# Compte-Rendu

## Deep Learning TP 3-4
## Introduction aux réseaux de neurones

**Yining Bao** **Hanshuo WANG** **Yongtao WEI**

## 1. Formalisation mathématique

### 1.1. Jeu de données

#### 1. A quoi servent les ensembles d'apprentissage, de validation et de test ?

Les ensembles d'apprentissage servent à entraîner le modèle. Il contient essentiellement l'ensemble des éléments à entraîner sauf les éléments de de validation et de test.

Les ensembles de validation sont pour régler les hyperparamètres du modèle. Les ensembles de validation fournissent une évaluation impartiale du modèle ajusté sur l'ensemble d'apprentissage.

Les ensembles de test sont utilisés pour fournir une évaluation impartiale du modèle final et prédire le score si on utilise le modèle final à prédire sur les autres données inconnues.

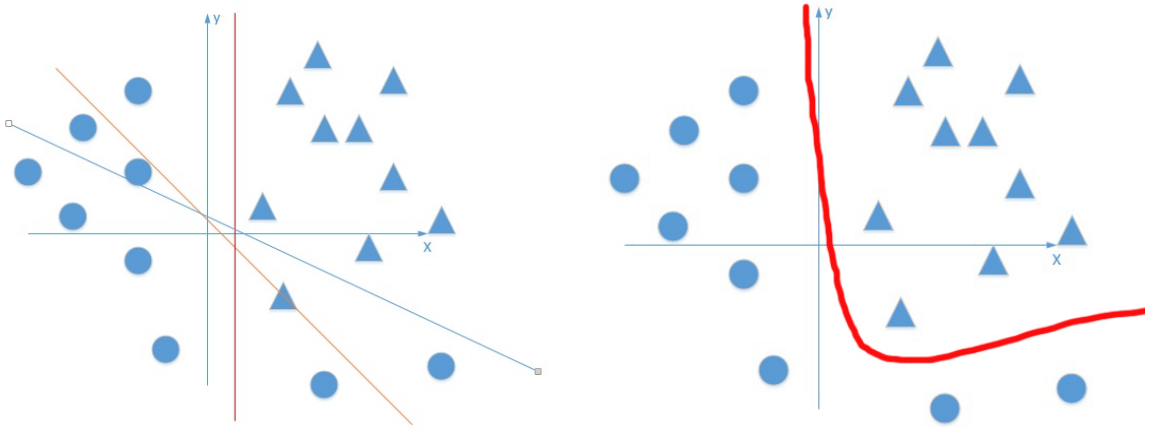
#### 2. Quelle est l'influence du nombre N d'exemples ?

La taille de N détermine la précision de notre modèle final formé. C'est-à-dire qu'on a plus d'exemples et on peut ajuster plus précisément sur le modèle, alors le modèle sera plus précis et fort.

### 1.2. Architecture du reseau (phase forward)

#### 3. Pourquoi est-il important d'ajouter des fonctions d'activation entre des transformations linéaires ?

Si la fonction d'activation n'est pas utilisée, la sortie de chaque couche est une fonction linéaire de l'entrée de la couche précédente. Quel que soit le nombre de couches du réseau de neurones, la sortie est une combinaison linéaire d'entrées. Il ne peut donc résoudre que des problèmes linéaires. (Voyez l'image à gauche)




Après utiliser la fonction d'activation, on introduit des facteurs non linéaires dans le neurone, le réseau neuronal peut approximer arbitrairement n'importe quelle fonction non linéaire, le réseau neuronal peut être appliqué à de nombreux modèles non linéaires. (Voyez l'image à droite)

## 4. Quelles sont les tailles nx, nh, ny sur la figure 1 ? En pratique, comment ces tailles sont-elles choisies ?

nx est la taille ou le nombre des entrées, dans la figure il y a 2 entrées x1 et x2, donc nx = 2. nh est la taille de neurones dans la couche cachée, dans la figure il y a 4 neurones h1, h2, h3 et h4, donc nh = 4. ny est la taille de sorties, dans la figure il y a 2 sorties yˆ1 e yˆ2, donc ny = 2.

La taille de nx est choisie par la taille des données en entrée.

Pour la taille de nh, étant donné que chaque neurone caché supplémentaire augmente le nombre de poids, il est généralement recommandé d'utiliser le nombre minimal de neurones cachés permettant d'accomplir la tâche. L'utilisation de plus de neurones cachés que nécessaire augmente la complexité et peut conduire à un sur-apprentissage.

La taille de ny est choisie par le nombre de classes pour la classification, parce que dans le sujet on utilise encodage one-hot pour la sortie.

## 5. Que représentent les vecteurs yˆ et y ? Quelle est la différence entre ces deux quantités ?

Le vecteur y représente le label correct d'une donnée d'entrée de x, et le vecteur yˆ est la sortie d'après le calcul du modèle à travers la couche cachée.

La différence entre les deux vecteurs est que le vecteur y est associé avec x et yˆ est associé avec f(x).

### 6. Pourquoi utiliser une fonction SoftMax en sortie ?

La fonction softmax est de faire la normalisation logarithmique du gradient de la distribution de probabilité discrète aux éléments finis. La fonction softmax peut donc d'avoir la classe plus probable.

SoftMax est utilisé dans le processus de multi-classification, qui fait correspondre la sortie de plusieurs neurones, dans l'intervalle (0,1), qui peut être compris comme des probabilités, pour effectuer la multi-classification.

### 7. Écrire les équations mathématiques permettant d'effectuer la passe forward du réseau de neurones, c'est-à-dire permettant de produire successivement $\tilde{h}$, h, $\tilde{y}$ et $\hat{y}$ a partir de x.

$$\tilde{h} = W_h * x + b_n$$

$$h = \tanh(\tilde{h})$$

$$\tilde{y} = W_y * h + b_y$$

$$\hat{y} = \text{SoftMax}(\tilde{y})$$

## 1.3. Fonction de coût

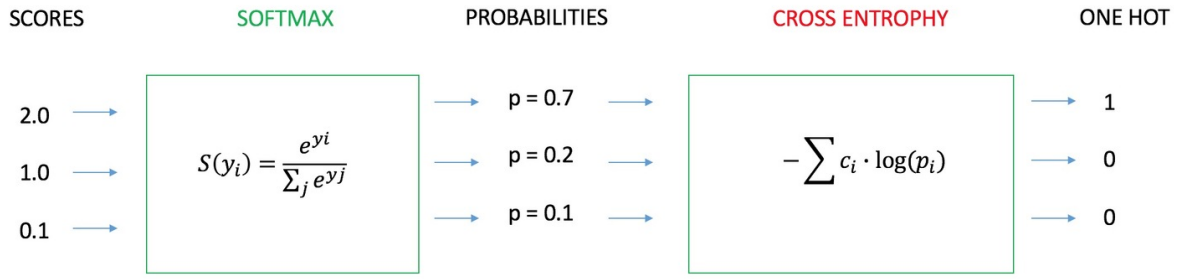
### 8. Pendant l'apprentissage, on cherche à minimiser la fonction de coût. Pour l'entropie croisée et l'erreur quadratique, comment les $\hat{y}_i$ doivent-ils varier pour faire diminuer la loss ?

Tant pour l'entropie croisée que pour l'erreur quadratique, pour réduire la loss, $\hat{y}_i$ devrait se rapprocher de plus en plus de $y_i$.

### 9. En quoi ces fonctions sont-elles plus adaptées aux problèmes de classification ou de régression.

Le modèle pour résoudre le problème de classification doit sortir la probabilité qu'une entrée soit une certaine classification, donc la fonction SoftMax (ou sigmoid) est utilisée comme dernière fonction d'activation, car il transforme la sortie de la propagation forward en une distribution de probabilité.

L'entropie croisée décrit la distance entre deux distributions de probabilité, une distance plus petite indiquant que les deux probabilités sont plus similaires et une plus grande indiquant que les deux probabilités sont plus différentes.




À l'inverse, lorsque l'on utilise sigmoïde/softmax, avec une fonction de perte MSE, et que l'on apprend par descente de gradient dans un problème de classification, on observe un déclin extrêmement lent de la fonction de perte pendant l'apprentissage du modèle (Disparition du gradient).

En outre, lorsque l'erreur quadratique et l'entropie croisée sont appliquées à un scénario de classification multiple, l'erreur quadratique accorde une grande valeur au résultat de chaque sortie, tandis que l'entropie croisée n'accorde de valeur qu'au résultat de la classification correcte. Par exemple, dans un modèle à trois classifications où la sortie du modèle est (a,b,c) et la vraie sortie est (1,0,0), la valeur de la fonction de perte correspondant à l'erreur quadratique et à l'entropie croisée est la suivante.

l'erreur quadratique :

$$c = (a-1)^2 + (b-0)^2 + (c-0)^2 = (a-1)^2 + b^2 + c^2$$

l'entropie croisée :

$$c = (-1) \cdot log(a) - 0 \cdot log(b) + 0 \cdot log(c) = -log(a)$$

Comme on peut le voir dans les équations ci-dessus, la fonction de perte de l'entropie croisée est uniquement liée au résultat prédit de la classification correcte, tandis que la fonction de perte de l'erreur quadratique est également liée à la classification incorrecte. La fonction de classification, en plus de rendre la classification correcte aussi grande que possible, rendra également la classification incorrecte moyenne, mais en pratique, cet ajustement n'est pas nécessaire dans un problème de classification. Pour les problèmes de régression, cependant, une telle considération devient importante.

Par conséquent, L'entropie croisée est plus adaptée aux problèmes de classification, et l'erreur quadratique est plus adaptée aux problèmes de régression.

### 1.4. Méthode d'apprentissage

**10. Quels semblent être les avantages et inconvénients des diverses variantes de descente de gradient entre les versions classique, stochastique sur mini-batch et stochastique online ? Laquelle semble la plus raisonnable à utiliser dans le cas général ?**

| | Avantages | Inconvénients |
|---|---|---|
| Descente de gradient classique | Il utilise des accélérations vectorielles[1] qui peuvent accélérer l'entraînement. | Lorsque la dimension des données d'entrée est grande, une seule itération prend trop de temps. |
| Descente de gradient stochastique online | 1. Il peut utiliser efficacement les informations de l'échantillon, notamment lorsque celles-ci sont redondantes.[2]<br>2. Il existe un caractère aléatoire, qui peut sortir de la région du minimum local. | Pas d'accélération vectorielle, moins efficace que la descente de gradient classique. |
| Descente de gradient stochastique sur mini-batch | 1. Il utilise des accélérations vectorielles.<br>2. Il peut utiliser efficacement les informations de l'échantillon, notamment lorsque celles-ci sont redondantes.<br>3. Il existe un caractère aléatoire, qui peut sortir de la région du minimum local.<br>4. Il n'est pas nécessaire d'attendre que l'ensemble de l'apprentissage soit traité avant de commencer le travail suivant. | Les résultats de l'apprentissage peuvent ne pas converger ou ne pas fluctuer dans une petite plage. |

D'après le tableau ci-dessus, nous pouvons voir que la descente de gradient stochastique sur mini-batch combine les avantages des deux autres et est le plus adapté à une utilisation dans la plupart des situations.

## 11. Quelle est l'influence du learning rate η sur l'apprentissage ?

Le taux d'apprentissage représente la vitesse du changement de w.

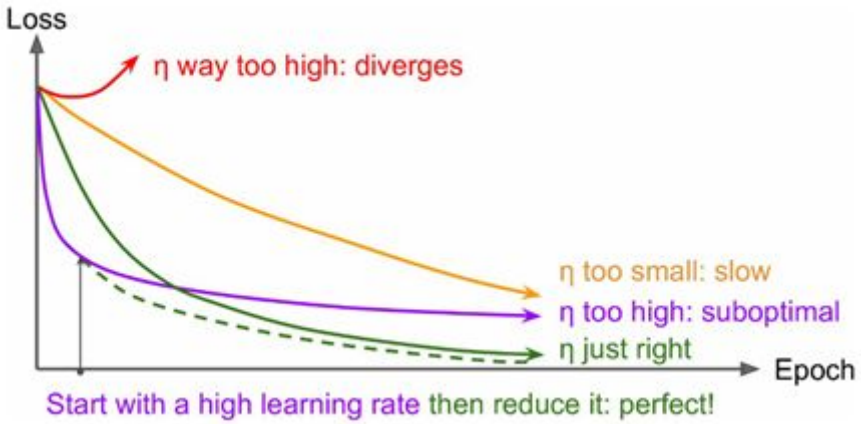
Si le taux d'apprentissage est fixé très haut, les résultats risquent de ne pas converger.

Si le taux d'apprentissage est fixé à un niveau relativement faible, l'apprentissage peut prendre trop de temps, bien qu'il puisse converger.

Si le taux d'apprentissage est légèrement plus élevé, l'apprentissage peut être rapide, mais il oscille, voire ne se stabilise pas, à mesure qu'il approche de sa valeur optimale.

Le choix du taux d'apprentissage peut avoir un impact significatif sur le processus et le résultat de l'apprentissage, comme le montre le graphique ci-dessous.

[1] En utilisant des vecteurs au lieu de boucles for, plusieurs boucles for peuvent être transformées en un seul calcul, ce qui permet d'accélérer le processus.
[2] Par exemple, si tous les échantillons doivent être optimisés dans une direction, la descente de gradient classique nécessite une itération de l'ensemble des échantillons, alors que la descente de gradient stochastique online peut obtenir le même résultat en optimisant un seul échantillon.




## 12. Comparer la complexité (en fonction du nombre de couches du réseau) du calcul des gradients de la loss par rapport aux paramètres, en utilisant l'approche naïve et l'algorithme de backprop.

1. Complexité spatiale : dans l'algorithme naïf, le calcul différentiel ne nécessite que la valeur de la couche d'entrée de la couche courante, et pas du tout les valeurs des couches précédentes, donc seules deux variables doivent être maintenues : la valeur différentielle de la couche courante par rapport à la couche d'entrée, et la valeur de la couche courante, soit une complexité = O(1). L'algorithme de back propagation est différent en ce sens qu'il doit calculer les valeurs une fois auparavant et sauvegarder les valeurs de toutes les couches pour le calcul ultérieur des valeurs différentielles de chaque couche, donc complexité = O(n).

2. Complexité en temps : comme l'algorithme naïf ne réutilise pas les valeurs différentielles déjà utilisées, on peut savoir que la complexité en temps de l'algorithme naïf est supérieure à celle de l'algorithme de back propagation.

En résumé, la rétropropagation est préférable puisque c'est précisément le problème des longs temps d'apprentissage et de la puissance arithmétique que nous sommes le plus intéressés à résoudre dans le processus d'apprentissage automatique.

## 13. Quel critère doit respecter l'architecture du réseau pour permettre la backpropagation ?

Comme l'algorithme de la backpropagation nécessite l'utilisation du théorème de dérivation des fonctions composées, nous devons nous assurer que chaque fonction du réseau est dérivable, c'est-à-dire que la fonction de Loss et les fonctions d'activation doivent être dérivables.

## 14. La fonction SoftMax et la loss de cross-entropy sont souvent utilisées ensemble et leur gradient est très simple. Montrez que la loss se simplifie en :

$$\ell = -\sum_i y_i \tilde{y}_i + \log\left(\sum_i e^{\tilde{y}_i}\right)$$

La fonction de la loss de cross-entropy :

$$\ell(y, \hat{y}) = -\sum_i y_i \log \hat{y}_i$$

La fonction SoftMax :

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

Par conséquent, on a :

$$\hat{y}_i = SoftMax(\bar{y}_i) = \frac{e^{\bar{y}_i}}{\sum_j e^{\bar{y}_i}}$$

$$l(y, \hat{y}) = -\sum_i y_i \log \frac{e^{\tilde{y}_i}}{\sum_i e^{\tilde{y}_i}} = -\sum_i y_i \left( \tilde{y}_i - \log \sum_i e^{\tilde{y}_i} \right)$$

Puisque yi est en codage one-hot, un seul yi a une valeur de 1 et les autres valeurs de yi sont 0. Donc on a :

$$\ell = -\sum_i y_i \tilde{y}_i + \log \left( \sum_i e^{\tilde{y}_i} \right)$$

## 15. Ecrire le gradient de la *loss (cross-entropy)* par rapport à la sortie intermédiaire y˜

Sachant que

$$\ell = -\sum_i y_i \tilde{y}_i + \log \left( \sum_i e^{\tilde{y}_i} \right)$$

$$\hat{y}_i = SoftMax(\bar{y}_i) = \frac{e^{\bar{y}_i}}{\sum_j e^{\bar{y}_i}}$$

On a :

$$\frac{\partial l}{\partial \bar{y}_i} = -y_i + \frac{e^{\bar{y}_i}}{\sum_i e^{\bar{y}_i}} = -y_i + \hat{yi}$$

Soit :

$$\nabla_{\bar{y}} l = \hat{y} - y$$

16. En utilisant la backpropagation, écrire le gradient de la *loss* par rapport aux poids de la couche de sortie $\nabla W_y l$. Notez que l'écriture de ce gradient utilise $\nabla \tilde{y} l$. Faire de même pour $\nabla by\ l$.

Sachant que

$$\tilde{y} = W_y * h + b_y$$

On a

$$\nabla W_y l = \nabla \tilde{y} l \cdot \nabla W_y \tilde{y} = \nabla \tilde{y} l \cdot h$$

$$\nabla b_y l = \nabla \tilde{y} l \cdot \nabla b_y \tilde{y} = \nabla \tilde{y} l$$

17. Calculer les autres gradients : $\nabla \tilde{h}\ l$, $\nabla W_h\ l$, $\nabla b_h\ l$.

Sachant que

$$\tilde{y} = W_y \cdot h + b_y \qquad h = \tanh(\tilde{h})$$

On a

$$\nabla_{\tilde{h}} l = \nabla_{\tilde{y}} l \cdot \nabla_h \tilde{y} \cdot \nabla_{\tilde{h}} h = \nabla_{\tilde{y}} l \cdot W_y \cdot (1 - h^2)$$

Sachant que

$$\tilde{h} = W_h \cdot X + b_h$$

On a

$$\nabla_{W_h} l = \nabla_{\tilde{h}} l \cdot \nabla_{W_h} \tilde{h} = \nabla_{\tilde{h}} l \cdot X$$

$$\nabla_{b_h} l = \nabla_{\tilde{h}} l \cdot \nabla_{b_h} \tilde{h} = \nabla_{\tilde{h}} l$$

## 2. Bonus : SVM

Q1. Essayer d'abord un SVM linéaire (sklearn.svm.LinearSVC dans scikit-learn). Est-ce que cela fonctionne bien ? Pourquoi ?

La fonction LinearSVC n'est pas efficace car elle ne peut réaliser qu'une classification linéaire. La distribution des données de l'ensemble de données que nous utilisons n'est pas linéaire, donc cette fonction ne peut pas très bien distinguer les échantillons de données. Par exemple, elle ne peut pas générer un cercle pour distinguer le centre des données. Il utilise la bibliothèque liblinear et ne peut classer que des données distribuées linéairement.

### Q2: Essayer d'autres kernels (possible avec sklearn.svm.SVC).

La fonction svc fonctionne mieux. Il utilise la bibliothèque libsvm, qui peut réaliser la classification de données distribuées non linéairement.

### Q3: Est-ce que le paramètre C de régularisation à un impact ? Pourquoi ?

Le paramètre de régularisation C est effectif. Grâce à des expériences, nous avons constaté que lorsque le paramètre de régularisation est trop grand, la précision de la prédiction diminue, ce qui devrait limiter la complexité du modèle. Cependant, si le paramètre C est trop petit, il y aura surapprentissage.

## 3. Bibliographie


https://www.zhihu.com/question/23765351

https://blog.csdn.net/weixin_41888969/article/details/89450163

https://zhuanlan.zhihu.com/p/70910873

https://zhuanlan.zhihu.com/p/31630368

https://www.zhihu.com/question/27239198/answer/154510111