This is a closed-book test. It is marked out of 35 marks. Please answer ALL of the questions. Here is some advice:

- The questions are NOT arranged in order of difficulty, so you should attempt every question.

- Questions that ask you to “justify your answer” or “briefly explain” something only require short (1-3 sentence) explanations. Don’t write a full page of text. We’re just looking for the main idea.

- None of the questions require long derivations. If you find yourself plugging through lots of equations, consider giving less detail or moving on to the next question.

- Many questions have more than one right answer.
Final mark: ______ / 35
1. [2pts] Recall that multilayer perceptrons are universal for the set of functions mapping binary-valued input vectors to binary valued outputs.

   (a) [1pt] What do we mean by universal?

   (b) [1pt] If multilayer perceptrons are universal, why do we still consider other architectures?

2. [1pt] Give an example of a data augmentation technique that would be useful for classifying images of cats vs. dogs, but not for classifying handwritten digits. Briefly explain your answer.

3. [1pt] Suppose we have a grayscale image represented as an array, where larger values denote lighter pixels. What is the effect when we convolve it with the following kernel?

\[
\begin{pmatrix}
0 & -1 & 0 \\
-1 & -4 & -1 \\
0 & -1 & 0
\end{pmatrix}
\]
4. **2pts** The learning rate is an important parameter for gradient descent.

   (a) **1pt** Briefly describe something that can go wrong if we choose too high a learning rate for *full batch* gradient descent.

   (b) **1pt** Briefly describe something that can go wrong if we choose too high a learning rate for *stochastic* gradient descent, but is not a problem in the full batch setting.

5. **2pts** Suppose we are training an RNN language model using teacher forcing (the method you implemented in Assignment 3).

   (a) **1pt** What are the inputs to the network at training time?

   (b) **1pt** What are the inputs to the network at test time?
6. [1pt] Here is a modified version of code from Programming Assignment 2. The methods `downconv1`, `rfconv`, etc. implement convolution layers. Add edges to the diagram to represent the network architecture this implements. You don’t need to justify your answer.

```python
class MyNet(nn.Module):
    ...

def forward(self, x):
    self.h1 = self.downconv1(x)
    self.h2 = self.downconv2(self.h1)
    self.h3 = self.rfconv(self.h2)
    self.h4 = self.upconv1(torch.cat([self.h3, self.h2], 1))
    self.h5 = self.upconv2(self.h4)
    self.out = self.finalconv(torch.cat([self.h5, x], 1))
    return self.out
```

![Diagram of network architecture](image-url)
7. [3pts] Recall the (multivariate) linear regression model:

\[ y = w^T x + b \]

\[ \mathcal{L}(y, t) = \frac{1}{2} (y - t)^2 \]

Your job is to implement full batch gradient descent in NumPy. In particular, suppose we are given an \( N \times D \) NumPy array \( X \) representing all the training inputs, and an \( N \) dimensional NumPy vector \( t \) representing the targets, where \( N \) is the number of data points, and \( D \) is the input dimension. The weights are represented with a \( D \)-dimensional NumPy vector \( w \), and the biases are represented with a scalar \( b \). The learning rate is given as \( \alpha \).

Write NumPy code which implements one iteration of batch gradient descent. It should be vectorized, i.e. it should not involve a \texttt{for}-loop. You don’t need to show your work, but doing so may help you get partial credit.
8. **[2pts]** Suppose we have a convolution layer which takes as input an array \( \mathbf{x} = (x_1 \ x_2 \ x_3) \) and convolves \( \mathbf{x} \) with the kernel \((2 \ -1)\). This layer has a linear activation function. The output is an array of length 4.

Now let’s design a fully connected layer which computes the same function. It has a linear activation function and no bias, so it computes \( y = W \mathbf{x} \), where the output \( y \) is a vector of length 4. Give the \( 4 \times 3 \) weight matrix \( W \) which makes this fully connected layer equivalent to the convolution layer above. You don’t need to justify your answer, but doing so may help you get partial credit.

*Hint: first write the values of each output as a linear function of the inputs.* To help you check your work, if \( \mathbf{x} = (1 \ 2 \ 3) \), your answer should give \( y = (2 \ 3 \ 4 \ -3) \).

9. **[1pt]** Briefly explain one flaw of encoder-decoder architectures for machine translation which do not use attention, and how attention can fix it.
10. [2pts] Recall that in order to add a new primitive operation to Autograd, you need to define a vector-Jacobian product (VJP). To refresh your memory, here is code which defines VJPs for exponentiation and multiplication.

```python
defvjp(exp, lambda g, ans, x: ans * g)
defvjp(multiply, lambda g, ans, x, y: y * g, lambda g, ans, x, y: x * g)
```

The arguments to `defvjp` are the primitive op, followed by functions implementing the VJPs for each of the arguments. The arguments to the VJP function are: the output gradient `g`, the output `ans` of the op, and the arguments fed to the op.

(a) [1pt] Write Python code that defines a vector-Jacobian product for `sin`.

(b) [1pt] Write Python code that defines a vector-Jacobian product for `divide`, the function which computes the elementwise division of two arrays (i.e. `divide(x, y)` is equivalent to `x / y`). (This is floating point division, not integer division.)
11. [4pts] Suppose we receive two binary sequences \( x_1 = (x_1^{(1)}, \ldots, x_1^{(T)}) \) and \( x_2 = (x_2^{(1)}, \ldots, x_2^{(T)}) \) of equal length, and we would like to design an RNN to determine if they are identical. We will use the following (rather unusual) architecture, drawn with self-loops on the left and unrolled on the right:

The computation in each time step is as follows:

\[
\begin{align*}
 h^{(t)} &= \phi \left( W x^{(t)} + b \right) \\
 y^{(t)} &= \begin{cases} 
 \phi \left( v^\top h^{(t)} + r y^{(t-1)} + c \right) & \text{for } t > 1 \\
 \phi \left( v^\top h^{(t)} + c_0 \right) & \text{for } t = 1,
\end{cases}
\end{align*}
\]

where \( \phi \) denotes the hard threshold activation function

\[
\phi(z) = \begin{cases} 
 1 & \text{if } z > 0 \\
 0 & \text{if } z \leq 0
\end{cases}
\]

The parameters are a \( 2 \times 2 \) weight matrix \( W \), a 2-dimensional bias vector \( b \), a 2-dimensional weight vector \( v \), a scalar recurrent weight \( r \), a scalar bias \( c \) for all but the first time step, and a separate bias \( c_0 \) for the first time step.

We’ll use the following strategy. We’ll proceed one step at a time, and at time \( t \), the binary-valued elements \( x_1^{(t)} \) and \( x_2^{(t)} \) will be fed as inputs. The output unit \( y^{(t)} \) at time \( t \) will compute whether all pairs of elements have matched up to time \( t \). The two hidden units \( h_{1}^{(t)} \) and \( h_{2}^{(t)} \) will help determine if both inputs match at a given time step. **Hint:** have \( h_{1}^{(t)} \) determine if both inputs are 0, and \( h_{2}^{(t)} \) determine if both inputs are 1.
(Question 11, cont’d) Give parameters which correctly implement this function:

\[ W = \]

\[ b = \]

\[ v = \]

\[ r = \]

\[ c = \]

\[ c_0 = \]
12. [2pts] Suppose we have flipped a coin multiple times, and it came up heads $N_H$ times and tails $N_T$ times. We would like to model the coin as a Bernoulli random variable, and fit the model using maximum likelihood.

(a) [1pt] Give the formula for the log-likelihood $\ell(\theta)$, where $\theta$ is the probability of heads.

(b) [1pt] Solve for the maximum likelihood estimate of $\theta$ by setting $d\ell/d\theta = 0$.


(a) [1pt] What might go wrong if we eliminate the discriminator terms from the cost function?

(b) [1pt] What might go wrong if we eliminate the reconstruction terms from the cost function?
14. [2pts] Recall that a GAN could, in principle, be trained using the following minimax formulation, where $G$ is the generator function, $D$ is the probability the discriminator assigns to the sample being data, and $J_D$ and $J_G$ are the cost functions for the discriminator and generator, respectively.

\[
J_D = \mathbb{E}_{x \sim D}[- \log D(x)] + \mathbb{E}_z[- \log (1 - D(G(z)))]
\]

\[
J_G = - J_D
\]

\[
= \text{const} + \mathbb{E}_z[\log (1 - D(G(z)))]
\]

However, in practice, the generator is usually trained with a different loss function.

(a) [1pt] What cost function do we typically use for the generator?

(b) [1pt] What is the reason to use this cost function rather than the one given above?

15. [2pts] We’ve covered autoregressive generative models based on both convolutional networks and RNNs.

(a) [1pt] Give one advantage of using a convolutional network rather than an RNN.

(b) [1pt] Give one advantage of using an RNN rather than a convolutional network.
16. [3pts] Reversible architectures are based on a reversible block. Let’s modify the definition of the reversible block:

\[ y_1 = r \circ x_1 + \mathcal{F}(x_2) \]
\[ y_2 = s \circ x_2, \]

where \( \circ \) denotes elementwise multiplication. This modified block is identical to the ordinary reversible block, except that the inputs \( x_1 \) and \( x_2 \) are multiplied elementwise by vectors \( r \) and \( s \), all of whose entries are positive.

You don’t need to justify your answers for this question, but doing so may help you receive partial credit.

(a) [1pt] Give equations for inverting this block, i.e. computing \( x_1 \) and \( x_2 \) from \( y_1 \) and \( y_2 \). You may use \( / \) to denote elementwise division.

(b) [1pt] Give a formula for the Jacobian \( \partial y / \partial x \), where \( y \) denotes the concatenation of \( y_1 \) and \( y_2 \).

(c) [1pt] Give a formula for the determinant of the Jacobian from part (b).
17. [2pts] Suppose we have an MDP with two time steps. It has an initial state distribution \( p(s_1) \), transition probabilities \( p(s_{t+1} \mid s_t, a_t) \), and deterministic reward function \( r(s, a) \). The agent is currently following a stochastic policy \( \pi_\theta(a \mid s) \) parameterized by \( \theta \).

(a) [1pt] Give the formula for the probability \( p(\tau) \) of a rollout \( \tau = (s_1, a_1, s_2, a_2) \).

(b) [1pt] What is the function that REINFORCE is trying to maximize with respect to \( \theta \)? (You can give your answer in terms of \( p(\tau) \).)

18. [1pt] Recall that the discounted return is defined as:

\[
G_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i},
\]

where \( \gamma \) is the discount factor and \( r_t \) is the reward at time \( t \). Give the definition of the action-value function \( Q^\pi(s, a) \) for policy \( \pi \), state \( s \), and action \( a \). You can either give an equation or explain it verbally.