Formation Deep Learning – programme détaillé :

# Introduction :

* **Buts : à la fin de la journée les participants devront avoir compris :**
  + Ce qu’est le deep learning et en quoi il se différencie des outils vu le premier jour
  + Avoir une idée claire de ce qu’est un réseau de neurones et des principaux concepts associés (weights, activation function, input, output, back-propagation algorithm, SGD)
  + De ce qu’est un convolutional network et en quoi ils sont utiles pour l’analyse d’images
  + Des principes de bases en ce qui concerne la conception et l’entrainement d’un réseau (over fitting, optimizer, loss function, training set, testing set, validation set)
  + Avoir vu qq applications
* **Introduction générale au Deep Learning :**
  + Une **forme d’intelligence artificielle** mais, contrairement au machine Learning vu le premier jour, il n’est plus nécessaire de définir les features a priori. Le but du Deep Learning est de trouver de façon itérative quelles sont les features utilies pour répondre au problème posé – reprendre la définition schématique proposée par F.Chollet.
  + Rapide historique de ce domaine. Préciser que c’est un domaine assez ancien mais dont l’essor depuis les années 2000 est essentiellement lié à des améliorations au niveau hardwares (CPU et GPU).
  + **Quelles applications pour l’analyse d’images** :
    - Segmentation
    - Image restoration
    - Prediction
    - Single molecule localization & super resolution
  + **Quand utiliser le Deep Learning**?
    - Applications où les outils classiques d’analyses sont manquants ou inefficaces
    - Nécessiter d’analyser de grandes quantités de données
    - Mettre en garde sur le fait que :
      * nécessaire d’entrainer le réseau et donc d’avoir suffisamment de données pour le faire – suppose souvent un travail préparatoire qui peut être assez long
      * Les réseaux sont souvent très spécialisés – pour chaque nouvelle application, il faudra entrainer le réseau avec de nouvelles données adaptées.
  + Rapide présentation de l’environnement de travail (Anaconda – Keras - Python) et si nécessaire rappeler brièvement ce qu’est une image et un tensor + les notions basiques sur le format nécessaire pour l’image / données que nous allons utiliser (float, normalisée, etc. )

# Qu’est ce qu’un réseau de neurones ?

* **Définition d’un neurone et introduction à Keras :**
  + Présenter un schéma d’un neurone simple en introduisant les concepts associés de input, weight, activation function, biais, output :
    - Préciser quelle est l’opération réalisée par le neurone
    - Que se passe t’il pendant l’entrainement ? Ajuster les paramètres weights & biais. Préciser que cette opération est en fait invisible puisque nous n’aurons accès qu’aux inputs/outputs
    - Rappeler brièvement le principe du SGD (qui aura été introduite le jour d’avant par Volker)
  + Example n°1 : classification de deux jeux de données séparés **linéairement**. Le but de cet exercice est d’illustrer rapidement ce qu’un seul neurone peut permettre de faire. Au travers de cet exercice nous introduirons les notions :
    - d’optimizer – le plus simple SGD
    - de Loss function – dans ce cas, binary cross entropy
    - des différentes «  activation function » (sigmoide)
    - de epochs
    - de classes et de classification

Pendant l’exercice, utiliser les schémas proposer par F.Chollet p10-11 qui permettent d’illustrer simplement le fonctionnement des algorithmes. Par ailleurs, commencer à introduire Keras et comment rapidement définir l’architecture du réseau.

* **Définition d’un réseau avec plusieurs couches :**
  + Example n°2 : aller un peu plus loin avec un jeu de données où les deux classes ne sont plus séparables linéairement. L’idée est cette fois d’introduire la nécessité de travailler en ajoutant des couches supplémentaires permettant au réseau d’apprendre des concepts complexe.
    - Premier test avec 1 seul neurone
    - Construire un nouveau réseau avec 4 neurones
    - Tester l’effet de la fonction Relu
    - Montrer l’intérêt du mini-batch
  + Rapidement dire que s’ils souhaitent aller plus loin, il y a l’outil de Tensor flow en ligne pour tester les effets des différentes activation, algortihme, etc…
  + Example n°3 : test avec MNIST ( ?). L’idée est d’introduire avec cet exemple les notions :
    - D’overfitting – comment le déceler, en quoi est-ce important de le détecter. **Essayer de voir si on peut introduire artificiellement de l’overfitting en sélectionnant un petit training set.**
    - Comment définir les training/testing/validation sets – insister sur le fait que chacun de ces sets de données doivent être indépendants
    - Quelles sont les techniques pour éviter l’overfitting : régularization – drop out – simplifier l’architecture du réseau – ajouter des données à l’entrainement

# Convolutional neural network :

* **Introduction au convNet :**
  + Rapidement définir la notion de convolution et de filtres.
  + Présenter l’avantage de ces réseaux par rapport au densely connected layers :
    - Densely connected pour l’image, on est obligé de linéariser l’image, donc de perdre la notion de contexte spatiale
    - Le convNet travaille directement sur l’image et de façon locale. Il est donc adaptable à toute taille d’images. **Dans ce cas, ce qu’on entraine sont les filtre**s. Si on revient à la comparaison avec le machine learning, le réseau apprendra par lui-même les features
  + Example n°4 : MNIST mais cette fois pour introduire le convNet
  + Example n°5 : Application de Francesco pour le calcul des positions des billes