Durham University

EXAMINATION PAPER

Examination Session:	Year:	Exam Code:
May/June	2026	MATH4267-WE01

Title: Deep Learning and Artificial Intelligence

Time:	2 hours	
Additional Material provided:	None	
Materials Permitted:	None	
Calculators Permitted:	No	Models Permitted: Use of electronic calculators is forbidden.

Instructions to Candidates:	Answer all questions. The indicative marks shown in brackets for the main parts of each question are given as a guide to the weighting the markers expect to apply. Write your answer in the white-covered answer booklet with barcodes. Begin your answer to each question on a new page.

Revision:	

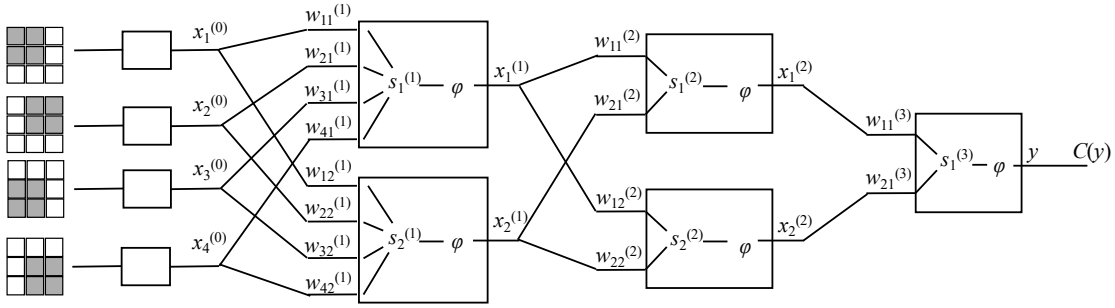
ED01/2026

SECTION A

1. Suppose we have a simple convolutional neural network which scans 3×3 pixel images using a 2×2 kernel:

$$\text{Image: } \begin{array}{|c|c|c|}\hline x_{11} & x_{12} & x_{13} \\ \hline x_{21} & x_{22} & x_{23} \\ \hline x_{31} & x_{32} & x_{33} \\ \hline \end{array} \qquad \text{Kernel: } \begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix}$$

The following diagram shows our network. In all layers, a differentiable activation function φ is used. All neurons have bias 0. We use a differentiable loss function C. Neuron outputs satisfy $x_i^{(\ell)} = \varphi(s_i^{(\ell)})$, where i indexes neurons within layer $\ell \in \{0, 1, 2, 3\}$, and value $w_{ij}^{(\ell)}$ indicates the weight for input i to neuron j in layer ℓ.



(a) Give a formula for $x_2^{(0)}$ in terms of the x_{ij}, $i, j \in \{1, 2, 3\}$ and p_{ij}, $i, j \in \{1, 2\}$. **[2]**

(b) Let $\delta_i^{(\ell)} = \frac{\partial C}{\partial s_i^{(\ell)}}$, for $i \in \{1, 2\}$ and $\ell \in \{1, 2, 3\}$. Show that:

$$\frac{\partial C}{\partial w_{11}^{(1)}} = \delta_1^{(1)} x_1^{(0)}.$$ **[2]**

(c) Find a formula for $\dfrac{\partial C}{\partial p_{12}}$ in terms of weights, inputs, and values $\delta_i^{(\ell)}$ with $i \in \{1, 2\}$, and without other partial derivatives. **[4]**

(d) Describe why we use backpropagation to find partial derivatives of loss functions with respect to parameters, as opposed to an 'empirical' derivative using finite differences (e.g. for some parameter w, calculating the losses $C(w)$ and $C(w+\epsilon)$ from passes through the network with W set to w or $w+\epsilon$ respectively, with $\partial C/\partial W \approx (C(w+\epsilon) - C(w))/\epsilon$). **[2]**

---

2. Denote by B^n the set of all sequences of values $\{-1, 1\}$ of length n. For instance, $B^2 = \{(-1,-1),(-1,1),(1,-1),(1,1)\}$. Let F^n denote the set of functions from B^n to $\{-1, 1\}$ (that is, taking n arguments, each in $\{-1, 1\}$).

Denote by sgn() the function:

$$\text{sgn}(x) = \begin{cases} 1 \text{ if } x \geq 0 \\ -1 \text{ if } x < 0 \end{cases}$$

(a) Consider the functions

$$AND(x_1, x_2, \ldots, x_r) = \begin{cases} 1 \text{ if } x_1 = x_2 = \cdots = x_r = 1 \\ -1 \text{ otherwise} \end{cases}$$

$$OR(x_1, x_2, \ldots, x_r) = \begin{cases} 1 \text{ if any of } x_1, x_2, \ldots, x_r \text{ equal } 1 \\ -1 \text{ otherwise} \end{cases}$$

where the neurons take some number r of inputs, each of which can take the value -1 or 1. Give weights and biases for neurons with activation function sgn() which implement these functions. **[3]**

(b) Suppose we have a fully connected neural network with a single hidden layer with 2^{n-1} neurons, and a single neuron in the second (output) layer, all with activation functions sgn(), and all of which may have a bias. Show that, by adjusting weights and biases, we can make this network implement any function in F^n. Hint: consider a table of all the elements in B^n, along with the values of the function for each of those elements. **[4]**

(c) Describe geometrically in $\mathbb{R}^n$ the class of functions which are implementable by a single neuron with activation sgn(). Find the VapnikChervonenkis dimension of this class if $n = 2$. **[3]**

SECTION B

3. We have a Boltzmann machine with n neurons, with temperature 1. Denote by X_i a random variable representing the output of the ith neuron taking values in $\{0, 1\}$. The weight of this output in the jth neuron is w_{ij}, and the bias of the ith neuron is b_i, where $w_{ij} = w_{ji}$ and $w_{ii} = 0$ for all $i, j \in \{1, 2, \ldots, n\}$. Denote by X_{-i} the ordered tuple of outputs X_j with $j \neq i$.

Denote by $P(\cdot)$ a probability under the equilibrium distribution of the Boltzmann machine. Consider an alternative distribution P_μ over states $x = (x_1, x_2, \ldots, x_n)$, parametrised by a vector $\mu = (\mu_1, \mu_2, \ldots, \mu_n)$, given by:

$$P_\mu(x) = \prod_{i=1}^{n} q_i(x_i)$$

where $q_i(1) = \mu_i$ and $q_i(0) = 1 - \mu_i$.

(a) Give expressions for $P(X_i = 1 | X_{-i} = x_{-i})$, where x_{-i} is an arbitrary vector of zeros and ones of length $n - 1$, and for $P(X_i = 1)$. Why is it difficult to calculate $P(X_i = 1)$ when n is large? **[3]**

(b) Show that a value of μ minimises the Kullback-Leibler divergence $D_{KL}(P_\mu || P)$ if and only if it also minimises:

$$F(\mu) = \mathbb{E}_{x \sim P_\mu}\{E(x)\} + \mathbb{E}_{x \sim P_\mu}\{\log(P_\mu(x))\}$$

where $\mathbb{E}_{x \sim P_\mu}\{\cdot\}$ is taken as expectation over the distribution P_μ of states x and $E(x)$ is the energy of state x. **[4]**

(c) Show that we have:

$$F(\mu) = -\frac{1}{2} \sum_{i,j \in \{1,2,\ldots,n\}} w_{ij}\mu_i\mu_j - \sum_{i=1}^{n} b_i\mu_i + \sum_i [\mu_i \log(\mu_i) + (1 - \mu_i)\log(1 - \mu_i)]$$

Hint: note that when $x \sim P_\mu$, x_i and x_j are independent for $i \neq j$. **[4]**

(d) Hence, or otherwise, show that $F(\mu)$ is minimised when

$$\mu_i = \sigma\left(b_i + \sum_{j=1}^{n} w_{ij}\mu_j\right),$$

where $\sigma(x)$ denotes the logistic function. You may consider μ_i to be in $(0, 1)$ for all i. How might this allow us to approximate $P(x_i = 1)$? **[4]**

---

4. Suppose that each year, a match is played between two sports teams 'N' and 'S'. Each match is independent, and team 'N' wins with probability p. The teams never draw. We wish to send the outcome (win/loss) of some large number n of matches by an expensive communication channel which can only transmit a 0-1 sequence.

(a) Suppose that team 'N' wins exactly pn games, where pn is an integer. Show that, in this case, the minimum number of bits $b(n)$ we need to transmit the sequence of results satisfies:

$$\lim_{n\to\infty} \frac{b(n)}{n} = -\left(p\log_2(p) + (1-p)\log_2(1-p)\right)$$

Describe why the same limit holds without the restriction that team 'N' wins exactly pn games, and state a general result of which this is an example. Hint: there exists a constant $c > 0$ such that, for all $n \geq 1$:

$$\left|\sum_{i=1}^{n} \ln(i) - n(\ln(n) - 1)\right| < c\sqrt{n}$$ **[7]**

(b) Suppose now that $p = 1/2$ and as well as the game outcomes, we want to transmit a series of images (as 1000×1000 matrices of grayscale pixel values in $\{0, 1, \ldots, 255\}$). Consider two scenarios:

(i) We consider the set G of every possible greyscale image, and partition this randomly into two sets G_N and G_S of about equal size. If team 'N' wins, we send a uniformly randomly chosen image from G_N, otherwise we send a uniformly randomly chosen image from G_S.

(ii) Team N's mascot is a newt, and team S's is a salamander. Each time team N wins, we catch a newt, take a grayscale picture, and send it. If S, we catch a salamander, take a grayscale picture, and send it.

Explain why we need fewer bits in the second scenario. **[3]**

(c) Our friend at the other end of the communication channel is trying to develop a predictor for the outcome of the game from the picture using a neural network. Denote by X the random variable associated with the picture and Y the random variable associated with the game outcome. Describe the relative benefits of using either a deep or a wide neural network for each scenario, with reference to the entropy of X, the mutual information between X and Y, and the entropy of any interim layers of the neural network. You may take the sets G_N and G_S to be fixed and known to our friend. **[5]**
