RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

METEO FRANCE
À VOS CÔTÉS, DANS UN CLIMAT QUI CHANGE

Machine Learning – correction du test n°1

Pierre Lepetit
ENM, le 22/11/2024

Correction du test :

1) Qu’est-ce qu’un problème de Machine Learning (supervisé) ?
Quels sont les trois points auxquels on prête généralement attention quand on en aborde un ?
Citer trois exemples de problèmes différents.

Correction du test :

1) Qu'est-ce qu'un problème de Machine Learning (supervisé) ?
Quels sont les trois points auxquels on prête généralement attention quand on en aborde un ?
Citer trois exemples de problèmes différents.

Pb de ML : Problème avec des x (entrées), des y (cibles) ; la recherche d'une fonction **f** qui « fait le lien » entre entrées et cibles.

Trois points d'intérêt : Expressivité, entraînabilité, performances en généralisation

Exemples : Classification d'image, segmentation d'images, régression.

Remarque : en général on ajoute un ingrédient à la définition :
un (ou plusieurs) score(s) pour évaluer le lien entre f(x) et y.

Correction du test :

2) Décrire de manière concise les étapes principales de la phase d’entraînement d’une boucle d’apprentissage standard.

Correction du test :

2) Décrire de manière concise les étapes principales de la phase d'entraînement d'une boucle d'apprentissage standard.

On parcourt le jeu d'entraînement en échantillonnant des batches d'entrées et pour chaque batch :

- Calcul des sorties associées aux entrées du batch
- Evaluation de la fonction de coût sur le batch
- Calcul des gradients de la fonction de coût par rapport aux poids du modèle
- Mise à jour des poids

Remarque : batch (anglais pratique) = mini-batch (anglais précis) = mini-lot (français précis)

Correction du test :

3) Soit $x = (x_i)$ un 22-uplet de réels, (w_i) les 22 poids synaptiques et b le biais d'un perceptron à une couche défini sous pytorch par `self.fc = nn.Linear(22, 1)`.
Ecrire l'expression mathématique de self.fc(x).

Correction du test :

4) On se donne l'architecture suivante :

```
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(3, 10, kernel_size=3, padding=1)
        self.conv2 = nn.Conv2d(10, 10, kernel_size=5, padding=2)
        self.fc1 = nn.Linear(1000, 50, bias=False)
        self.fc2 = nn.Linear(50, 10)

    def forward(self, x):
        x = self.conv1(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
        x = self.conv2(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
        x = x.view(-1, 1000)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

Correction du test :

4) On se donne l'architecture suivante :

```
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(3, 10, kernel_size=3, padding=1)
        self.conv2 = nn.Conv2d(10, 10, kernel_size=5, padding=2)
        self.fc1 = nn.Linear(1000, 50, bias=False)
        self.fc2 = nn.Linear(50, 10)

    def forward(self, x):
        x = self.conv1(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
        x = self.conv2(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
        x = x.view(-1, 1000)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

4.a) Nombre de poids :

convolutions :
10*3*3*3 + 10 + 10*10*5*5 + 10
= 270 + 2500 + 20 = 2790

complètement connectés
1000*50 + 50*10 + 10 = 50510

→ 53600 poids

Correction du test :

4) On se donne l'architecture suivante :

```
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(3, 10, kernel_size=3, padding=1)
        self.conv2 = nn.Conv2d(10, 10, kernel_size=5, padding=2)
        self.fc1 = nn.Linear(1000, 50, bias=False)
        self.fc2 = nn.Linear(50, 10)

    def forward(self, x):
        x = self.conv1(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
        x = self.conv2(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
        x = x.view(-1, 1000)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

4.a) Nombre de poids :

convolutions :
10*3*3*3 + 10 + 10*10*5*5 + 10
= 270 + 2500 + 20 = 2790

complètement connectés
1000*50 + 50*10 + 10 = 50510

→ 53600 poids

4.b) F.relu et max_pool2d

Correction du test :

4) On se donne l'architecture suivante :

```
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(3, 10, kernel_size=3, padding=1)
        self.conv2 = nn.Conv2d(10, 10, kernel_size=5, padding=2)
        self.fc1 = nn.Linear(1000, 50, bias=False)
        self.fc2 = nn.Linear(50, 10)

    def forward(self, x):
        x = self.conv1(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
        x = self.conv2(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
        x = x.view(-1, 1000)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

4.a) Nombre de poids :

convolutions :
10*3*3*3 + 10 + 10*10*5*5 + 10
= 270 + 2500 + 20 = 2790

complètement connectés
1000*50 + 50*10 + 10 = 50510

→ 53600 poids

4.b) F.relu et max_pool2d
4.c) 3x40x40 (ou 3x20x80...)

Correction du test :

5) Quel est le rôle de la phase de validation dans une boucle d’apprentissage ?

Correction du test :

5) Quel est le rôle de la phase de validation dans une boucle d’apprentissage ?

Surveiller les performances en généralisation du modèle au cours de l’apprentissage.

Correction du test :

6) Quel speed-up (ordre de grandeur) peut-on attendre d'un passage sur carte graphique pour l'apprentissage d'un modèle de la classe ResNet ?

Correction du test :

6) Quel speed-up (ordre de grandeur) peut-on attendre d'un passage sur carte graphique pour l'apprentissage d'un modèle de la classe ResNet ?

facteur 10 à 100

Remarque : cela dépend bien sûr de la durée des autres étapes (ie échantillonnage des lots, augmentation de donnée)

Correction du test :

8) Donner deux qualités qu’on peut attendre d’un jeu de données d’apprentissage.

Correction du test :

8) Donner deux qualités qu'on peut attendre d'un jeu de données d'apprentissage.

Représentativité ; Non redondance.

Mais aussi : qualité des cibles, données équilibrées, etc