1 Convolutional layers (weight 6%)

Assume you have an input image x and consecutively apply two convolutional layers, both with kernel size $[3, 3]$ and stride 1 in both height and width dimension. We define the *field of view* of a neuron as the pixels in the image that may affect the output of the neuron. The neurons in the first layer thus have a field of view of size 3×3. What is the size of the field of view for a neuron in the second convolutional layer?

Solution: 5×5

Comments:

2 Optimization (weight 6%)

Why do we often prefer *stochastic* gradient descent, i.e. estimate the gradient on a mini-batch, rather than calculate the gradient on the whole dataset?

Solution: For a large number of samples, calculating the full gradient may become very expensive. Calculating a crude estimate based on a small fraction may give us a good enough approximation, and there are many other *descent directions*, other than the one of steepest descent, that may be good directions to move in. How large the batch size should be is a trade-off between the compute time for the estimate, and the accuracy of the estimate.

Comments:

- 3 points for mentioning computational cost
- 3 points for mentioning that gradient is correct *on average*, or that as long as we get a *descent*-direction we might be fine. Failing to mention this, but instead mentioning other commonly used arguments like avoiding *local minima* or *regularization effect* gives 1.5 points.

3 Bidirectonal RNN (weight 6%)

One common way to include future context when making a prediction for RNNs is to *delay* the predictions a certation number of time steps. What is a possible disadvantage of this compared to bidirectional RNNs?

Solution: Having a *fixed* number of delay steps has the disadvantage of figuring out the *right* number of steps. More context may always seem beneficial, but it comes at the cost of the RNN having to store information relevant for a prediction longer. Also, we do not necessarily need the same amount of context for each prediction, may be hard to find a delay time that works reasonably for all circumstances

Comments:

- Mentioning issue with *fixed* number of time steps gives full mark. Only mentioning that it may not get all relevant context gives 3 points (as this could be solved by making the delay arbitrarily large).

4 External memory (weight 6%)

a. (3%%) Do one bit memory cells make sense when using *location*-based addressing? Give a brief explanation.

Solution: Yes, the program itself is responsible for the interpretation of the content in the memory cell, so it would know how to interpret a zero versus a one value.

Comments:

- It is also acceptable to say e.g. ”Yes, as we use the index to do the lookup”.

b. (3%%) Do one bit memory cells make sense when using *content*-based addressing? Give a brief explanation.

Solution: No, we can’t in any meaningful way perform a query against one bit memory cells. Recall that with content-based addressing each memory cell needs to be unique to add information. *self-contained.*

Comments: In addition to the argument above, it is acceptable to come with the following arguments:

- With content-based addressing, the memory cells need to be *self-contained.*
- We can’t meaningfully perform any query with one-bit memory cells.

5 RL reward shifting (weight 6%)

If we shift all intermediate rewards by a constant, e.g. subtract 1, may that change the optimal policy? Does this depend on if you are in a continual or episodic environment?

Solution: For episodic environments the solution may be altered. If we subtract a negative constant to the rewards we might get an agent that tries to finish the episode faster. For continual environments the solution will not change, as the agent cannot affect this reward through its actions in any way.

Comments: Full score if mentions that it may affect agents ”will to live”.

6 Data efficiency (weight 6%)

Some machine learning tasks typically needs more data than others, in order to perform well. Order the task of *image classification*, *object detection* and *pixel segmentation* in terms of how many training images you typically need. Why are some of these tasks often more data efficient, even on the same image material?

Solution: Ordered from many to fewer

1. Image classification

2. Object detection

3. Pixel segmentation

Scoring: Correct ordering give 3%
More specific labels are more data efficient.
Answers related to specific labels get 3%. Answers related to typical homogeneity/heterogeneity in the typical data sets get 1%, as it is true for some (common) examples.

7 Tracking in video (weight 10%)

a. (5%%) A siamese network can be used for object tracking in video. Describe a method for tackling changes in depth/distance for a siamese tracking network.

Solution: Either running the networks with multiple image scales or using a region-proposal (RPN) branch, are tested approaches.
4% for mentioning one of these approaches.

Full score (5%) for those either mentioning that the multi-scale approach has to scale the bounding box appropriately or maintaining a reference size.

or

Full score (5%) for mentioning that the RPN network create the convolutional kernel from processing the fingerprint-image.

b. (5%%) Why can a siamese network be a simple and efficient way of tracking *multiple* objects in a video?

Solution: Explanations mentioning any of these give full score:

- You only need run the entire network *once* for each input image.
- No runtime training

8 Graph-convolutions (weight 10%)

Graph convolutions are often used for 3D segmentation. Parametric kernels are one way to do graph-convolutions.

a. (5%%) How do you get the weights for your filter in a parametric kernel convolution? Give two examples of possible kernels.

Solution: 2% for answers mentioning that you get the weights from the output from functions.

2% for mentioning that you use input position as input to the functions **or** that you train the parameters of the function.

1% for providing somewhat logical kernels functions.

Typical kernels:

- Gaussian
- Linear
- Neural network.

b. (5%%) What is the main benefit of a graph-convolutional operator compared to a standard convolutional operator.

Solution: Full score (5%) for mentioning any of the following. Graph-convolutional operators can:

- work on arbitrary graphs
- be more effective on point-clouds
- use higher resolution input
- avoid errors with voxelization

9 Generative Adverserial Network

a. (5%%) Sketch and briefly explain the general architecture of a Generative Adverserial Network that generates images of numbers from 1 to 10.

Solution: 3% for drawing or explaining GAN with: random z-vector, generator network and discriminator network.

1% for drawing/describing that the discriminator both gets generated and real images as input and the task of the discriminator.

1% for some explanation of how the networks are trained jointly, either:

- That the generator and discriminator are competing e.g. minimizing and maximizing.
- examples of loss functions.
- flipping of gradient between discriminator and generator.

b. (5%%) Explain how you would modify the architecture, so you can control which number are generated at each run.

Solution: 3% for mentioning that you input the class-label as part of the z-vector.

2% for mentioning that you also need to give the discriminator access to the class-label or that a branch in the discriminator tries to estimate the class label (InfoGAN).

10 Bayesian deep learning (weight 11%)

a. (5%) Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ respectively be a sequence of input vectors and the corresponding outputs of a neural network parameterized by $\boldsymbol{w}$. Assume that the neural network models the likelihood distribution $p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{w})$ and that the prior distribution $p(\boldsymbol{w})$ is known. In Bayesian approach we try to estimate the posterior distribution $p(\boldsymbol{w}|\boldsymbol{Y}, \boldsymbol{X})$. Write an equation for the posterior distribution in terms of the likelihood and prior distributions.

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

Comments:

- Full marks given to correct equation, otherwise zero marks.
- Either integral or summation is acceptable in the denominator.
- Possibly dropped $\boldsymbol{X}$ is also acceptable

b. (3%) Explain why it is difficult to perform exact Bayesian inference in practical deep learning.

Solution: Complexity of computation of the denominator of the expression above increases exponentially with the length of the vector $\boldsymbol{w}$. Therefore, this computation is intractable for practical deep neural networks whose number of parameters (same as the length of the vector $\boldsymbol{w}$) is typically greater than couple of hundreds.

Comments:

- Identify that denominator is the problem draws 2%.
- Indicating in some way that the complexity increases exponentially draws 1%.

c. (3%) Explain how the Bayesian approach can be utilized in safety critical applications such as medical diagnosis, self-driving cars and military applications

Solution: With Bayesian approach one can calculate the variance of the output of a neural network for a given input, because distributions of the network parameters are learned rather than point estimates. Output variance is an indication of the reliability of the network output and therefore suitable for safety critical applications.

Comments:

- Indicating that output variance can be calculated draws full marks.
- Mentioning only 'uncertainty' is given 1%. Elaboration of uncertainty measure, for example, variance of the output is interpreted as uncertainty, is needed for full score.

11 Deep learning for control (weight 12%)

a. (3%) Briefly explain two differences between control policy learning and perception learning.

Solution:

- In perception learning we can assume that input data samples are independently distributed, whereas in control policy learning input samples are correlated.
- In perception learning, it is easier to obtain a direct supervision signal, whereas in control policy learning supervision signal is often indirect.
- In perception learning, it is easier to collect a data set prior to training, while in control policy learning, training and data collection are often intertwined.

Comments:

- For any of the key concepts *independent data, direct supervision, offline data collection* indicating anyway in a reasonable explanation, assign 1.5%.

b. (3%) Guided Cost Learning (GCL) is a sampling based maximum entropy inverse reinforcement learning algorithm. Why is it more suited for systems with larger state-spaces than dynamic programming based maximum entropy inverse reinforcement learning?

Solution: Dynamic programming approach requires looping over all the states which is computationally expensive if the state space is large. GCL avoids this by taking samples of the trajectories through the state space.

Comments:
Indicating any of the following or equivalent draws full marks.

- Dynamic programming approach requires looping over whole state space.
- Dynamic programming becomes intractable/computationally very costly for large state spaces.
- Through sampling one can avoid traversing the whole state space.

c. (3%) GCL can be seen as a form of Generative Adversarial Network (GAN) which is typically characterized by a *generator, a discriminator, image data samples* and a *real image data samples.* What are the corresponding quantities of GCL?

Solution:

- generator=policy
- discriminator=cost function or reward function
- image data samples= state-action trajectories
- real image data samples= expert demonstrations or expert generated state-action trajectories

Comments:

- 1% for each of the first two items above and 0.5% for each of the last two items. Any reasonable wording is accepted.

d. (3%) State the main difference between GCL and Generative Adversarial Imitation Learning (GAIL).

Solution: In GCL the "discriminator" generates the costs (or rewards), so it is not a strict classifier. In GAIL, the discriminator classifies the trajectories into two classes: real and fake.

Comments:
Any reasonable wording that indicates the above draws full marks.(Eg: GCL learns reward function, GAIL not)

12 Sequence modeling (weight 11%)

a. (2%) Briefly describe the phenomenon known as *exposure bias* in sequence generation training tasks such as machine translation and abstractive summarization.

Solution: In training in such tasks, the ground truths are used to generate the next word at each time step. However, in testing, the system generated word in the previous time step is used to generate the next word. This difference in testing and training is referred to as exposure bias.

Comments:
Any explanation indicating the above difference in testing and training is given full marks. If only testing or training is mentioned assign 1%.

b. (3%) Why is reinforcement learning a solution to the problem of exposure bias?

Solution: If the problem is approached with reinforcement learning, in training one can sample a word from the previous time step (instead of using the ground truth) to generate a word at the current time step. In this way the setup in testing and training would be similar.

Comments:
If it is mentioned that words are sampled in training, assign full marks.

c. (3%) A machine translation system based on recurrent neural networks is trained using reinforcement learning with the BLEU evaluation metric. Identify *agent, state, policy, action* and *reward* of the associated reinforcement learning system.

Solution:

- agent = Recurrent neural network (RNN)
- state = Internal state (including hidden states, attention weights and context vector) of the RNN.
- policy= A function mapping the current state to the generated word. This is parameterized by the RNN parameters.
- action= generated word
- reward= BLEU score

Comments:
Correct identification of reward is given 1%. Each of the correctly identified other entities are assigned 0.5%.

d. (3%) In reinforcement learning based sequence generation task, we aim at maximizing the expected reward $L(\theta) = \sum_{\boldsymbol{w}} p_\theta(\boldsymbol{w}) r(\boldsymbol{w})$, where $\boldsymbol{w}$ is the generated word sequence, θ is the model parameters and $r(\boldsymbol{w})$ is the total reward for the word sequence $\boldsymbol{w}$. The gradient of $L(\theta)$ can be shown to be

$$\nabla_\theta L(\theta) = \sum_{\boldsymbol{w}} p_\theta(\boldsymbol{w}) r(\boldsymbol{w}) \nabla_\theta \log p_\theta(\boldsymbol{w})$$

This quantity can be approximated with the sample mean. Write an expression for this approximated gradient assuming that N samples of word strings $\boldsymbol{w}^s, \quad s = 1, 2, \ldots, N$ are drawn from the sequence generator.

Solution:

$$\nabla_\theta L(\theta) \approx \frac{1}{N} \sum_{s=1}^{N} r(\boldsymbol{w}^s) \nabla_\theta \log p_\theta(\boldsymbol{w}^s)$$

Comments:
Any semantically similar expression draws full marks. If division by N is missing assign 2%. Any other expression hinting at taking average of the reward times log gradient draws 2%.