Marking Scheme TEK5040/9040 H2020 Trial Exam

1 Stride in Convolutional Neural Networks (weight 12%)

$300x150$

2 Image Captioning (weight 12%)

We do teacher forcing during validation, the next word we predict is always based on a subsequence in the validation data. When generating captions we may go outside the distribution we have trained on. The validation loss does not measure such an effect.

3 Proximal Policy Optimization (weight 12%)

- Ignoring changes in state-visitation frequencies, i.e. replace $\rho_{\tilde{\pi}}$ with ρ_{π}.
- Approximation in the advantage estimation, $\hat{d_{\pi}} \approx d_{\pi}$.
- Approximation expectation by a finite number of samples.

4 Generative Adversarial Networks (weight 12%)

For each color image x we can construct a grayscale image x' by taking a weighted mean over the color channels. We then get a paired dataset $(x_1, x'_1), (x_2, x'_2), \ldots, (x_N, x'_N)$, where the grayscale image, x'_i, is the condition variable.

5 Learning Concepts (weight 14%)

This objective function indicates a separate meta learner. An example of this sort of meta-learning is the LSTM meta-learner. In this case, the learner is a neural network of parameters θ. The meta learner is an LSTM with parameters ϕ.

- In each iteration, input, label pairs from a training support set is fed to the learner. The parameters of the learner is provided by the meta learner.
- In a given iteration, the learner calculates the output, loss and gradient of the loss based on the input,label pair and the current parameters θ_t provided by the meta learner.
- Meta-learner generates a new parameter set θ_{t+1} based on the current loss and gradient of the loss.
- This is repeated for all the input-label pairs of the support set.
- Finally, an input,label pair from the query set is fed to the learner. Based on the latest parameter set provided by the meta-learner, it generates an output and a loss is calculated.
- The gradient of the loss is back-propagated through the meta-learner and parameters ϕ are updated.

The final loss for the optimization is based on input-label pair $\mathbf{x}, \mathbf{y}$ from the query set and it is defined to be the conditional probability $p_{\theta'}(\mathbf{y}|\mathbf{x})$. Further the probability is parameterized by the learner parameters which is provided by the meta-learner, i.e. $\theta' = g_{\phi}(\cdot, \cdot)$. Here $g_{\phi}(\cdot, \cdot)$ is the function approximated by the meta-learner. We can also see that the inputs to this function is the previous learner parameters θ and losses/gradients of losses which are in turn functions of support set samples. Therefore $\theta' = g_{\phi}(\theta, \mathcal{D}_{\text{support}})$.

Hence the optimization uses the given objective function.

Marks:

- identification of learner and meta-learner: 2 marks
- learner parameters predicted by the meta-learner: 4 marks
- Final loss calculation based on query set samples: 4 marks
- Explanation of objective and attach it to the procedure: 4 marks

6 Bayesian Deep Learning (weight 12%)

a. (4%) KL divergence contains a term of posterior distribution $p(\mathbf{w}|\mathcal{D})$ which we are going to estimate and hence unknown at this point. Alternatively, $p(\mathbf{w}|\mathcal{D})$ is typically intractable because of the normalization term in Bayes formula.

Equivalent statement: 4 marks

b. (8%) Use the log derivative trick

$$\nabla_\lambda q = q \nabla_\lambda \ln q$$

Hence the integral gets the form of integral of a function multiplied by a probability distribution. By definition of expectation we can write it in the given form.

Motivation for this conversion is to be able to estimate using the samples drawn from $q(\mathbf{w})$.

Marks:

- log derivative trick: 4 marks
- equivalent motivation: 4 marks

7 Deep Learning for Control (weight 12%)

a. (4%) Challenges in IRL:

- IRL is an ill defined problem (i.e. multiple reward function giving rise to the same expert demonstrations)
- Expert demonstrations are not guaranteed to be optimal

Marks: 2 marks for each reason

b. (8%) Entries in the following order:

- Policy (1 mark)
- Expert generated trajectories (0.5 marks)
- Policy generated trajectories (0.5 marks)
- Policy maximizes reward (assuming reward is defined as minus logarithm of discriminator assigned probability for the policy generated trajectories) (2 marks)
- Discriminator minimizes probability of expert generated trajectories (2 marks)
- Discriminator maximizes policy generated trajectories (2 marks)

8 3D Processing(weight 14%)

a. (8%) Pointnet segmentation steps:

- PointNet processes each point through parallel operations with MLPs (2 marks)
- Combine all MLP parallel output with max-pooling (or average pooling) over the processing paths (global feature) (1 mark)
- Concatenate the global feature with each output of the last MLP layer. (2 marks)
- Process the concatenated features through parallel operations with MLPs. The final MLP should output a vector of length c, where c is the number of segmentation classes. (2 marks)
- The maximum of each output vector gives the segmentation class of each point. (1 mark)

b. (6%) Graph convolution algorithm:

- for each point
  - for each neighbour
    * feed the own feature vector and the feature vector of the neighbor through the MLP
  - end
  - find the maximum of all MLP outputs and assign it to the point
- end

4 marks for the correct algorithm

Each point gets a 16 dimensional feature vector, therefore the resultant feature map is a 2D structure of dimension16xn. (2 marks)