Intelligence Artificielle 3

Cours 2 - Création de modèles ML/DL

Table des matières

Définition

Un réseaux de neurones est normalement séquentiel puisque les couches sont interconnectées en chaîne du début (input) jusqu'à la fin (output).

Le modèle peut avoir un nombre arbitraire de couches importées par le programmeur.

Ceci-dit, il y a des règles de base à respecter au niveau de la conception des couches :

- Entrées
- Interconnections
- Sortie(s)

Définition - Entrée

Entrée : La taille d'entrée du réseau de neurones (ML ou DL) doit être la même que la donnée d'entrée que l'on veut prédire le résultat.

Les colonnes (données tabulaires) et les pixels (images) sont tous les deux des **features (caractéristiques)** !!!

Par exemple :

- **Real Estate** : Le bon nombre de critères de l'appartement (ici les colonnes qui décrivents la propriété).
- **MNIST** : Le bon nombre de pixels qui compose l'image (ici les valeurs de la matrice sont des pixels).

Définition - Entrée

```
clim = clim.reset_index()
clim.head()
```

	Date Time	p (mbar)	T (degC)	Tpot (K)	Tdew (degC)	rh (%)	VPmax (mbar)	VPact (mbar)	VPdef (mbar)	sh (g/kg)	H2OC (mmol/mol)	rho (g/m**3)	wv (m/s)	max. wv (m/s)	wd (deg)
0	01.01.2009 00:10:00	996.52	-8.02	265.40	-8.90	93.3	3.33	3.11	0.22	1.94	3.12	1307.75	1.03	1.75	152.3
1	01.01.2009 00:20:00	996.57	-8.41	265.01	-9.28	93.4	3.23	3.02	0.21	1.89	3.03	1309.80	0.72	1.50	136.1
2	01.01.2009 00:30:00	996.53	-8.51	264.91	-9.31	93.9	3.21	3.01	0.20	1.88	3.02	1310.24	0.19	0.63	171.6
3	01.01.2009 00:40:00	996.51	-8.31	265.12	-9.07	94.2	3.26	3.07	0.19	1.92	3.08	1309.19	0.34	0.50	198.0
4	01.01.2009 00:50:00	996.51	-8.27	265.15	-9.04	94.1	3.27	3.08	0.19	1.92	3.09	1309.00	0.32	0.63	214.3

Figure: https://medium.com/@vineet.pandya/use-tensorflow-lstm-for-time-series-forecasting-770ec789d2ce

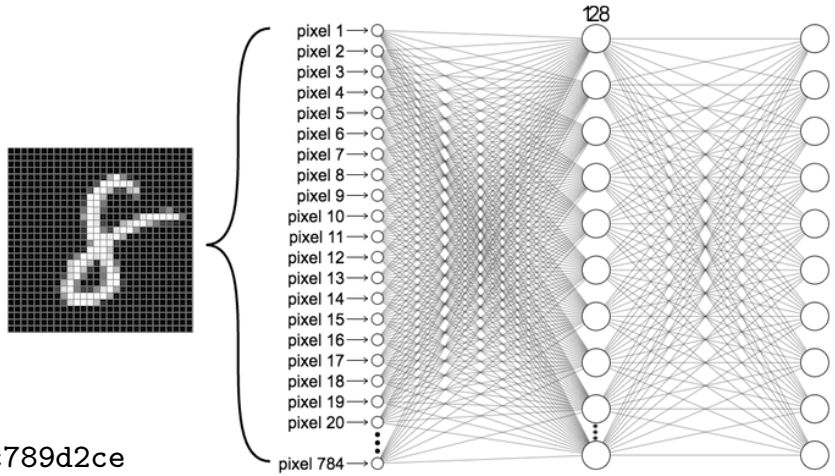


Figure: https://colab.research.google.com/github/aamini/introtodeeplearning/blob/master/lab2/Part1_MNIST.ipynb

Définition - Interconnections

Interconnections : Chaque couche interconnectée est normalement supposé ne pas diluer l'extrant à un point où il n'existe plus rien à la sortie.

De plus, chaque couche qui se suit doit pouvoir être mutuellement compatible avec sa précédente.

Même principe que l'entrée, où une image de mauvaise taille ne rentre pas dans la première couche puisque la taille ne concorde pas.

Définition - Sortie(s)

Sortie(s) : La taille, le type et son expression (ce que la réponse veut dire) doit être lisible par le logiciel et/ou la personne qui utilise le résultat en question.

Aussi, elle doit respecter les propriétés théoriques restreintes par les types de couches disponibles à ce jour (voir la documentation appropriée).

Par exemple :

- **Real Estate** : Une valeur continue représentant le coût de la propriété (i.e. 521045).
- **MNIST** : Un tableau d'activation booléen de taille N (ici 10 puisque 10 classes) représentant qu'elle classe est déduite par le modèle.

Définition - Trouver le modèle optimal

Comment créer un modèle optimal ?

Par soi-même : Essayer plusieurs possibilités de modèles avec des couches différentes et des paramètres (i.e. dropout) différents. Ensuite, comparer les évaluation et prendre le meilleur.

Définition - Trouver le modèle optimal

La meilleure méthode pour trouver des topologies : Lire des Notebooks de spécialistes (i.e. Kaggle), des sites de documentation officielle, et l'état de l'art actuel récent (i.e. articles de conférences et de journaux) pour avoir des topologies de modèles fait par des experts (chercheurs académiques/industriels, étudiants à la maîtrise, PhD, Postdoc, etc.) avec des résultats concrets prouvés dans l'expérience publiée de l'article (évalué par les pairs) en question.

Définition - Trouver le modèle optimal

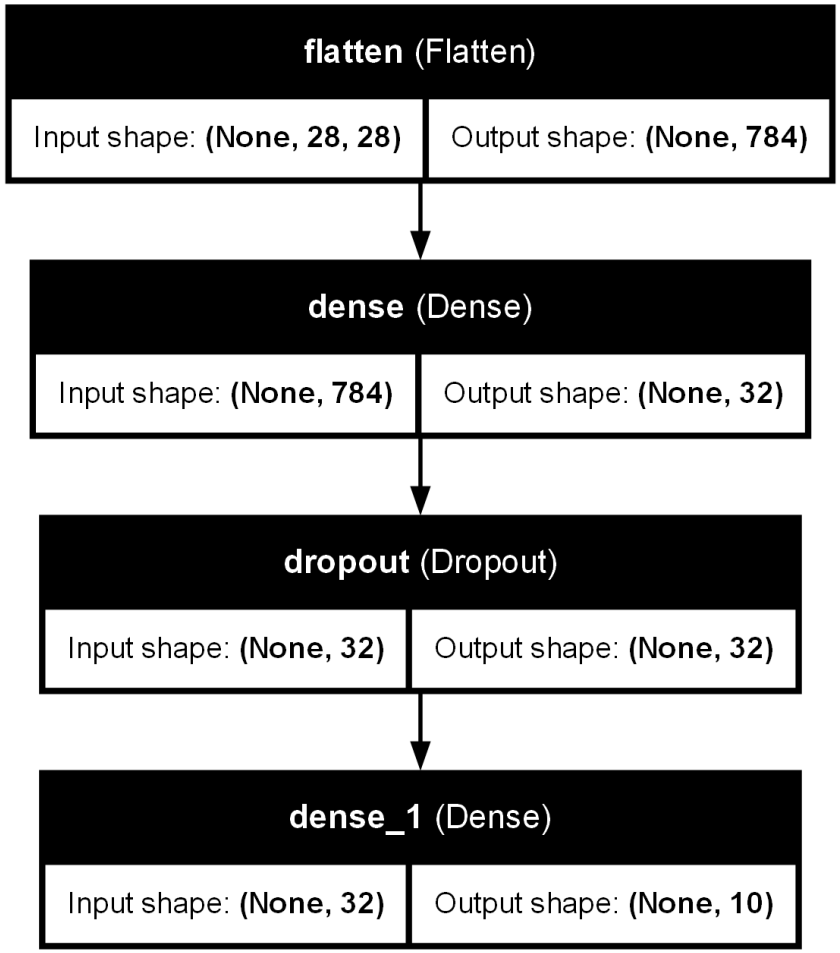
NB 1 : Des modèles différents peuvent bien tous donner des résultats (prédictions) acceptable.

NB 2 : Il est important de tester et de regarder le code que vous trouver sur des blogs “obscurs” ou provenant de ChatGPT, ils sont généralement de confiance basse et non validé par les pairs de la communauté. **Ils valent tout de même la peine d’être testés/considérés si vous voyez qu’ils font du sens.**

Définition - Rappel

```
1  # Copyright 2020 Google LLC. https://www.apache.org/licenses/LICENSE-2.0
2
3  # All models in this course are sequential.
4  model = tf.keras.models.Sequential()
5
6  # The features are stored in a two-dimensional 28X28 array. Flatten that
7  # two-dimensional array into a one-dimensional 784-element array.
8  model.add(tf.keras.layers.Flatten(input_shape=(28, 28)))
9
10 # Define the first hidden layer.
11 model.add(tf.keras.layers.Dense(units=32, activation='relu'))
12
13 # Define a dropout regularization layer.
14 model.add(tf.keras.layers.Dropout(rate=0.2))
15
16 # Define the output layer. The units parameter is set to 10 because
17 # the model must choose among 10 possible output values (representing
18 # the digits from 0 to 9, inclusive).
19 # Don't change this layer.
20 model.add(tf.keras.layers.Dense(units=10, activation='softmax'))
```

Couches


flatten (Flatten)
Input shape: (None, 28, 28)
Output shape: (None, 784)
dense (Dense)
Input shape: (None, 784)
Output shape: (None, 32)
dropout (Dropout)
Input shape: (None, 32)
Output shape: (None, 32)
dense_1 (Dense)
Input shape: (None, 32)
Output shape: (None, 10)

Couches

Pour plus d'information, il est possible de voir la documentation de Keras avec les explications de chaque type de couche séquentielle.

Flatten :
`https://keras.io/api/layers/reshaping_layers/flatten/`

Architectures

Ceci vient d'une documentation officielle, mais il faut quand même le tester.

NB : Code n'est pas en Python, le but est de montrer les poids (weights) et comment les images sont dilluées tout au long du pipeline.

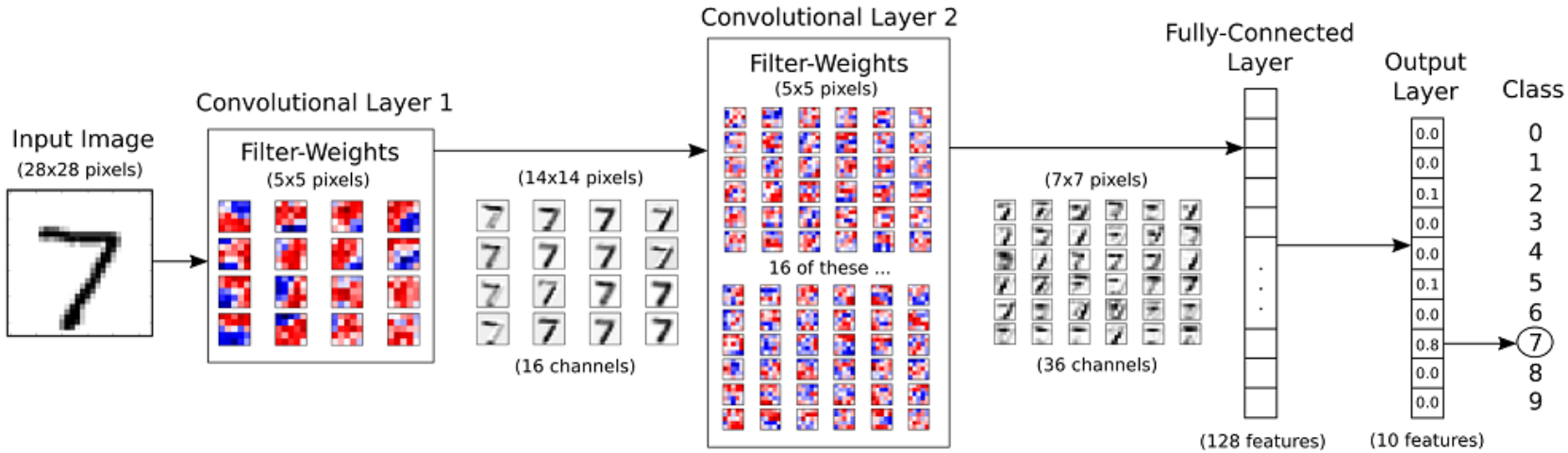


Figure: `https://tensorflownet.readthedocs.io/en/latest/ConvolutionNeuralNetwork.html`

Architectures

Ceci vient d'un blog, donc il faudrait impérativement le tester...

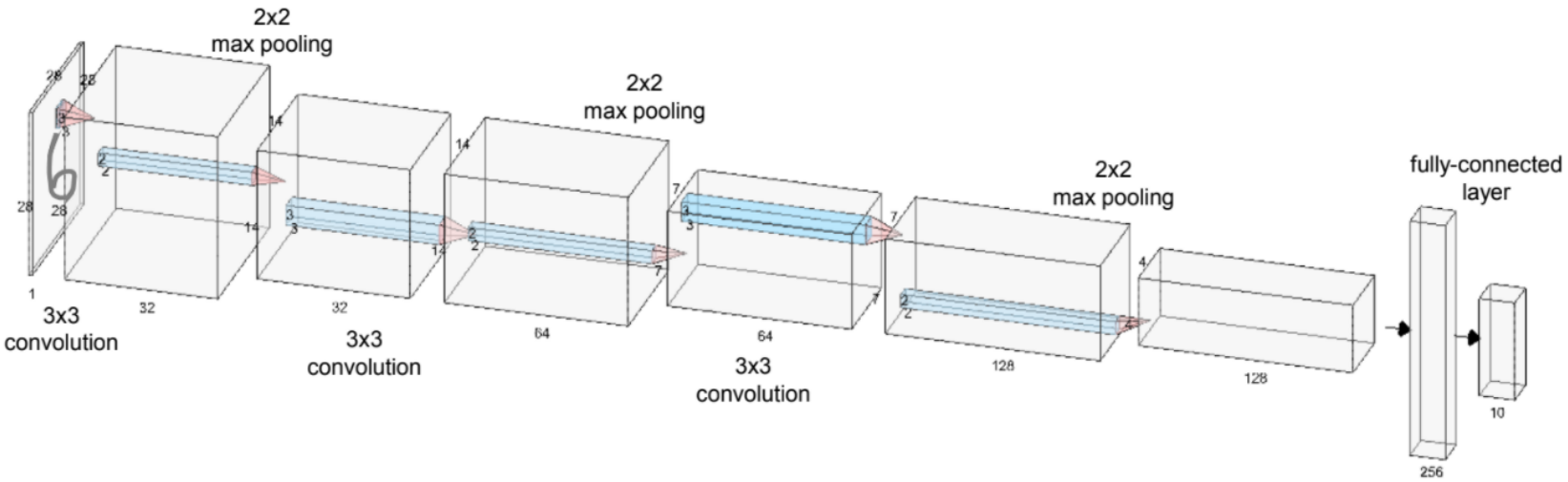


Figure: `https://goodboychan.github.io/python/deep_learning/tensorflow-keras/2020/10/10/01-CNN-with-MNIST.html`

Bibliographie

- https://keras.io/api/layers/reshaping_layers/flatten/
- https://keras.io/api/layers/reshaping_layers/flatten/
- https://goodboychan.github.io/python/deep_learning/tensorflow-keras/2020/10/10/01-CNN-with-MNIST.html
- https://colab.research.google.com/github/aamini/introtodeeplearning/blob/master/lab2/Part1_MNIST.ipynb
- https://medium.com/@vineet.pandya/use-tensorflow-lstm-for-time-series-forecasting-770ec7