Exam : Deep-Learning

Exercise 1: Basic questions - 7pts

1. **(1 pts)** Explain why and on what problems a linear classifier could fail to solve while the Multi-Layer Perceptron model would succeed.
2. **(1 pts)** Give the formal description of Multi-Layer Perceptron with one hidden layer and a binary classification layer (provide generic formula)
3. **(2 pts)** Propose different loss functions (at least two) for the above MLP (provide name and formula)
4. **(1 pts)** What is overfitting? How can it be detected? What methods can alleviate overfitting?
5. **(2 pts)** Show that minimizing:

$$\mathcal{L}(f_\theta(x), y) - \log\left(e^{-\frac{1}{2}\frac{(\theta-\mu)^2}{\sigma^2}}\right) \tag{1}$$

(with $\mu = 0$ and $\sigma^2 = \frac{1}{\lambda}$) using vanilla gradient descent (ϵ as learning rate) is equivalent to using weights decay update:

$$\theta_{t+1} = \theta_t - \epsilon(\nabla_{\theta_t}\mathcal{L}(f_{\theta_t}(x), y) + \lambda\theta_t) \tag{2}$$

What is the hypothesis on the weights in equation 1? What is the objective of using weight decay?

Exercise 2: The backpropagation algorithm - 7 pts

1. **(1 pts)** What structure do we use to backpropagate the gradient? On what rule/property does it rely upon? Explain briefly the mechanism
2. **(2 pts)** Compute the gradient of the following function according to weights a and b:

$$\begin{aligned} f_\theta :& \mathbb{R} \to \mathbb{R} \\ & x \mapsto a^{(2)}\sigma\left(a^{(1)}x + b^{(1)}\right) + b^{(2)} \end{aligned}$$

With σ being the hyperbolic tangent function and $a^{(i)}$ and $b^{(i)}$ scalars

3. **(2 pts)** Explain the backpropagation algorithm by illustrating your explanation using the following loss function and the previous MLP:

$$\mathcal{L}(x, y) = (f(x) - y)^2$$

4. **(2 pts)** What is the principle of the gradient descent considering momentum? Give the formula of the weights update.

Exercise 3: Deep-Learning Models - 12 pts

1. **(2 pts)** Give in pseudo-code the application of convolution on an input matrix X producing an output matrix Y (one channel input and output, no stride and no padding).

2. **(2 pts)** Let consider a CNN 2-dimensional layer (no stride and no padding) taking as input $x \in \mathbb{R}^{100\times100\times4}$ an image and produce 32 features maps, the kernel or filter size is 4×4. How many trainable parameters (float) does the model have? What would be the number of parameters if we considered a linear layer producing the same output size (considering the same number of floats in the output)? Show your calculation. 1/2

3. **(2 pts)** What are the differences between variational auto-encoder and classical auto-encoder? What is the main modification in the optimized error function? How can you generate examples in a variational auto-encoder?

4. **(2 pts)** How can be approximated the gradient according to ϕ of $\mathbb{E}_{z\sim q_\phi(z|x)} p_\theta(x|z)$. Give the formula of the gradient considering the reparametrization trick with $q_\phi(z|x)$ modeled by a Gaussian distribution of mean μ and variance σ^2

5. **(3 pts)** We consider q modelled by $\mathcal{N}(\mu, \sigma^2)$ and p by $\mathcal{N}(0, 1)$. Express the Kullback Liebler Divergence for the two distributions q and p as a function of μ and σ (without expectation $\mathbb{E}$), i.e. $KL(q||p)$.

6. **(1 pts)** Explain the principle of the Generative Adverserial Network? What is the associated optimization problem? (the formula is not mandatory but can be useful to explain the principle)