Contrôle de Connaissance
Master Recherche Informatique, parcours AIC - Université Paris-Saclay
TC Deep Learning
Alexandre Allauzen, Michèle Sebag

9:00-12:00 Nov. 13th, 2018

Documents autorisés: supports et notes de cours

Lisez tout l'énoncé. Pour toutes les questions la clarté de la rédaction joue un rôle important ; justifiez vos réponses brièvement.

Partie I. Questions de cours (5 points)

1. During training, we observe that the loss function is increasing on training data. What is going on? (single answer)
   - A. There is not enough regularization and the network is overfitting;
   - B. There is too much regularization and the network is underfitting;
   - C. The learning rate is too big;
   - D. The learning rate is too small.
2. Let inputs of a network be vectors $x \in \mathbb{R}^d$. Describe the architecture of a convolutional network using these inputs. Which properties does it satisfy?
3. We now consider matrix inputs $x \in \mathbb{R}^{d \times d'}$, e.g. images. Describe a convolutional network and its properties.
4. How should a neural network be initialized when it is build with sigmoid activation functions? Which issue should be avoided? Does this problem exist with deep and/or shallow networks?
5. Will two differently initialized networks with the rule you proposed in question 4 attain the same solution after training?
6. Will a neural network be correctly trained if all weights are initialized to 0? and if all bias vectors are initialized to 0?
7. Let
   $$\mathcal{E} = \{(x_i, y_i), i = 1 \ldots n, x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}\}$$
   be a set of examples. We assume an acyclic computation graph G with n neurons, where arcs (i, j) are sampled randomly and where weights $w_{i,j}$ are sampled from distribution $\mathcal{N}(0, 1)$. All neurons are connected to inputs. We only train the n output neurons. What function is optimized? Is it a convex or a non-convex optimization problem?

Partie II. Algorithme d'apprentissage (9 points)

We assume a feed-forward network with a single hidden layer. Let $\boldsymbol{x}^{(1)}$ be the input vector. The hidden layer is $\boldsymbol{y}^{(1)} = f^{(1)}(\boldsymbol{W}^{(1)}\boldsymbol{x}^{(1)})$. The output layer is $\boldsymbol{y}^{(2)} = f^{(2)}(\boldsymbol{W}^{(2)}\boldsymbol{x}^{(2)})$, with $\boldsymbol{x}^{(2)} = \boldsymbol{y}^{(1)}$. The activation function ($f^{(1)}$) of the hidden layer is hyperbolic tangent. We consider a binary classification task, so the output layer as a single neuron that can be interpreted as the probability that $\boldsymbol{x}^{(1)}$ belongs to class $c = 1$. We maximize the likelihood of training data, i.e.:

$$l(\boldsymbol{\theta}, \boldsymbol{x}, c) = -\big(c \log(\boldsymbol{y}^{(2)}) + (1 - c) log(1 - \boldsymbol{y}^{(2)})\big),$$

where $\boldsymbol{x}$ is a training sample, c the gold output ($c = 0$ or 1) and $\boldsymbol{\theta}$ the parameters of the network. Note that the output is a scalar so $\boldsymbol{W}^{(2)}$ is a row matrix.

1. Which function should we apply to the output?

2. Write the update rule for the output layer $w_j^{(2)}$ which correspond to element j of $\boldsymbol{W}^{(2)}$. To this end, proceed as follows:
   - Express the value $y^{(2)}$ in term of $\boldsymbol{x}^{(2)}$ and $\boldsymbol{W}^{(2)}$.
   - Compute the derivative of $w_j^{(2)}$ with respect to the loss.
   - Write and describe the update rule.
3. Similarly for hidden layer $w_{kj}^{(1)}$.[1]
4. Describe the learning algorithm for this network.

Partie III. Auto-encodeurs (5 points)

We consider an unlabeled dataset:

$$\mathcal{E} = \{x_i, i = 1 \ldots n, x_i \in \mathbb{R}^d\}$$

- What is an auto-encoder? What is the associated loss function? What is the goal of auto-encoders?
- What is the loss function of a denoising auto-encoder? What is its purpose against standard auto-encoders?
- How do you choose the number of hidden neurons of an auto-encoder?
- Is the euclidean distance in the hidden state space a good bound on the distance in the initial space?

Partie IV. Exemples adversariaux (5 points)

We now consider adversarial examples. We assume a training set defined as follows:

$$\mathcal{E} = \{(x_i, y_i), i = 1 \ldots n, x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}\}$$

. The network is a feed-forward network with 3 hidden layers and d neurons per layer, trained on $\mathcal{E}$.

- What is an adversarial example? How to construct an adversarial example?
- Can a human be fooled by an adversarial example?
- Propose a loss function inspired by previous questions.

[1] The derivative of the hyperbolic tangent function $tanh$ is $tanh'(a) = 1 - tanh^2(a)$.