Marking Scheme TEK5040/9040 H2021 Exam

1 Field of view (weight 12%)

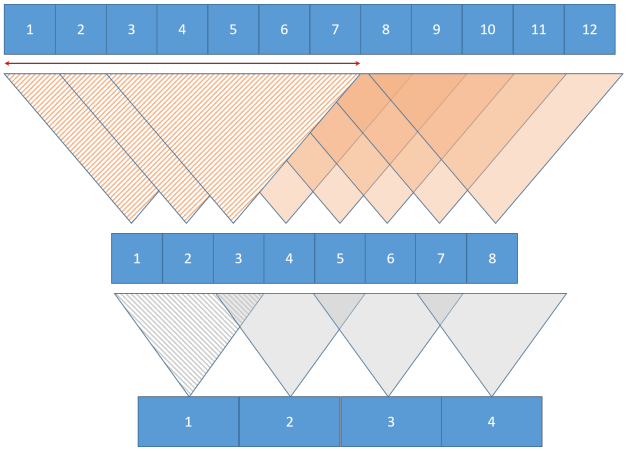

Figure 1: Field of view

Answer = 7×7 or 7

1.1 Marks

- **12 points** if the answer is correct
- If the answer is incorrect and there is illustration or description:
  - **4 points** if the first layer is correct
  - **4 points** if the second layer is correct
  - **2 points** if it is identified that the stride in the second layer is irrelevant to the answer.

2 Self-attention (weight 12%)

Disadvantages:

- Loss of positional information (permutation invariance) because of symmetry of the self-attention operation
- Computational complexity varies quadratically with the sequence length

Advantages:

- Parallelizability (equivalently maximum number of sequential operations needed is independent of input length)
- Input-to-output mapping goes through only one operation (equivalently maximum path length is independent of input length or all input elements are treated equally)

Remedies:

- Loss of positional information: Positional encoding (i.e. add external positional information to the input)
- Computational complexity: (Consider only a neighbourhood of each input element to calculate the self-attention)

2.1 Marks

- **4 points** to any disadvantage above or equivalent
- **4 points** to any advantage above or equivalent
- **4 points** to any remedy above or equivalent

3 Policy Gradients (weight 14%)

Examples of reinforcement algorithms that is based on the given formulation are: Trust Region Policy Optimization (TRPO) and Proximal Policy Optimization (PPO).

The proof is as follows:

$$\begin{aligned}
\nabla_\theta U(\theta) &= \nabla_\theta E_{\tau\sim\theta_{old}}[\frac{P(\tau|\theta)}{P(\tau|\theta_{old})}R(\tau)] \\
&= E_{\tau\sim\theta_{old}}[\nabla_\theta\left[\frac{P(\tau|\theta)}{P(\tau|\theta_{old})}R(\tau)\right]] \qquad \text{(step 1)} \\
&= E_{\tau\sim\theta_{old}}[\frac{\nabla_\theta P(\tau|\theta)}{P(\tau|\theta_{old})}R(\tau)] \qquad \text{(step 2)}
\end{aligned}$$

Here the second line follows from the fact that expectation is taken with respect to θ_{old} which is independent of θ, and the third line follows due to the fact that all terms except the $P(\tau|\theta)$ are independent of θ.
Now substitute using the log derivative trick

$$\nabla_\theta P(\tau|\theta) = \nabla_\theta \log(P(\tau|\theta))P(\tau|\theta)$$

we get

$$\begin{aligned}
\nabla_\theta U(\theta) &= E_{\tau\sim\theta_{old}}[\frac{\nabla_\theta P(\tau|\theta)}{P(\tau|\theta_{old})}R(\tau)] \\
&= E_{\tau\sim\theta_{old}}[\frac{\nabla_\theta \log(P(\tau|\theta))P(\tau|\theta)}{P(\tau|\theta_{old})}R(\tau)] \qquad \text{(step 3)}
\end{aligned}$$

Therefore

$$\begin{aligned}
\nabla_\theta U(\theta)|_{\theta_{old}} &= E_{\tau\sim\theta_{old}}[\frac{\nabla_\theta \log(P(\tau|\theta))|_{\theta_{old}}P(\tau|\theta_{old})}{P(\tau|\theta_{old})}R(\tau)] \qquad \text{(step 4)} \\
&= E_{\tau\sim\theta_{old}}[\nabla_\theta \log(P(\tau|\theta))|_{\theta_{old}}R(\tau)] \\
&= \nabla_\theta J(\theta)|_{\theta_{old}}
\end{aligned}$$

3.1 Marks

- **2 points** for mention of TRPO or PPO or other correct algorithm
- proof:
  - **6 points** for step 3
  - **2 points** each for steps 1,2, and 4 (or equivalents)

– Alternatively one can expand the expectation and prove that $J(\theta) = U(\theta)$ and hence the gradients. In this case **4 points** each for steps 1,2,and 3

$$\begin{aligned}
\nabla_\theta U(\theta) &= \nabla_\theta E_{\tau\sim\theta_{old}}[\frac{P(\tau|\theta)}{P(\tau|\theta_{old})}R(\tau)] \\
&= \nabla_\theta \int \frac{P(\tau|\theta)}{P(\tau|\theta_{old})}R(\tau)P(\tau|\theta_{old})d\tau \qquad \text{(step 1)} \\
&= \nabla_\theta \int P(\tau|\theta)R(\tau)d\tau \qquad \text{(step 2)} \\
&= \nabla_\theta E_{\tau\sim\theta}[R(\tau)] \qquad \text{(step 3)} \\
&= \nabla_\theta J(\theta)
\end{aligned}$$

4 Generative Adversarial Networks (weight 12%)

- In applying conditional GAN, we need paired data in the form of **(data_sample, condition)** and in this case the data_sample would be an optical image and the condition would be the corresponding sonar image. But we do not have paired data and hence cannot make sure that the generated data_sample (optical image) is corresponding to the condition (sonar image).
- Since we have unpaired data, we can use the CycleGAN. Denote a sonar image by I_s and an optical image by I_o. Then the main steps of the CycleGAN are:
  - In training, use two generators G and F and associated discriminators D_o and D_s
  - Learn G and D_o such that $G(I_s)$ is distributed as I_o
  - Learn F and D_s such that $F(I_o)$ is distributed as I_s
  - Learn with cycle-consistency loss, $F(G(I_s)) \approx I_s$ and $G(F(I_o)) \approx I_o$
  - In generation, use G on any given sonar image to generate the corresponding optical image.

4.1 Marks

- Conditional GAN: Total **6 points**
  - **3 points** for recognizing the dataset is unpaired
  - **3 points** for mentioning the core principle of conditional GAN i.e. condition=sonar image, data_sample= optical image.
- CycleGAN: Total **6 points**
  - **2 points** each for forward step, backward step and consistency loss

5 Learning Concepts (weight 12%)

a. (10%) Meta-Learning

Siamese networks are trained with data example pairs $(x_{support}(n), x_{query}(n))$ and their labels $(c_{support}(n), c_{query}(n))$ where the n^{th} support set $\mathcal{D}^{(n)}_{support} = (x_{support}(n), c_{support}(n))$ and the n^{th} query set $\mathcal{D}^{(n)}_{query} = (x_{query}(n), c_{query}(n))$. Note also that the meta-training set $\mathcal{D}_{train} = \{\mathcal{D}^{(n)}_{support} | n = 1, 2, \cdots, N\}$ and meta-test set $\mathcal{D}_{test} = \{\mathcal{D}^{(n)}_{query} | n = 1, 2, \cdots, M\}$

The network is trained to give a low similarity score when $c_{support}(n) \neq c_{query}(n)$ and a high similarity score otherwise. This score can be normalized and seen as the class probability of $x_{query}(n)$ in a binary

classification problem where the correct class is 1 when $c_{support}(n) = c_{query}(n)$ and 0 otherwise.

Siamese networks therefore use the given objective function, because in training it maximizes the conditional probability of the query set example class $\mathbf{y}$ = class of $x_{query}(n)$. Moreover, from the procedure it is clear that this probability is calculated conditioned on the support set examples $\mathcal{D}^{(n)}_{support} = (x_{support}(n), c_{support}(n))$ as well as the query set example input $\mathbf{x} = x_{query}(n)$. Therefore objective function of Siamese networks is the given conditional probability.

b. (2%) Self-supervised learning

Description:

- Pretext task: Task on which a network is pre-trained and labels can be automatically generated.
- Downstream task: Target task on which the network is fine-tuned and used.

5.1 Marks

- Meta-learning: **Total 10 points**
  - **4 points** for mention of data pairs from query and support sets
  - **2 points** for mention of maximization of similarity measure as the output probability
  - **4 points** for mention of query set class probability computed conditioned on support set and query set samples
- Self-supervised learning: **1 point** each for correct/reasonable definitions

6 Bayesian Deep Learning (weight 12%)

a. (4%) Bayesian Deep Learning

Bayes Formula:

$$p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{Y}) = \frac{p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w})p(\boldsymbol{w})}{\int p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w}}$$

or for the discrete case

$$P(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{Y}) = \frac{p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w})P(\boldsymbol{w})}{\sum_{\boldsymbol{w}} p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w})P(\boldsymbol{w})}$$

Denominator on the right hand side of the formula is difficult to evaluate when a deep network is used to represent $p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{w})$. The reasons for this is that

- Analytical evaluation of the integral/sum is often not possible, therefore numerical evaluation is necessary
- Numerical evaluation becomes intractable even for a moderate number of parameters N, because the number of different weight vectors increases exponentially with N.

b. (8%) Evidence Lower bound

There are two possible procedures:

- Black box variational inference
  - Differentiate $\mathcal{L}(\lambda)$ with respect to λ and get an expression for the gradient
  - Use log derivative trick to express the gradient as an expectation with respect to $q(\boldsymbol{w}, \lambda)$

- Draw S samples $\boldsymbol{w}^s,\ s = 1, 2, \cdots, S$ from $q(\boldsymbol{w}, \lambda)$ and evaluate the expectation above as the sample mean.

- Bayes by backprop
  - Draw S samples from a standard Gaussian distribution $\epsilon^s,\ s = 1, 2, \cdots, S$
  - Use re-parameterization trick to construct $\boldsymbol{w}^s = \boldsymbol{w}(\lambda, \epsilon^s),\ s = 1, 2, \cdots, S$, where $\boldsymbol{w}$ is a smooth differentiable function.
  - Estimate the ELBO $\hat{\mathcal{L}}(\lambda)$ as a sample mean using the constructed values $\boldsymbol{w}^s$
  - Differentiate the estimated ELBO $\hat{\mathcal{L}}(\lambda]$ with respect to λ

6.1 Marks

- Bayesian deep learning : Total **4 points**
  - **2 points** for the correct formula (either discrete or continuous)
  - **1 point** for pointing the denominator for the difficulty
  - **1 point** for mentioning exponential increase of different weight vectors (or equivalent)
- Evidence lower bound: Total **8 points**
  - Either BBVI or Bayes by backprop is acceptable
  - **2 points** for the correct order (sampling followed by differentiation or vice-versa)
  - **4 points** for mentioning log-derivative trick or re-parameterization trick
  - **2 points** for mentioning sample mean estimation of the expectation

7 Deep Learning for Control (weight 12%)

a. (4%) Inverse Reinforcement learning

Challenges:

- IRL is an ill defined problem (**2 points**)
- Expert demonstrations may not be drawn from an optimal policy (**2 points**)

b. (8%) Generative Adversarial Imitation Learning

Correspondence:

GAN	GAIL	**Score**
Generator (G)	Policy (P)	**1 point**
Discriminator (D)	Discriminator (D)	**1 point**
Real data x^{real}	Expert demonstration trajectories τ_e	**1 point**
Fake data x^{fake}	Policy generated trajectories τ_p	**1 point**
G maximizes the probability of x^{fake}	P maximizes the expected return $J(P(\tau_p))$	**2 points**
D maximizes the probability of x^{real}	D maximizes the probability $P(\tau_e)$	**1 point**
D minimizes the probability of x^{fake}	D minimizes the probability $P(\tau_p)$	**1 point**

Table 1:

The behaviour of GAIL discriminator (D) depends on how J varies with $P(\tau_p)$. If it is a monotonically increasing function, the behaviour of D is as given in the table. Otherwise, i.e. if it is monotonically decreasing, then D should behave in the opposite manner (i.e. D minimizes the probability $P(\tau_e)$ and maximizes the probability $P(\tau_p)$). This variant can also be accepted, even though not explained.

8 3D Processing and tracking(weight 14%)

a. (3%) 3D Processing

100 (**3 points**)

b. (3%) 3D Processing

This refers to the approach in Pointnet. The disadvantage of this approach is that it cannot extract locality information (equivalently it does not model interaction among points or topological information) (**3 points**)

c. (5%) Multi-object tracking

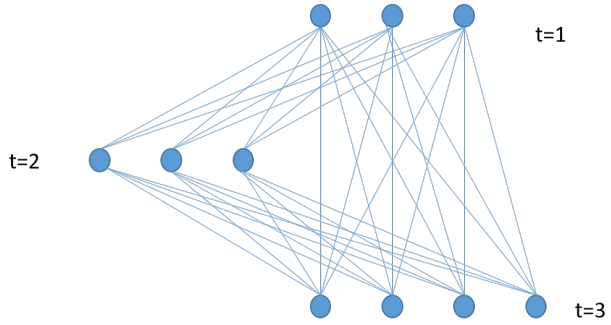

Figure 2: Association

Graph shown in Fig. 2 is given **1 point**.

- Initial node features: Send each detection through a CNN and extract features from an appropriately chosen layer (**1 point**)
- Initial edge features: Take the difference between the bounding box co-ordinates of the detections corresponding to the end nodes of the edge. This can be transformed using some function such as MLP if desired. (**1 point** for mentioning the bounding box difference)
- Starting with initial node and edge features, perform graph convolution operations (also known as message passing) for a predefined number of times (equivalent to number of layers of the graph neural network). Then flatten features at the last graph convolution operation to a vector. This vector is transformed to an output vector using a dense layer, representing the probabilities of each edge. If the probability of any given edge is above some threshold, then retain the edge, otherwise delete the edge. The retaining edges show the associations. (**2 points** for a reasonable description mentioning graph convolution and edge classification)

d. (3%) Single Object tracking

We need translation equivariance for the technique to work. Fully convolutional nets satisfy that. Equivalently locations of the score map can be traced back to the input image only if the network is fully convolutional. (**3 points** for any statement implying score map, input image correspondence)