Note that I, Andrej Ilić, **changed** my HW partner from Nadim Hammoud to Georgy Antonov. Nadim was able to find another partner and the reason for this change was logistical.

Question 1 Information Theory

(a) Assuming the order does not matter, for placing 6 pawns on the checkerboard we have

$$^{64}C_6 = \frac{64!}{6!(64-6)!} = 74,974,368$$

combinations. Self-information (or simply information) is defined as

$$I(X) = -\log_2 P(X)$$

Hence, the information content of the aforementioned scenario is

$$I(X) = -\log_2 \left(\frac{1}{74,974,368}\right) = 26.160 \text{ bits}$$

(b) Entropy (or Shannon entropy) is a quantity that represents the average information one can obtain from observing a random variable. It is computeted as follows

$$H(X) = E[-\log_2 P(X)] = -\sum_{x \in X} P(x) \log_2 P(x)$$

The units of entropy are also bits. For the English alphabet letter frequency distribution A, the computed Shannon entropy was $H(A) = 4.176$ bits

(c) A uniform distribution is by definition a distribution where the occurrence of each outcome is equiprobable. Thus, for a uniform English letter frequency distribution $B \sim Uniform(26)$, the Shannon entropy is

$$H(B) = \sum_{i=1}^{26} -P\left(\frac{1}{26}\right) \log_2 P\left(\frac{1}{26}\right) = 4.700 \text{ bits}$$

Uniform distribution is also called the maximum entropy probability distribution, for since all the events are equiprobable then one does not reduce one's uncertainty about the future outcomes. Therefore, entropy of random variable B is greater than that of A.

(d) Cross-entropy between two probability distributions p and q over set X is defined as

$$H(p, q) = -\sum_{x \in X} p(x) \log_2 q(x)$$

Physical interpretation of this is the average amount of information one would need to identify an event from probability distribution p assuming it follows distribution q. Alternatively, it can also be defined through the Kullback-Leibler divergence

$$H(p, q) = H(p) + D_{KL}(p||q)$$

The computed cross-entropy for $H(A, B)$ and $H(B, A)$ was 4.700 and 5.590 bits, respectively. Cross-entropy for $H(A, B)$ is the same as that for the Shannon entropy $H(B)$, for since B is uniformly distributed, its log-sum can be factored out from the first cross-entropy equation, effectively giving $H(B)$. It is higher than $H(A)$ which makes sense because one needs more information to identify events following a uniform distribution. The cross-entropy $H(B, A)$ was even higher, for here we assume non-uniform priors for a uniform distribution which results in information loss.

(e) With the new distribution C, which is effectively B but with one random outcome probability set to zero, the computed cross-entropy $H(A, C)$ was 4.570 bits. Knowing that one outcome does not occur reduces uncertainty and thus decreases the amount of information one obtains from observing this random variable.

Question 2 Softmax

(a) Consider the following simplification

$$\log(\text{softmax}(\mathbf{x}_c)) = \log\left(\frac{e^{x_c}}{\sum_{i=1}^{|C|} e^{x_i}}\right)$$

Consider $x_M = \max_i x_i$

$$= \log\left(\frac{e^{x_c}}{e^{x_M}\sum_{i=1}^{|C|} e^{x_i - x_M}}\right) = x_c - x_M - \log\left(\sum_{i=1}^{|C|} e^{x_i - x_M}\right)$$

Observe that the sum in the above expression is at least 1 (because of the summand where $i = M$). At the same time, no element of this sum is bigger than 1. In other words, we prevent the scenarios of both "log(0)" and overflow from arising.

(b) Applying the insight from the previous exercise we know that $H(A, U) = H(U)$, where A is any distribution and U is a uniform distribution on the same set. This precisely corresponds to our situation here, given the distribution-like quality of the softmax. Namely, $A = \mathbf{y}_{\text{true}}$ and $U = \text{softmax}(\mathbf{x}_c)$. Entropy of the discrete uniform distribution on n items is

$$\sum_{i=1}^{n}\left[-\frac{1}{n}\log\left(\frac{1}{n}\right)\right] = \log(n)$$

so the cross-entropy loss for 10, 100 and 1000 classes is 2.3, 4.6 and 6.9 bits, respectively.

(c)

$$(-1)^s \times \overbrace{1.\underbrace{10101000101...}_{m}}^{base} \times 2^{e-\text{bias}}$$

IEEE 754 representation shown above has three components:

(1) Sign **[1 bit]**
Stored as 1 bit. 0 represents a positive number and 1 represents a negative number.

(2) Exponent [**5 bits**] (half precision) or [**8 bits**] (single precision)
Exponent is stored as a raw binary number of length e with special meanings associated with "all 0" and "all 1" values. This raw number corresponds to the sum of the encoded exponent and the bias. This allows positive and negative exponents.

(3) Mantissa [**10 bits**] (half precision) or [**23 bits**] (single precision)
As specified above (the 1 before the decimal point is fixed).

Overflow
The values of bias are 15 and 127, respectively, for the two precisions. That means that the largest exponents that are encodable are $11110_2 - 15_{10} = 15$ and $11111110_2 - 127_{10} = 127$ (remember, all 1s is an encoding reserved for infinity and NaN).

We also want to maximize the mantissa by setting it to all 1s. That corresponds the base values of $2 - 2^{-10}$ and $2 - 2^{-23}$.

Finally this corresponds to the maximal encodable numbers of $2^{15} * (2 - 2^{-10}) = 65504$ and $2^{127} * (2 - 2^{-23})$. If we want to avoid overflow e^x we should make sure that $x < \ln(65504) \approx 11.08$ and $x < \ln(2^{127} * (2 - 2^{-23})) \approx 88.72$.

Underflow
Similarly, the smallest exponents that are encodable are $1 - 15_{10} = -14$ and $1 - 127_{10} = -126$ (remember, all 0s is an encoding reserved for -infinity and NaN).

This time we want to minimize mantissa by setting it to all 0s. That corresponds the base value of 1.

Finally this corresponds to the maximal encodable numbers of 2^{-14} and 2^{-126}. If we want to avoid underflow e^x we should make sure that $x > \ln(2^{-14}) \approx -9.7$ and $x > \ln(2^{-126}) \approx -87.33$.

Question 3 Gradient and Hessian

(a)

$$f(\mathbf{x}, \theta) = \underbrace{w_5 \text{ReLU}\Big[x_1 w_1 + x_2 w_2\Big]}_{f_1(\mathbf{x},\theta)} + \underbrace{w_6 \text{ReLU}\Big[x_1 w_3 + x_2 w_4\Big]}_{f_2(\mathbf{x},\theta)}$$

$$\nabla_\theta(f) = \nabla_\theta(f_1) + \nabla_\theta(f_2)$$

where

$$\nabla_\theta(f_1) = \begin{cases} (x_1 w_5, x_2 w_5, 0, 0, x_1 w_1 + x_2 w_2, 0)^T, & x_1 w_1 + x_2 w_2 > 0 \\ \mathbf{0}, & \text{otherwise} \end{cases}$$

and

$$\nabla_\theta(f_2) = \begin{cases} (0, 0, x_1 w_6, x_2 w_6, 0, 0, x_1 w_3 + x_2 w_4)^T, & x_1 w_3 + x_2 w_4 > 0 \\ \mathbf{0}, & \text{otherwise} \end{cases}$$

(b)

$$\mathbf{H}_{5,1} = \mathbf{H}_{1,5} = x_1;\ \mathbf{H}_{5,2} = \mathbf{H}_{2,5} = x_2$$
$$\mathbf{H}_{6,3} = \mathbf{H}_{3,6} = x_1;\ \mathbf{H}_{6,4} = \mathbf{H}_{4,6} = x_2$$

Hessian at other positions takes value of 0.

(c) If Hessian is positive definite we would have, for any $\mathbf{z} \neq \mathbf{0}$

$$\mathbf{z}^T\mathbf{H}\mathbf{z} > 0$$

LHS evaluates to

$$z_1z_5x_1 + z_2z_5x_2 + z_3z_6x_1 + z_4z_6x_2 + z_5(x_1z_1 + x_2z_2) + z_6(x_1z_3 + x_2z_4)$$
$$= 2(z_1z_5x_1 + z_2z_5x_2 + z_3z_6x_1 + z_4z_6x_2)$$

which can be made negative by an adverserial choice of z, given x.

(d) The matrix would then store $500,000^2 = 250000000000$ parameters of 4 bytes which yields 10^{12} bytes which is 1000GB or - a terabyte.