D. Makowski
Université Paris-Saclay
INRAE

Outline

- Definition & main principles
- Several extensions of linear regression
- Trees and forests
- Deep learning

What is it?

- Often considered as a special type of machine learning adapted to
  - Very large datasets
  - Unstructured input data (images, texts, sounds etc.)
- Based on neural network algorithms
- Inputs are replaced by a large number of « features » summarizing the characteristics of the original inputs
- These « features » are automatically generated by the algorithms

Application: recognition of numbers

- 60,000 images of numbers available for training of a predictive algorithm
- 10,000 number images available for testing the trained algorithm

Upload images

```
#Images
x_train <- mnist$train$x
#Etiquettes
y_train <- mnist$train$y

#Images
x_test <- mnist$test$x
x_test_image <- mnist$test$x
#Labels
y_test <- mnist$test$y
```

Training

Test

One image of x_train and its label in y_train

```
#Example of number
plot(as.raster(x_train[26,,]), max=255)
y_train[26]
```

```
> y_train[26]
[1] 2
```

Another image x_train and its lable y_train

```
#Exemple chiffre
plot(as.raster(x_train[62,,]), max=255)
image(1:28,1:28,NUM)
y_train[62]
```

```
> y_train[62]
[1] 4
```

```
> dim(x_train)
[1] 60000    28    28
> x_train[1,,]
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14] [,15] [,16] [,17]
 [1,]    0    0    0    0    0    0    0    0    0     0     0     0     0     0     0     0     0
 [2,]    0    0    0    0    0    0    0    0    0     0     0     0     0     0     0     0     0
 [3,]    0    0    0    0    0    0    0    0    0     0     0     0     0     0     0     0     0
 [4,]    0    0    0    0    0    0    0    0    0     0     0     0     0     0     0     0     0
 [5,]    0    0    0    0    0    0    0    0    0     0     0     0     0     0     0     0     0
 [6,]    0    0    0    0    0    0    0    0    0     0     0     0     3    18    18    18   126
 [7,]    0    0    0    0    0    0    0    0   30    36    94   154   170   253   253   253   253
 [8,]    0    0    0    0    0    0    0   49  238   253   253   253   253   253   253   253   253
 [9,]    0    0    0    0    0    0    0   18  219   253   253   253   253   253   198   182   247
[10,]    0    0    0    0    0    0    0    0   80   156   107   253   253   205    11     0    43
[11,]    0    0    0    0    0    0    0    0    0    14     1   154   253    90     0     0     0
[12,]    0    0    0    0    0    0    0    0    0     0     0   139   253   190     2     0     0
[13,]    0    0    0    0    0    0    0    0    0     0     0    11   190   253    70     0     0
[14,]    0    0    0    0    0    0    0    0    0     0     0     0    35   241   225   160   108
[15,]    0    0    0    0    0    0    0    0    0     0     0     0     0    81   240   253   253
[16,]    0    0    0    0    0    0    0    0    0     0     0     0     0     0    45   186   253
[17,]    0    0    0    0    0    0    0    0    0     0     0     0     0     0     0    16    93
```

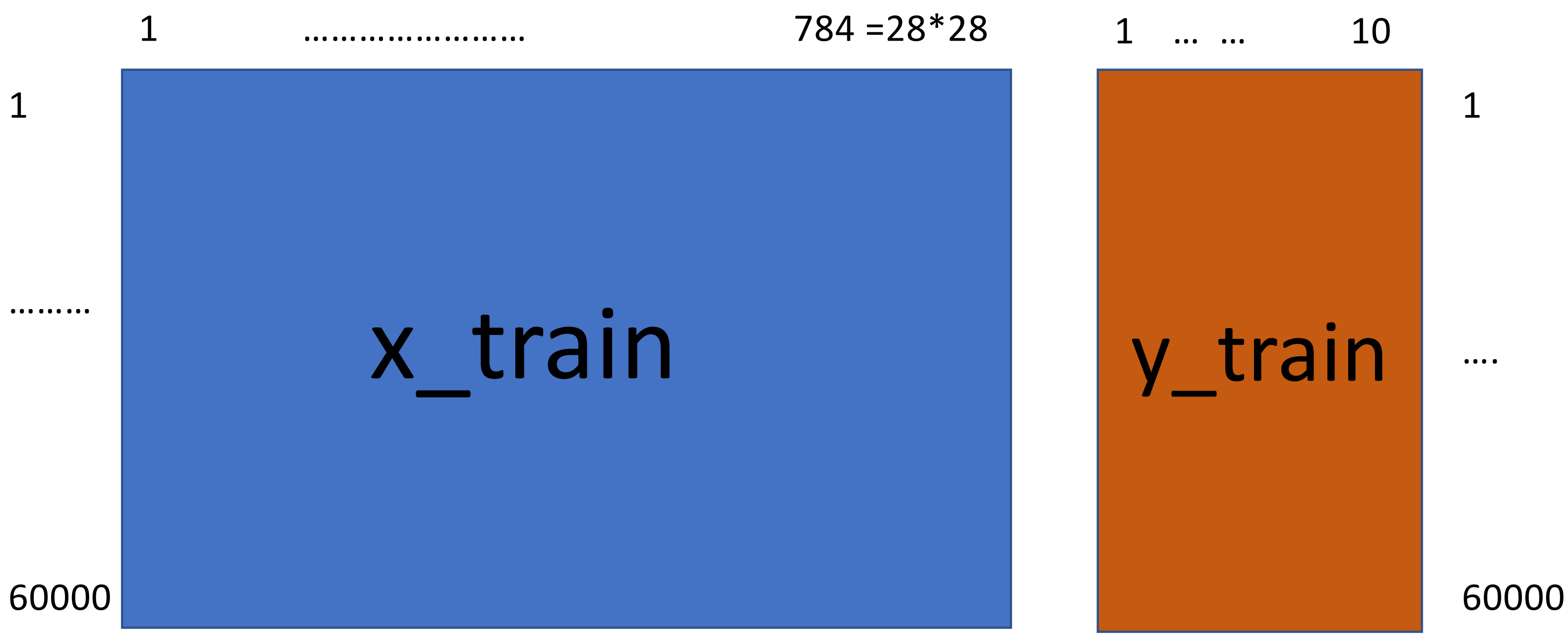
```r
# reshape
x_train <- array_reshape(x_train, c(nrow(x_train), 784))
x_test <- array_reshape(x_test, c(nrow(x_test), 784))
# rescale
x_train <- x_train / 255
x_test <- x_test / 255

y_train <- to_categorical(y_train, 10)
y_test <- to_categorical(y_test, 10)
```

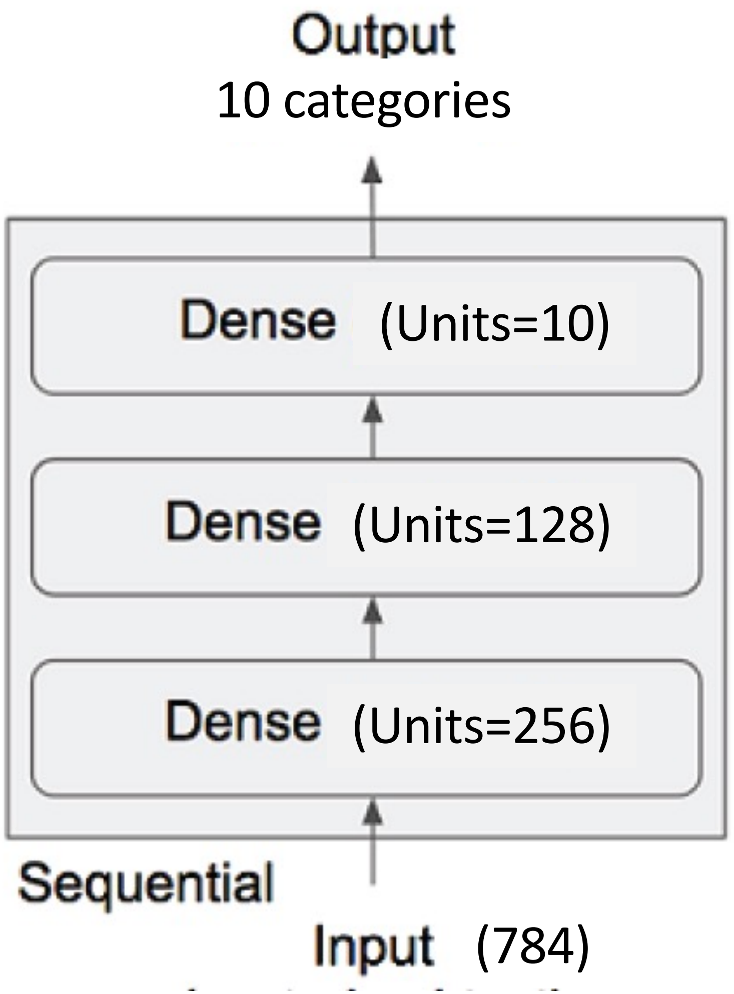
```
> dim(x_train)
[1] 60000   784
```

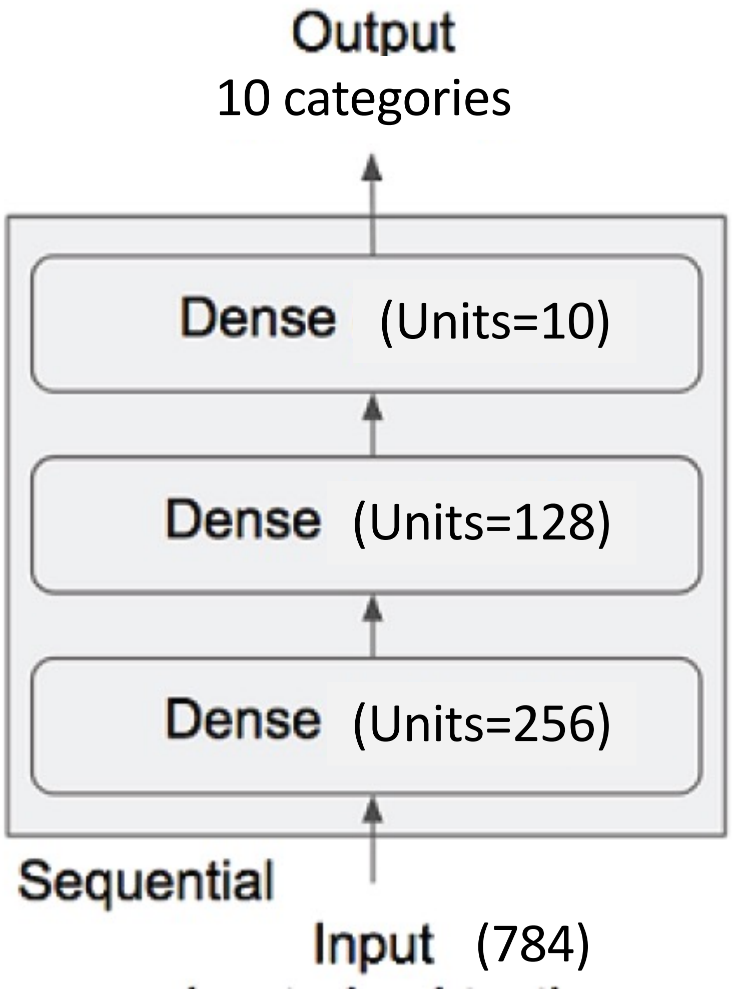
```
       [,129]    [,130]    [,131]    [,132] [,133] [,134] [,135] [,136] [,137] [,138] [,139]
[1,] 0.0000000 0.0000000 0.0000000 0.0000000      0      0      0      0      0      0      0
[2,] 0.6235294 0.9921569 0.6235294 0.1960784      0      0      0      0      0      0      0
[3,] 0.0000000 0.0000000 0.0000000 0.0000000      0      0      0      0      0      0      0
[4,] 0.0000000 0.0000000 0.0000000 0.0000000      0      0      0      0      0      0      0
[5,] 0.0000000 0.0000000 0.0000000 0.0000000      0      0      0      0      0      0      0
[6,] 0.0000000 0.0000000 0.0000000 0.0000000      0      0      0      0      0      0      0
```

```
> dim(y_train)
[1] 60000    10
> head(y_train)
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    0    0    0    0    0    1    0    0    0     0
[2,]    1    0    0    0    0    0    0    0    0     0
[3,]    0    0    0    0    1    0    0    0    0     0
[4,]    0    1    0    0    0    0    0    0    0     0
[5,]    0    0    0    0    0    0    0    0    0     1
[6,]    0    0    1    0    0    0    0    0    0     0
```


1
……………………
784 =28*28
1
………
60000
x_train
1 … … 10
y_train
1
….
60000

Neural network with several layers



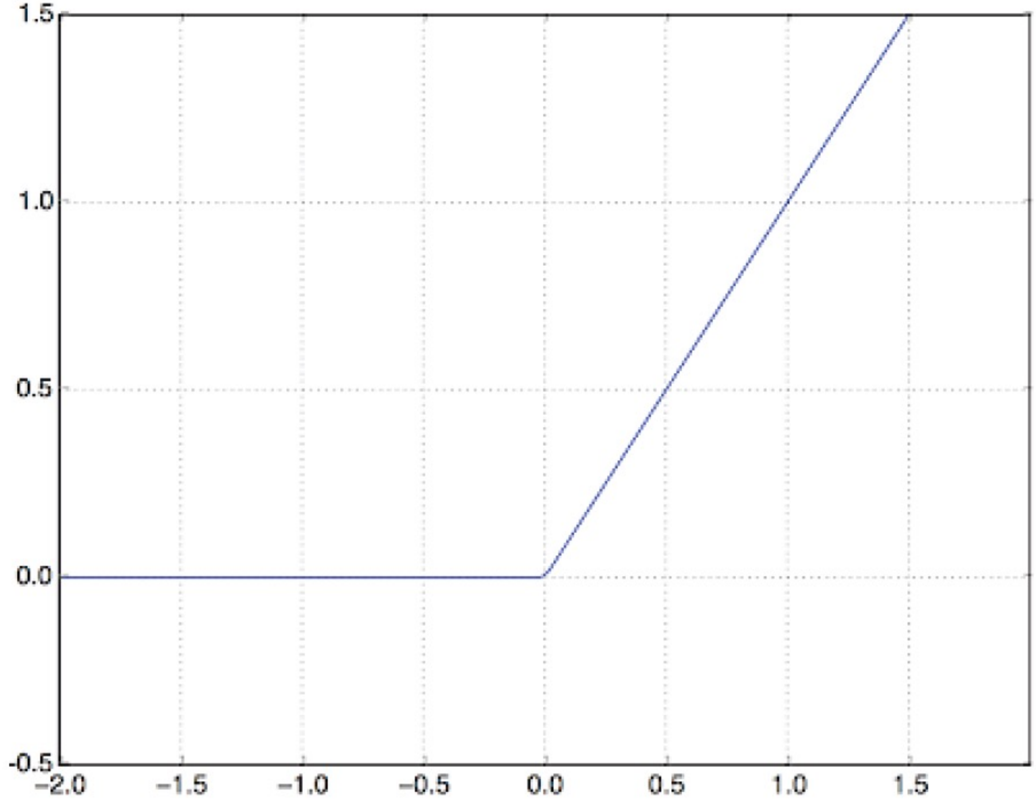


$$f\left(b3+\sum_{k=1}^{128} w_{k,3} * X_{k,3}\right) \qquad *10$$

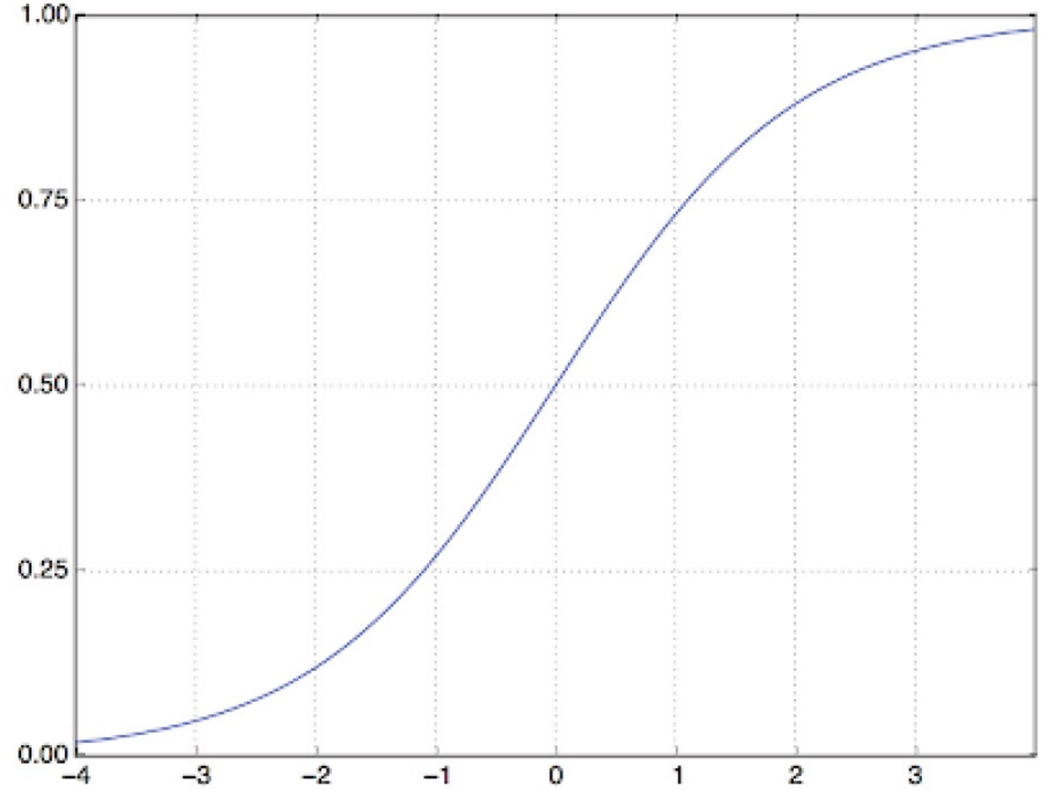
$$f\left(b2+\sum_{j=1}^{256} w_{j,2} * X_{j,2}\right) \qquad *128$$

$$f\left(b1+\sum_{i=1}^{784} w_{i,1} * X_{i,1}\right) \qquad *256$$

f=RELU



f=softmax



Neural network with several layers

```
model <- keras_model_sequential()
model %>%

  layer_dense(units = 256, activation = 'relu',
       input_shape = c(784)) %>%

  layer_dropout(rate = 0.4) %>%

  layer_dense(units = 128, activation = 'relu') %>%

  layer_dropout(rate = 0.3) %>%

  layer_dense(units = 10, activation = 'softmax')

summary(model)
```

Many parameters

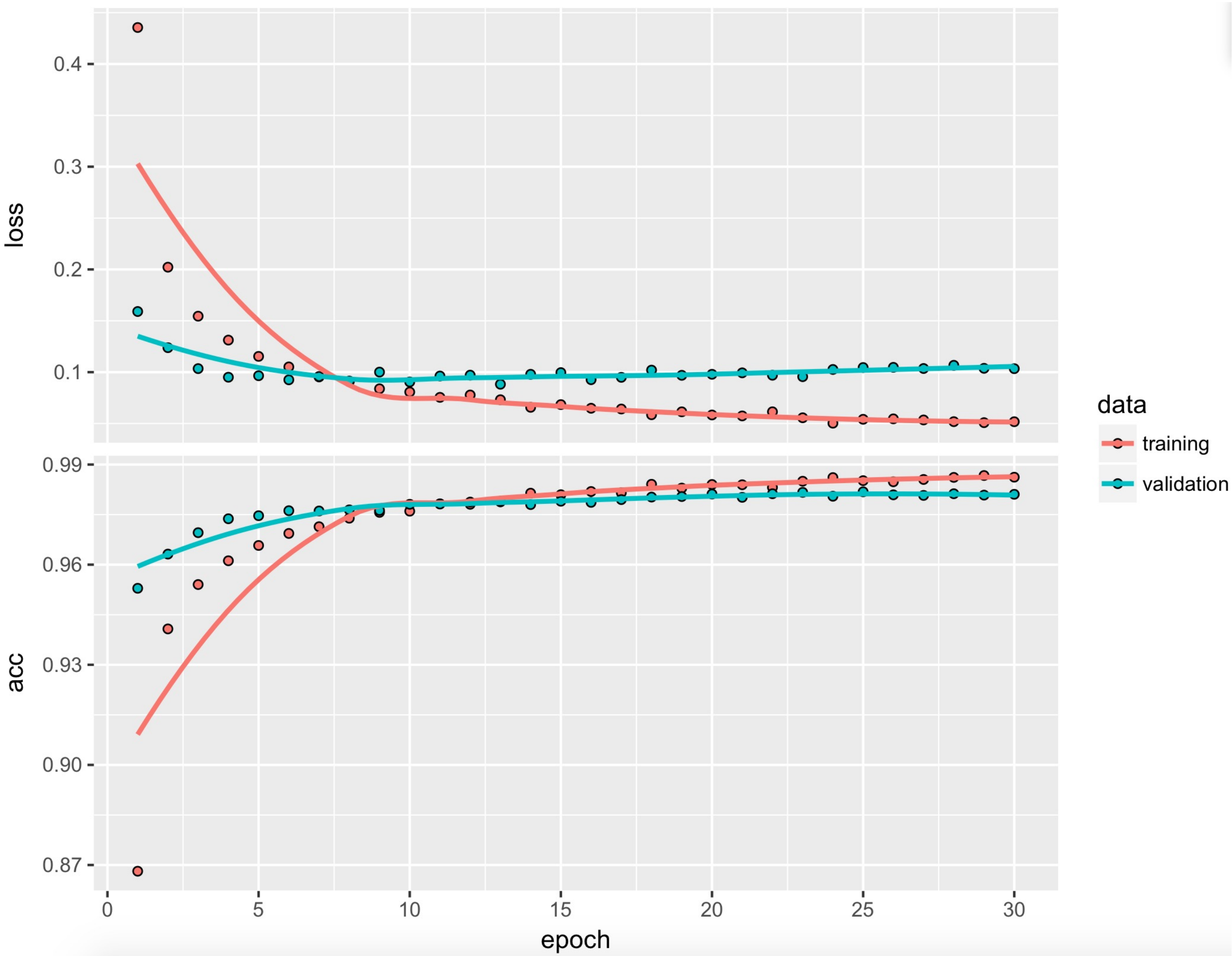
```
> summary(model)
________________________________________________________________________________
Layer (type)                      Output Shape                   Param #
================================================================================
dense_1 (Dense)                   (None, 256)                    200960
________________________________________________________________________________
dropout_1 (Dropout)               (None, 256)                    0
________________________________________________________________________________
dense_2 (Dense)                   (None, 128)                    32896
________________________________________________________________________________
dropout_2 (Dropout)               (None, 128)                    0
________________________________________________________________________________
dense_3 (Dense)                   (None, 10)                     1290
================================================================================
Total params: 235,146
Trainable params: 235,146
Non-trainable params: 0
```

Model training

```
model %>% compile(
  loss = 'categorical_crossentropy',
  optimizer = optimizer_rmsprop(),
  metrics = c('accuracy')
)


history <- model %>% fit(
  x_train, y_train,
  epochs = 30, batch_size = 128,
  validation_split = 0.2
)

plot(history)
```



loss
acc
epoch
data
training
validation
0.4
0.3
0.2
0.1
0.99
0.96
0.93
0.90
0.87
0
5
10
15
20
25
30

Model testing

```
##Evaluation
results<- model%>%evaluate(x_test,y_test)
results
```

```
> results
$loss
[1] 0.1017851

$acc
[1] 0.981
```

```
> Prediction<-model %>% predict_classes(x_test)
> #Exemple
> k<-780
> plot(as.raster(x_test_image[k,,], max=255))
> y_test[k,]
 [1] 0 0 0 0 0 1 0 0 0 0
> Prediction[k]
[1] 5
```

Multi-layer neural network with **convolutions**

What is a convolution?

Convolution=Filter

- Each filter « slides » over the object (e.g. over the pixels of an image) to create a new object called « feature map »

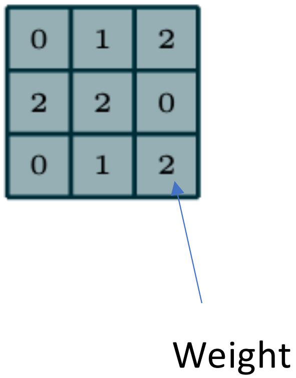
- A feature map shows a specific aspect of an object (e.g., one part of an image)

- Each filter is charaterized by a size and by weights
  - ✓The size is pre-specified,
  - ✓the weights are optimized like the other model parameters

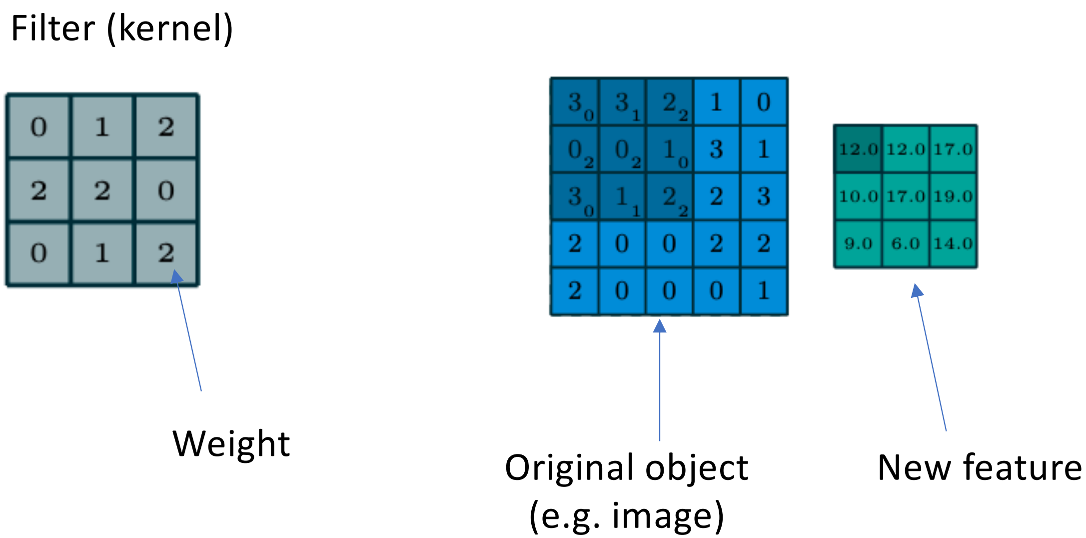
- The generated feature map is used as a new source of inputs for the neural network

- Several filters are generally used successively, leading to several feature maps

Filter (kernel)

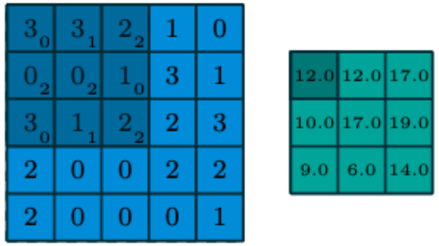


Dumoulin and Visin 2018. https://arxiv.org/pdf/1603.07285.pdf



Dumoulin and Visin 2018. https://arxiv.org/pdf/1603.07285.pdf

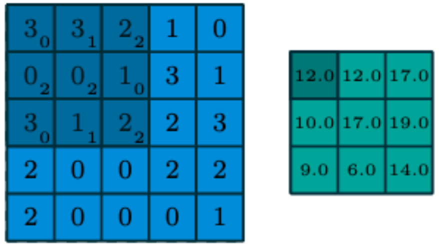
Filter (kernel)

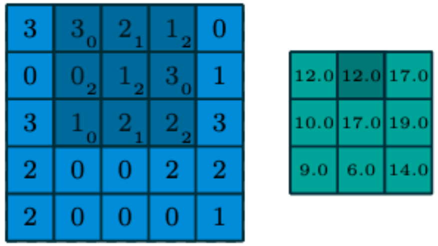


Dumoulin and Visin 2018. https://arxiv.org/pdf/1603.07285.pdf

Filter (kernel)

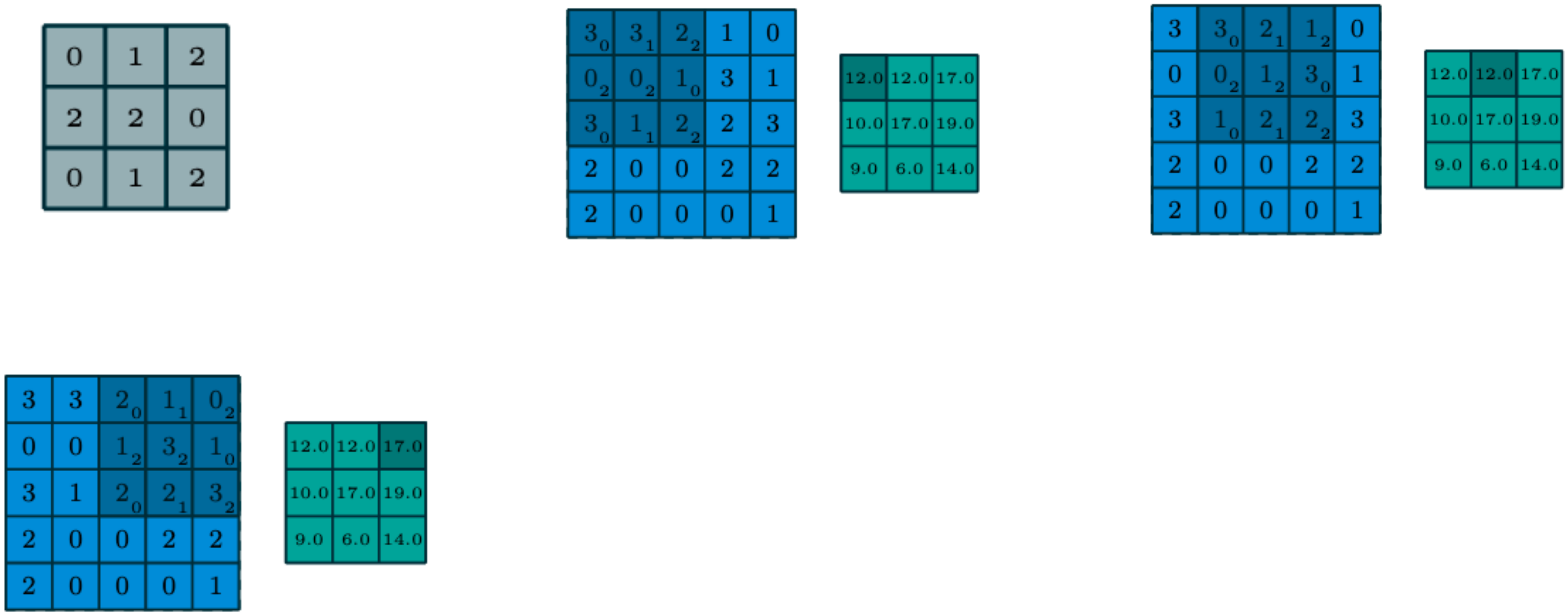
0	1	2
2	2	0
0	1	2





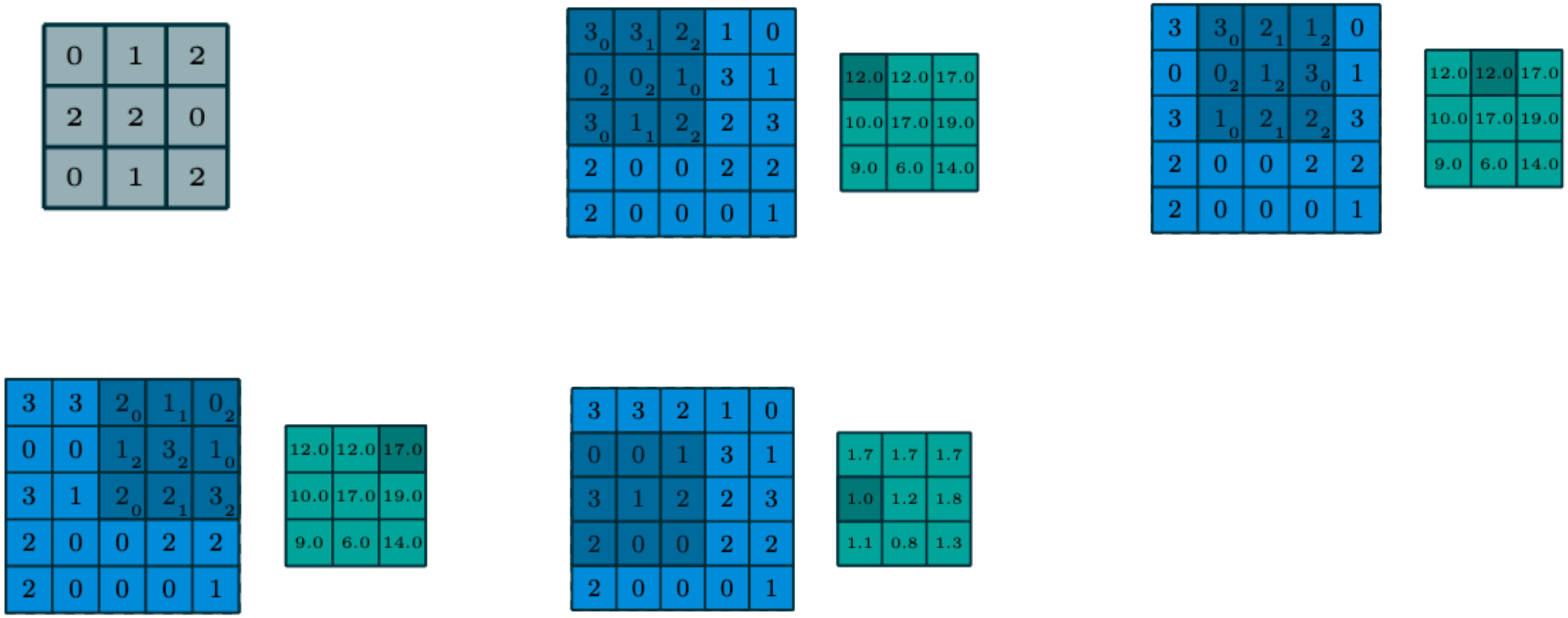
Dumoulin and Visin 2018. https://arxiv.org/pdf/1603.07285.pdf

Filter (kernel)



Dumoulin and Visin 2018. https://arxiv.org/pdf/1603.07285.pdf

Filter (kernel)



Dumoulin and Visin 2018. https://arxiv.org/pdf/1603.07285.pdf

```r
x_train <- mnist$train$x
y_train <- mnist$train$y
x_test <- mnist$test$x
y_test <- mnist$test$y


# rescale
x_train <- x_train / 255
x_test <- x_test / 255


x_train<-array(x_train,dim=c(60000,28,28,1))
x_test<-array(x_test,dim=c(10000,28,28,1))


y_train <- to_categorical(y_train, 10)
y_test <- to_categorical(y_test, 10)
```

```r
model <- keras_model_sequential()
model %>%
  layer_conv_2d(filters = 32, kernel_size = c(3,3), activation = 'relu',
                input_shape = c(28,28,1)) %>%
  layer_conv_2d(filters = 32, kernel_size = c(3,3), activation = 'relu') %>%
  layer_max_pooling_2d(pool_size = c(2,2)) %>%
  layer_dropout(rate = 0.25) %>%
  layer_conv_2d(filters = 64, kernel_size = c(3,3), activation = 'relu') %>%
  layer_conv_2d(filters = 64, kernel_size = c(3,3), activation = 'relu') %>%
  layer_max_pooling_2d(pool_size = c(2,2)) %>%
  layer_dropout(rate = 0.25) %>%
  layer_flatten() %>%
  layer_dense(units = 256, activation = 'relu') %>%
  layer_dropout(rate = 0.25) %>%
  layer_dense(units = 10, activation = 'softmax')

summary(model)
```

```r
model %>% compile(
  loss = 'categorical_crossentropy',
  optimizer = optimizer_rmsprop(),
  metrics = c('accuracy')
)


history <- model %>% fit(
  x_train, y_train,
  epochs = 2, batch_size = 128,
  validation_split = 0.2
)


plot(history)
```

```
> Prediction<-model %>% predict_classes(x_test)
> #Exemple
> k<-780
> plot(as.raster(x_test_image[k,,], max=255))
> y_test[k,]
 [1] 0 0 0 0 0 1 0 0 0 0
> Prediction[k]
[1] 5
```

More and more applications

- Classification of objects (numbers, plants, animals, words etc.)
- Diagnosis of diseases
- Weather forecasts
- Automatic translation of text

ARTICLE OPEN

Deep learning enables robust assessment and selection of human blastocysts after in vitro fertilization

Pegah Khosravi[1,2], Ehsan Kazemi[3], Qiansheng Zhan[4], Jonas E. Malmsten[4], Marco Toschi[4], Pantelis Zisimopoulos[1,2], Alexandros Sigaras[1,2], Stuart Lavery[5], Lee A. D. Cooper[6], Cristina Hickman[5], Marcos Meseguer[7], Zev Rosenwaks[4], Olivier Elemento[1,2,8], Nikica Zaninovic[4] and Iman Hajirasouliha[1,2]

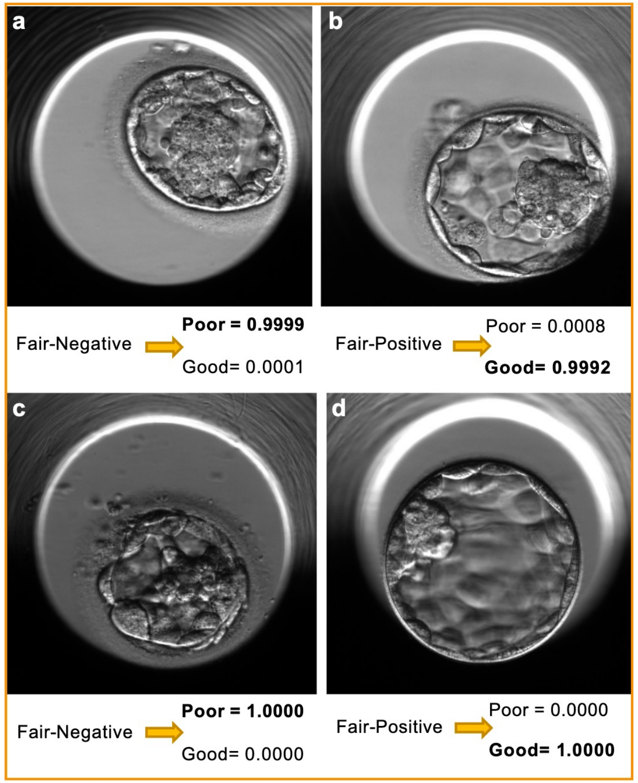


Fig. 4 STORK vs. embryologists classification: STORK classifies the fair-quality images into existing good-quality and poor-quality classes. For example, panels "**a**" and "**b**" are labeled 3A-B (fair-quality) according to the Veeck and Zaninovic grading system, while STORK classified them as poor-quality and good-quality, respectively. Also, panels "**c**" and "**d**" are both labeled 3BB (fair-quality). However, the algorithm correctly classified panel "**c**" as poor-quality and panel "**d**" as good-quality. As the figure shows, the outcome in the embryos in "**b**" and "**d**" is positive live birth, whereas it is negative live birth in "**a**" and "**c**"

https://doi.org/10.1038/s41746-019-0096-y

Outline

- Brief overview of machine learning
- Trees and forests
- Deep learning
- **Conclusion**

Main challenges in machine learning projects

- Choose a relevant question (Which Y? Which X?)
- Find reliable data
- Calibrate the hyper-parameters
- Assess prediction accuracy without bias
- Optimize computation time
- Vizualisation of output responses

Start simple

Start with two simple methods:

- Penalized linear regression (ex: LASSO)
- Random forest

Some trends

- Visualization tools (to open « the black boxes »)
- Image and text analyses (text mining, deep learning)
- Packages to streamline the development of predictive models (keras, caret, H2O…)
- Including expert knowledge in machine learning

Machine learning to emulate complexe models

Contents lists available at ScienceDirect

Journal of Environmental Management

journal homepage: www.elsevier.com/locate/jenvman

Research article

Meta-modeling methods for estimating ammonia volatilization from nitrogen fertilizer and manure applications

Maharavo Marie Julie Ramanantenasoa[a,b], Sophie Génermont[a,*], Jean-Marc Gilliot[a], Carole Bedos[a], David Makowski[c,d]

ACCEPTED MANUSCRIPT • OPEN ACCESS

Maize yield and nitrate loss prediction with machine learning algorithms

Mohsen Shahhosseini[1], Rafael A Martinez-Feria[2], Guiping Hu[3] and Sotirios Archontoulis[1]

Accepted Manuscript online 29 October 2019 •