ESISA **202...**

Examen : Session normale
Machine Learning and Artificial Intelligence
DL4CV

Questions

1. Dans le contexte de la classification d'images, quels sont les avantages d'un réseau de neurones convolutif (ou CNN) par rapport à un réseau de neurones profond (ou DNN) intégralement connecté?

2. Prenons un CNN constitué de trois couches de convolution, chacune avec des noyaux 33, un pas de 2 et un remplissage "same". La couche inférieure produit 100 cartes de caractéristiques, la couche intermédiaire, 200, et la couche supérieure, 400. L'entrée est constituée d'images RVB de 200×300 pixels:

   (a) Quel est le nombre total de paramètres du CNN ?

   (b) Si l'on utilise des nombres à virgule flottante sur 32 bits, quelle quantité de RAM minimale faut-il à ce réseau lorsqu'il effectue une prédiction pour une seule instance ?

   (c) Qu'en est-il pour l'entraînement d'un mini-lot de cinquante images ?

3. Si la carte graphique vient à manquer de mémoire pendant l'entraînement d'un CNN, quelles sont les cinq actions que vous pourriez effectuer pour tenter de résoudre le problème ?

4. Pourquoi voudriez-vous ajouter une couche de pooling maximum plutôt qu'une couche de convolution avec le même pas ?

5. Quand devriez-vous ajouter une couche de normalisation de réponse locale ?

6. Citez les principales innovations d'AlexNet par rapport à LeNet-5 ? Quelles sont celles de GoogLeNet, de ResNet, de Xception et d'EfficientNet ?

7. Qu'est-ce qu'un réseau entièrement convolutif ? Comment pouvez-vous convertir une couche dense en une couche de convolution ?

8. Quelle est la principale difficulté technique de la segmentation sémantique ?

9. Construisez votre propre CNN à partir de zéro et tentez d'obtenir la meilleure exactitude possible sur le jeu MNIST.

10. Utilisez le transfert d'apprentissage pour la classification de grandes images :

    (a) Créez un jeu d'entraînement contenant au moins 100 images par classe. Vous pouvez, par exemple, classer vos propres photos en fonction du lieu (plage, montagne, ville, etc.), ou utiliser simplement un jeu de données existant (par exemple, venant de TensorFlow Datasets).

    (b) Découpez-le en un jeu d'entraînement, un jeu de validation et un jeu de test.

    (c) Entraînez le modèle sur le jeu d'entraînement et évaluez-le sur le jeu de test.

    (d) Construisez le pipeline d'entrée, appliquez les opérations de prétraitement appropriées, et ajoutez éventuellement une augmentation des données.

    (e) Ajustez un modèle pré-entraîné sur ce jeu de données.