Marking Scheme TEK5040/9040 H2020 Exam

1 Field of view (weight 12%)

9×9

2 Recurrent Neural Network attention (weight 12%)

Some potential issues:

- Computations no longer constant per step, but increases with sequeunce lenth. Thus computations no no longer scale linearly (but quardratically) with sequence length.
- For testing/inference we now need to store all previous states. Memory is thus no longer constant with respect to sequence length, but scales linearly.

One of the above is satisfactory, other answers may also be given some points.

3 Proximal Policy Optimization (weight 12%)

- `sample efficiency` Create dataset we use more than once.
- `stability` Don't encourage too much change, conservative update through clipping of reward function. (Also batching and sampling less correlated subset).

4 Generative Adversarial Networks (weight 12%)

Note: The methods below is not the only ways in which this can be done, but probably the ones closest to the methods we have discussed in this course.

Solution A (following hint in exercise): Each pair of ages forms two sets of unpaired images. E.g. all images of people aged 20 and all images of people aged 60. You could then train a CycleGAN model on these datasets. This could be done for all pairs of ages. In the application we then use current age and desired age to find the appropriate generator to use, and feed the image to this generator to get the desired output image. One might also divide into age groups, e.g. 20-25, 25-30, etc., to get more data per model we train, at the cost of coarser granularity.

Solution B (does not follow hint and we don't divide the datasets in subsets): Combine conditional GAN with CycleGAN. The generator now has two inputs

- image of person
- desired output age of person

and the generator should then try to output an image of a person of the desired age. The discriminator takes two inputs as well. The real samples are actual (image, age) pairs from our dataset, while the fake samples are (generated image, desired age) pairs where the generated image is the generated image from the generator. We then need to combine this with the cycle-consistency loss of Cycle-GAN. There are a few more details to decide upon, but that is the main idea and would be a sufficient answer. This method is a bit more advanced, but might perhaps work better in practice. Note that this approach doesn't need to take current age as input (though perhaps we could include this if we wanted as well). Also not that we could get away with not taking current age as input with the first approach as well if we e.g. trained a classifier on our labeled dataset to predict current age.

5 Learning Concepts (weight 12%)

Examples of meta-learning (few shot learning) techniques which make use of this objective function includes Siamese networks, matching network and relation network. Description of any of these techniques (or any other relevant technique) is acceptable.

We describe the matching network here:

Consider a support set and a query set in the training set. Assume k examples in the support set and a single example in the query set.

- The network has two sub-networks f and g
- f-network generates k feature vectors corresponding to the k examples in the support set
- g-network generates a feature vector for the example in the query set.
- cosine distance between the query feature vector and each support feature vector are calculated.
- These distances are normalized using their sum
- Each normalized sum is treated as the probability that the class of the query set example equals the class of the concerned support set example
- In training, adjust network parameters to maximize the probability of the query set example as calculated above.
- In testing, pick the class of the support example which has the highest probability (as calculated above) to be the class of the query set example.

Matching network uses the given objective function, because in training it maximizes the conditional probability of the query set example class $\mathbf{y}$. Moreover, from the procedure it is clear that this probability is calculated based on all examples in the support set $\mathcal{D}_{support}$ as well as the query set example input $\mathbf{x}$. Therefore objective function is the given conditional probability.

6 Bayesian Deep Learning (weight 12%)

a. (4%) Reasons:

- KL distance includes $p(\mathbf{w}|\mathcal{D})$ which is the goal of the computation and it is not available. Alternatively $p(\mathbf{w}|\mathcal{D})$ is difficult to calculate (intractable) because of the normalization term in the Bayes formula: (1 mark)
- ELBO includes $p(\mathbf{w}, \mathcal{D})$ which can be written as $p(\mathcal{D}|\mathbf{w})p(\mathbf{w})$. Both of those can be calculated relatively easily, because $p(\mathbf{w})$ is a prior which is assumed and $p(\mathcal{D}|\mathbf{w})$ is the result of the forward pass through the network (3 marks)

b. (8%) Sampling from $q(\mathbf{w}, \lambda)$ results in a non smooth (non differentiable) relationship between $\mathbf{w}^s$ and λ. Since ELBO is expressed in terms of $\mathbf{w}^s$, the relationship between ELBO and λ becomes non-differentiable. Hence Paul cannot find the gradient of ELBO with respect to λ and optimize it.

However, Paul could use the re-parameterization trick and make it differentiable. In this approach for example, sample from a unit Gaussian to get ϵ^s and construct the parameter samples using a deterministic function $\mathbf{w}^s = \mathbf{w}(\lambda, \epsilon^s) = \mu + \sigma\epsilon^s$ where $\lambda = (\mu.\sigma)$.

Alternatively Paul could differentiate ELBO before sampling and use the log derivative trick.

7 Deep Learning for Control (weight 12%)

- Compounding errors: System needs to be trained with more exploration. Use techniques like Dagger or reinforcement learning. However, these techniques require more interaction with the environments with non mature policies something which increases the risk of catastrophic events. Alternatively, use tricks such as those in NVIDIA self-driving system, but success is limited with such techniques. (2 marks for how to address the issue and 2 marks for their limitations)
- Complex reward functions: System can learn the reward functions using inverse reinforcement learning (IRL) based on expert demonstrations. However, IRL is an ill defined problem and expert demonstrations can be non optimal. (2 marks for IRL, 2 marks for the problems of the IRL)
- High risk in exploration: Exploration can be limited to known good policies or use simulators. However, limited exploration will give rise to problems such as compounding errors and it is hard to develop realistic simulators something which can lead to reality gap. (2 marks for how to address the issue and 2 marks for limitations)

8 3D Processing(weight 16%)

a. (3%) In regular convolutions we do not consider strides in the channel dimension (strides in only height and width dimensions). In 3D-convolutions however, we consider strides in all three dimensions. (3 Marks)

b. (3%) A 256 dimensional vector for each of the 100 points means that the dimensions of the output is 100x256. Specify in the other way around, (i.e. 256x100) is also acceptable (3 marks)

c. (3%) Student needs a symmetric operation across the points. Max pooling, average pooling or any other symmetric operation across points can be used. (3 Marks)

d. (3%) Student can add a spatial transformer network operating on each point. (3 points)

e. (4%) Student can use a 3D convolution network (3D CNN) to form a siamese network. The basic steps are:

   - Prepare point cloud pairs containing similar and dissimilar images
   - Feed each pair through the 3D CNN and generate two 3D feature maps.
   - Perform a 3D convolution between the feature maps to generate a 3D score map.
   - Generate a loss function based on the score map and train the 3D CNN to minimize the loss.