IHDCM037 – Machine Learning

Travaux Pratiques

Séance 5 : Réseaux de neurones

Cette séance est consacrée aux réseaux de neurones, et plus particulièrement à leur version la plus simple : le Multi Layer Perceptron. Le jeu de donnée étudié sera MNIST, un des datasets les plus connus dans le domaine du machine learning.

Première partie

Voici les étapes à suivre pour créer votre modèle de classification avec PyTorch :

1. Téléchargez le dataset MNIST à l'aide du module Python `torchvision.datasets.MNIST`. Le dataset est déjà pré-divisé en données d'entraînement (60 000) et de test (10 000), utilisez le paramètre `train=True` ou `False` pour accéder aux données d'entraînement ou de test
2. Créez un loader pour les 2 parties du dataset à l'aide du module `torch.utils.data.Dataloader`. Utilisez le paramètres `batch_size=128` et `shuffle=True`
3. Utilisez la fonction `plot_mnist`, fournie sur WebCampus, pour visualiser des examples du dataset
4. Utilisez le module `torch.nn` pour créer un modèle vierge de MLP, comportant une couche `Flatten()`, une couche `Linear(784,128)`, une couche `ReLU`, une couche `Linear(128,10)` et une couche `LogSoftmax(dim=1)`
5. Initialisez une instance de votre modèle, une fonction de *loss* (`torch.nn.NLLLoss()`), et un optimisateur (`torch.optim.Adam`, avec le paramètre `lr=1e-3`)

Pour plus de documentation, consultez
https://docs.pytorch.org/tutorials/beginner/basics/intro.html

Deuxième partie

Créez une boucle d'entraînement pour votre modèle. Pour ce faire, il faut mettre le modèle en mode *train* (`model.train()`) et placer la boucle dans la macro `with torch.no_grad():`. Pour chaque mini-batch dans le `DataLoader` d'entraînement, les étapes à suivre sont les suivantes :

1. Remettez les gradients du modèle à 0 avec `optimizer.zero_grad()`
2. Calculez les valeurs de sortie pour le mini-batch (*forward pass*)

3. Calculez la valeur de la fonction de *loss* – la distance entre les valeurs de sortie et la *ground truth*
4. Executez un passage de *backpropagation* (`loss.backward()`)
5. Updatez les poids du modèle en executant un « pas » de l'optimisateur (`optimizer.step()`)

Créez une boucle de test pour évaluer la performance de votre modèle. Pour ce faire, il faut mettre le modèle en mode *eval* (`model.eval()`). Pour chaque mini-batch dans le `DataLoader` de test, les étapes à suivre sont les suivantes :

1. Calculez les valeurs de sortie pour le mini-batch (*forward pass*)
2. Déterminez le label prédit en prenant le maximum des valeurs de sortie
3. Comparez ce label avec la *ground truth*

Troisième partie

Variez les paramètres suivants et expliquez vos observations :

- Batch size dans le DataLoader d'entraînement
- Nombre de neurones dans la couche « cachée » (initialement à 128)
- Nombre de couches « cachées » (initialement une seule)
- Nombre d'epochs (nombre de fois que la boucle d'entraînement passe sur l'entièreté de l'ensemble d'entraînement)