TP 3 : Introduction à TF

Jérôme Pasquet

Exercice 1 : Premier réseau avec TF

Nous nous proposons d'implémenter un petit réseau neuronal en utilisant la framework TensorFlow. Nous considérerons le problème de séparation de distributions 2D qui est défini dans l'exercice 2 du TP1.

1. Définir les entrées du problème et implémenter les placeholders
2. Implémenter la première couche. Supposons que celle-ci contienne N_1 neurones, quelle sera la dimension de son tenseur de poids ?
   Rajouter à cette couche un biais et une fonction d'activation de type ReLU.
3. Implémenter la deuxième couche avec N_2 neurones et les mêmes propriétés que la couche 1.
4. La troisième couche est la couche de sortie du réseau. Elle contiendra **un seul neurone** dont la valeur d'activation est 1 pour la classe 1 et 0 pour la classe 0. Utiliser une fonction d'activation apropriée pour effectuer cette opération.
5. Définisser et implémenter votre fonction de perte.
6. Définisser et implémenter votre session et initialisez vos variables.
7. Définissez une base d'entraînement et de validation sur vos données. Puis effectuez l'apprentissage incluant une phase de test périodiquement.
8. Répétez l'expérience en rajoutant des couches si nécessaire. Calculez le nombre de paramètres dans chacun des scénarios.

N_1	N_2	N_3	N_4	$Perf$
8	8	-	-	
32	32	-	-	
128	128	-	-	
32	32	32	-	
8	8	8	8	
32	32	32	32	
64	64	64	64	

Exercice 2 : Premier réseau sur images

Nous allons maintenant réutiliser et adapter notre architecture sur la base de données de mnist. Celle-ci est téléchargeable en effectuant les commandes suivantes :

```
mnist = tf.contrib.learn.datasets.load_dataset("mnist")
train_data = mnist.train.images  # Returns np.array
train_labels = np.asarray(mnist.train.labels, dtype=np.int32)
eval_data = mnist.test.images  # Returns np.array
eval_labels = np.asarray(mnist.test.labels, dtype=np.int32)
```

1. Afficher une image ainsi que son label. Adapter les placeholders.
2. La fonction de perte est-elle encore adaptée ? Modifier là en prenant en compte le type des données.
3. Répetez l'apprentissage défini dans la question 8 de l'exercice 1.

4. Modifier la fonction d'activation en une sigmoid. Dans les scénarios utilisant 4 couches de neurones comparez l'évolution de la fonction de perte et des performances en fonction du type d'activation.

5. Le *data augmentation* consiste à accroître la taille de la base d'apprentissage de manière synthétique en modifiant les données d'entrées. Proposez et implémentez un type ou plusieurs types de data augmentation. Que constatez-vous ?

6. Comparez les temps d'apprentissage avec et sans carte graphique.

7. Sur la couche d'entrée et une couche cachée, calculez les histogrammes d'activation de dix neurones aléatoires lorsque l'on passe en entrée du réseau les images d'un label donné. Dans notre cas, le label transmis sera 1, 7, 8 et 9. Que concluez-vous?

Devoir maison

-> Finissez votre TP pour demain ! **Lisez et comprenez l'article vu en cours.**