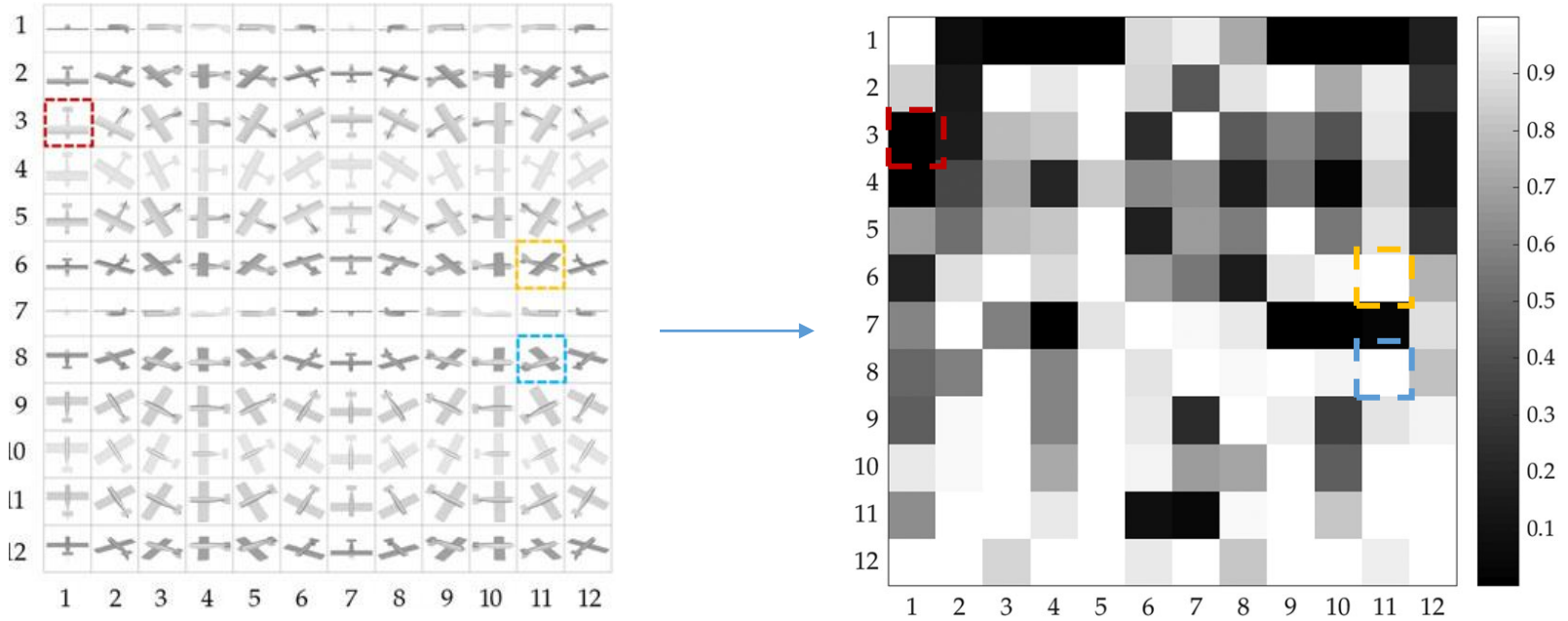
3.2 Learning with View Confidence

- 3 schemes for enhancing the information flow of gradient provides a basis for keeping a balance between the subnetworks by solving the boundary issue
- However, **classification subnetwork is still easier to train** than the view estimation subnetwork

  -> the learned policy is easily trapped in local optimization

  -> learning with **view confidence** for REINFORCE

3.2.1 View confidence for REINFORCE



l_t with high c_t can mean the view is close to the "best"

-> should encourage u_{t+1} to move to l_t

- for all training shapes, extract confidence c_t of image x_t:

- extract features v_t for each x_t of all views
- label x_t with the same category of 3D shape $\#categroies$
- input $(v_t, \#categroies)$ to the $LogSoftMax$ network

$$P(y_t|x_t) = LogSoftMax(Linear(v_t, \#categroies))$$

- output probability of $LogSoftMax$ corresponding to $\#categroies$ is c_t

3.2.1 View confidence for REINFORCE

- Previous gradient:

$$\frac{dR}{du_t} = sign \cdot (R - b) \times \frac{d\ ln(f(l_t, u_t))}{du_t}$$

  - If classified correctly, encourage u_{t+1} to move to l_t
  - otherwise, encourage u_{t+1} to move away from l_t

- Refresh gradient with c_t:

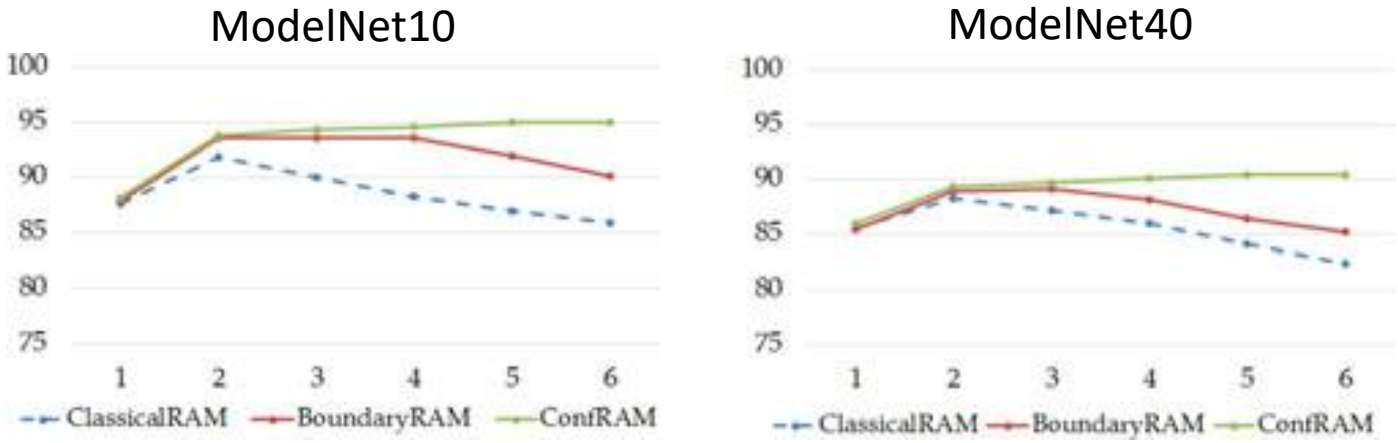
$$\frac{dR}{du_t} = sign \begin{cases} (R - b) \times \frac{d\ ln(f(l_t, u_t))}{du_t} \times c_t \\ (R - b) \times \frac{d\ ln(f(l_t, u_t))}{du_t} \times (1 - c_t) \end{cases}$$

l_t with high c_t can mean the view is close to the "best"

-> should encourage u_{t+1} to move to l_t

- If classified correctly but c_t is low, **weaken** encouraging u_{t+1} to move to l_t
- If classified wrongly but c_t is high, **weaken** encouraging u_{t+1} to move away from l_t

3.2.2 Experiments of View confidence for REINFORCE - ConfRAM



average class-level accuracy

- BoundaryRAM drops after T = 4 on ModelNet10
  after T = 3 on ModelNet40
- **ConfRAM can exploit more views** and converge at about T = 6 with significant higher performance

-> ConfRAM can achieve a stable and fairly good performance

3.3.1 Learning with View Location Constrains

- Learning with view confidence can achieve a stable and good performance
- However, the visited view of each time step may **overlap**

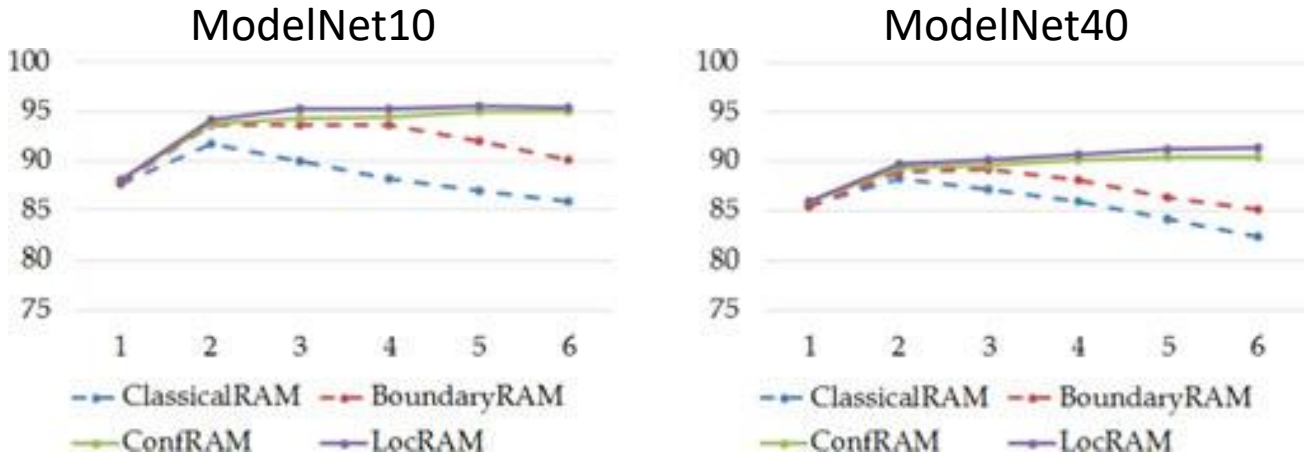
  -> Add regular term for view location constrains to loss

- Add pairwise distance layer for each two visited view l_i and l_j after T

  -> force the visited views to separate from each other

$$Loss(l_i, l_j) = max(0, 1/12 - PairDistance(l_i, l_j)).$$

3.3.2 Experiments of View Location Constrains – LocRAM



average class-level accuracy

- LocRAM equals to the whole VERAM
- Learning with the weak regular term of view location constrains **can obtain improvement**

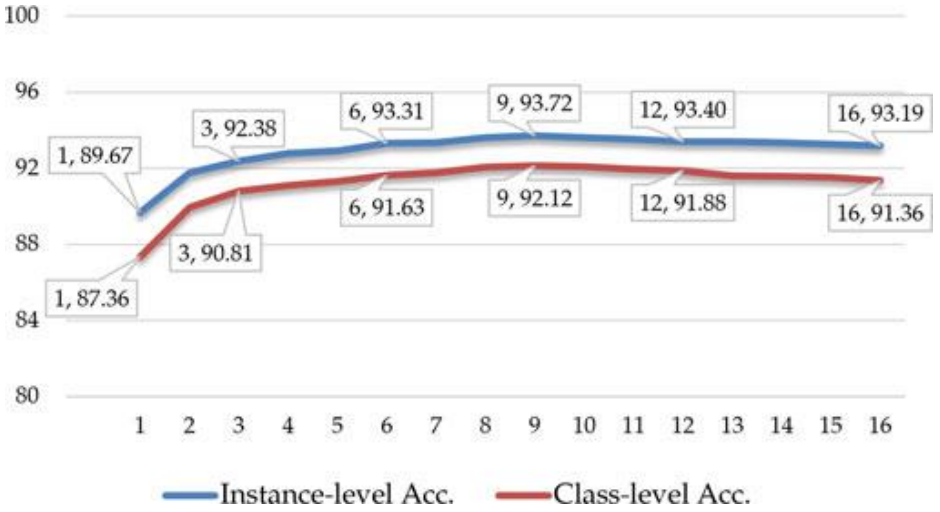
-> The regular term of view location constrains is only a weak constrain

3.4 Experiments of VERAM (AlexNet + LSTM)

Comparison of Classification Accuracy of Methods Based on Deep Neural Networks on ModelNet10 & 40

	Method	ModelNet10		ModelNet40	
		Inst.	Class	Inst.	Class
shape-based	3DShapeNets [26]	-	83.5	-	77.3
	VoxNet [27]	-	92.0	-	83.0
	SubVolSup [13]	-	-	89.2	86.0
	AniProbing [13]	-	-	89.9	85.6
	VRN [28]	93.61	-	91.33	-
	VRN-Ensemble [28]	97.14	-	95.54	-
	FPNN [29]	-	-	88.4	-
	PointNet [30]	-	-	89.2	86.2
	PointNet++ [31]	-	-	91.9	-
	O-CNN [32]	-	-	90.6	-
	So-Net [33]	95.7	95.5	93.4	90.8
view-based	DeepPano [34]	-	88.7	-	82.5
	PANORAMA-NN [35]	91.12	-	90.70	-
	PANORAMA-ENN [36]	96.85	-	95.56	-
	MVCNN [12]	-	-	-	90.1
	MVCNN-Alex [13]	-	-	92.0	89.7
	MVCNN-MultiRes [13]	-	-	93.8	91.4
	Pairwise [16]	94.0	-	92.0	-
	DomSetClust [14]	-	-	93.8	92.8
	RotationNet [17]	98.46	-	97.37	-
	VERAM	95.5	95.3	93.7	92.1
mix	FusionNet [48]	93.1	-	90.8	-

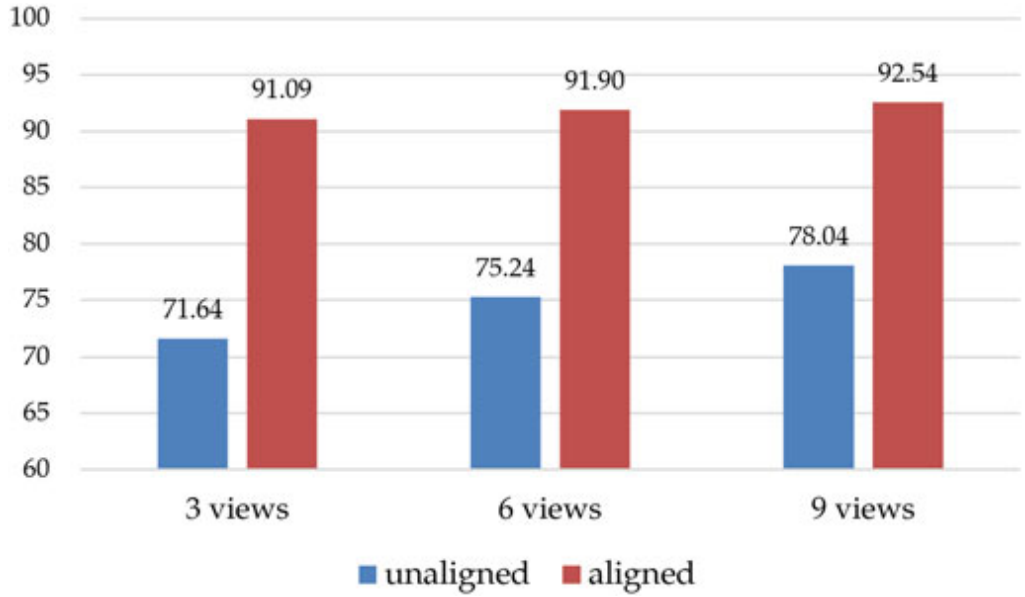
Best accuracy of VERAM on ModelNet40 with T from 1 to 16



- High accuracies
- Can quickly converge within a few time steps

3.4 Experiments of VERAM (AlexNet + LSTM)

Comparison of the instance-level accuracy of VERAM on unaligned and aligned test dataset of ModelNet40



- Trained 3 VERAM models (T = 3, 6, 9) with aligned training dataset

  Tested on both aligned and unaligned dataset
- Instance-level accuracy of unaligned shapes only got about 75 percent
- Only increases slightly with the increment number of views

-> VERAM is sensitive to the pre-defined viewpoints

04

Conclusion

4.1 Innovation Points

- VERAM is a recurrent attention model of actively selecting views for highly accurate 3D shape classification.
- To summarize, the **main contributions** of this work is to address the unbalanced training problem commonly found in RNN-based attention models:

1. enhance the information flow of gradient backpropagation for the view estimation subnetwork
2. design a highly informative reward function for the reinforcement training of view estimation
3. formulate a novel loss function that avoid view duplication.

4.2 Limitations

- VERAM is sensitive to **shape alignment**
- VERAM uses the **fixed time steps** for predication
- VERAM **omits the continuous images obtained when moving the camera** from the current selected viewpoint to the next one