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

9 :00-12 :00 13 novembre 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. Parfois une équation ou un schéma bien expliqué vaut mieux qu'un long discours.

Partie I. Questions de cours (5 points)

1. Si la fonction de perte sur les données d'apprentissage augmente au cours des époques, que se passe-t-il ? (une seule réponse) A. Le poids de régularisation est trop bas et le modèle overfitte ; B. Le poids de régularisation est trop haut et le modèle underfitte ; C. Le taux d'apprentissage (learning rate) est trop grand ; D. Le taux d'apprentissage est trop petit.
2. Soit un problème dont les exemples sont des vecteurs ($x \in \mathbb{R}^d$). Décrivez la structure d'un réseau neuronal convolutionnel. Quelle propriété vérifie-t-il ?
3. On considère maintenant le cas d'exemples matriciels ($x \in \mathbb{R}^{d \times d'}$. Indication, pensez à une image). De même, décrivez la structure d'un réseau neuronal convolutionnel et les propriétés vérifiées.
4. Comment initialiser un réseau neuronal feedforward dont la fonction d'activation est une sigmoïde ? Quel est le problème qu'on cherche à éviter ? Ce problème est-il plus grave ou moins grave lorsque le nombre de couches augmente ?
5. Deux initialisations différentes (mais suivant la règle de la question 4) d'un réseau neuronal feedforward conduisent à la même solution finale : vrai ou faux ?
6. Etant donné une base d'exemples

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

   on considère un réseau feedforward dont la fonction d'activation est RELU, dont l'architecture et les poids sont aléatoires. On se limite à optimiser les poids menant des neurones de la dernière couche à la sortie. Quelle est la fonction à optimiser ? S'agit-il d'un problème d'optimisation convexe ou non-convexe ?
7. Peut-on apprendre si les poids sont initialisés à 0 ? Si les biais sont initialisés à 0 ?

Partie II. Algorithme d'apprentissage (6 points)

Considérons un réseau de neurones feed-forward avec une couche cachée avec les notations suivantes : le vecteur d'entrée est $\boldsymbol{x}^{(1)}$; la couche cachée est représentée par le vecteur $\boldsymbol{y}^{(1)} = f^{(1)}(\boldsymbol{W}^{(1)}\boldsymbol{x}^{(1)})$; la sortie est $\boldsymbol{y}^{(2)} = f^{(2)}(\boldsymbol{W}^{(2)}\boldsymbol{x}^{(2)})$, avec $\boldsymbol{x}^{(2)} = \boldsymbol{y}^{(1)}$. La fonction d'activation ($f^{(1)}$) de la couche cachée est la tangente hyperbolique. La tâche considérée est de la classification binaire, donc la couche de sortie n'a qu'un seul neurone qui peut directement s'interpréter comme la probabilité que $\boldsymbol{x}^{(1)}$ appartienne à la classe $c = 1$. La fonction objectif est le maximum de vraisemblance, soit pour un exemple :

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

où $\boldsymbol{x}$ est l'exemple d'apprentissage , c la classe de référence à prédire ($c = 0$ or 1) et $\boldsymbol{\theta}$ regroupe tous les paramètres du modèle. Notons que la couche de sortie n'a qu'un neurone et donc que $\boldsymbol{W}^{(2)}$ est une matrice ligne.

1. Quelle fonction d'activation proposez-vous pour le neurone de sortie ?
2. Ecrire la formule de mise à jour des paramètres de la couche de sortie , $w_j^{(2)}$ qui correspond à l'élément j de $\boldsymbol{W}^{(2)}$. Pour cela, vous pouvez si vous le souhaitez suivre les étapes suivantes :
   — Exprimer la valeur du neurone de sortie $y^{(2)}$ en fonction de $\boldsymbol{x}^{(2)}$ et $\boldsymbol{W}^{(2)}$.

— Partant de la fonction objectif calculer sa dérivée par rapport à $w_j^{(2)}$.
— Donner et interpréter la formule de mise à jour.

3. Faire de même avec la couche cachée $w_{kj}^{(1)}$. Vous pouvez pour cela adapter la démarche précédente aux spécificités de la couche cachée [1].
4. Décrire de manière plus globale l'algorithme d'apprentissage de ce réseau (définir la fonction objectif sur l'ensemble des données d'apprentissage et comment l'optimiser, ...).

Pour ces questions la clarté de la rédaction joue un rôle important.

Partie III. Auto-encodeurs (5 points)

On considère la base d'apprentissage non supervisée :

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

— Rappeler la définition d'un auto-encodeur et son critère d'apprentissage. Quel est le but d'un auto-encodeur ?
— Rappeler le critère d'apprentissage d'un denoising auto-encodeur. Quel est l'intérêt de ce critère par rapport à un auto-encodeur simple ?
— Comment ajuster le nombre de neurones d'un auto-encodeur à une couche cachée ?
— La distance euclidienne dans l'espace de la couche cachée (espace latent) permet-elle de définir une borne sur la distance dans l'espace initial ? (Indication, les fonctions encodeur / décodeur sont des fonctions continues dérivables).

Partie IV. Exemples adversariaux (5 points)

Ce problème considère les exemples adversariaux. On suppose qu'il existe une base d'entrainement :

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

et un réseau feedforward à 3 couches cachées avec d neurones sur chacune des couches (architecture $d - d - d$), entrainé sur la base $\mathcal{E}$.

— Rappeler la définition d'un exemple adversarial. Quelles conditions doit-il vérifier ? Ecrire l'algorithme permettant de construire un exemple adversarial.
— Un exemple adversarial réussit-il à tromper un humain ? Pourquoi ?
— Proposer une modification de la fonction de perte en s'inspirant de la question précédente (indication : on pourra s'inspirer des denoising auto-encodeurs).

1. La dérivée de la fonction tangente hyperbolique $tanh$ est $tanh'(a) = 1 - tanh^2(a)$