DataScientest • com

Rakuten

Présentation du projet eShopPye

Equipe :

Nada STAOUITE
Bastien PIQUEREAU
Lucas GANDY

Mentor :

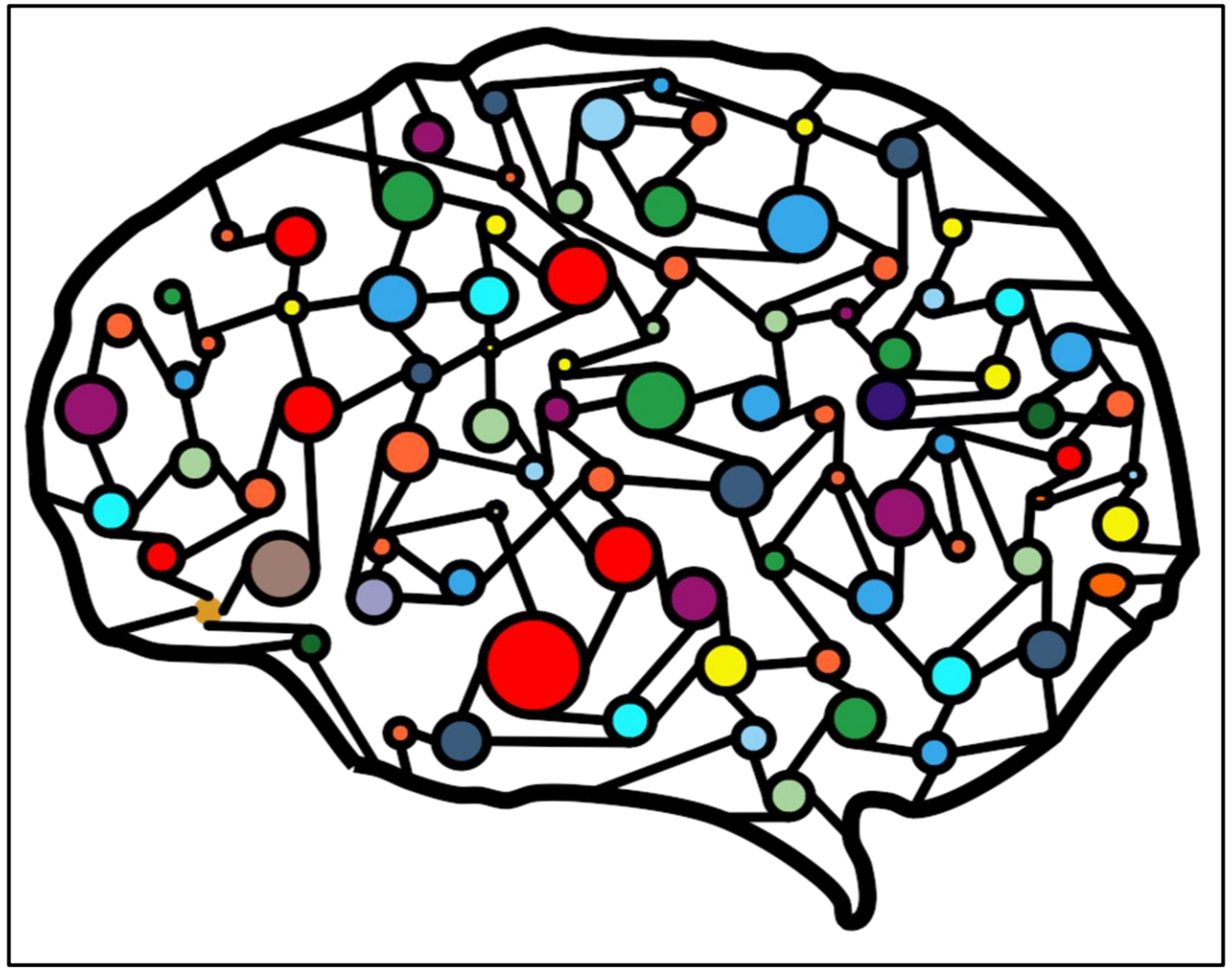
Chloé GUIGA

Promotion :

Bootcamp Décembre 2020

Plan

1. Introduction
2. Présentation des données
3. Pré traitement
4. Modèles et performances
    1. Image
    2. Texte
    3. Bimodale
5. Conclusion et perspectives d'amélioration

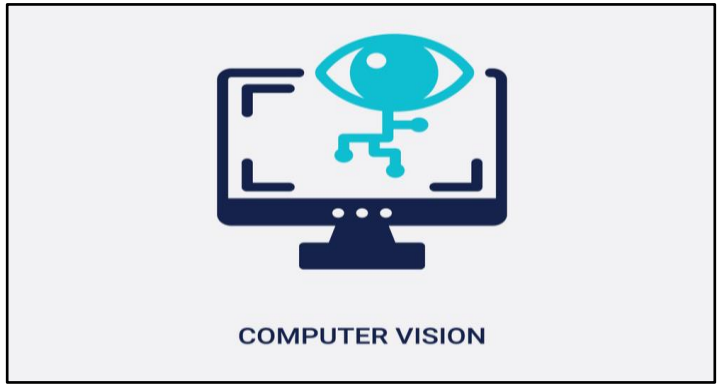


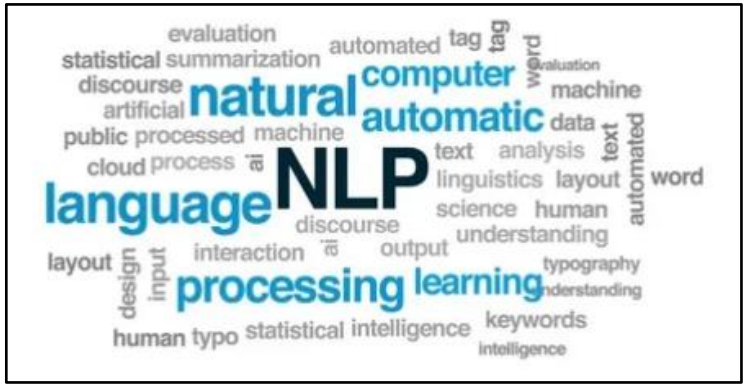
1. Introduction

- Projet : Challenge Rakuten France Multimodal Product Data Classification

- Objectif : classification des articles à partir des informations textuelles et de l'image associées

- Technique : Deep Learning (Natural Language Processing ,Computer Vision)





2. Présentation des données

- 27 variables cibles (nombre de catégories de produits)
- Aucun doublon
- Codes *productid & imageid* unique par article
- *Champ description contient* 35,09% de NAN
- Champ de désignation : moyenne 11 mots par observation (50 Mo)
- Images: couleur, 500x500 pixels, encodées au format JPG (2,5 Go

Données textes **Id fichiers images** **Label**

	designation	description	productid	imageid	prdtypecode
0	Olivia: Personalisiertes Notizbuch / 150 Seite...	NaN	3804725264	1263597046	10
1	Journal Des Arts (Le) N° 133 Du 28/09/2001 - L...	NaN	436067568	1008141237	2280
2	Grand Stylet Ergonomique Bleu Gamepad Nintendo...	PILOT STYLE Touch Pen de marque Speedlink est ...	201115110	938777978	50

84916 lignes

3. Pré traitement

a. Données textes

Exemple avec la désignation : « <p>Ce robot de piscine d&#39un design innovant! »

Etape	Résultat
Encodage html	<p>Ce robot de piscine d’un design innovant!
Suppression balise	Ce robot de piscine d’un design innovant!
Suppression ponctuation	Ce robot de piscine d’un design innovant
Minuscule	ce robot de piscine d’un design innovant
Stemming et suppression stopwords	robot piscin design innov
Tokenizer	[186 , 4, 199 ,4488]
Padding	[186 , 4, 199 ,4488,0,0,0,0,0,0,0,0,0,0,0,0,0]

3. Pré traitement

b. Données images

- Dispersion des images (même intra-classe)

- ImageDataGenerator :
  - Streaming par batches
  - Data augmentation
  - Redimensionnement (en 256x256 pixels)
  - Preprocessing

4. Modèles et performances

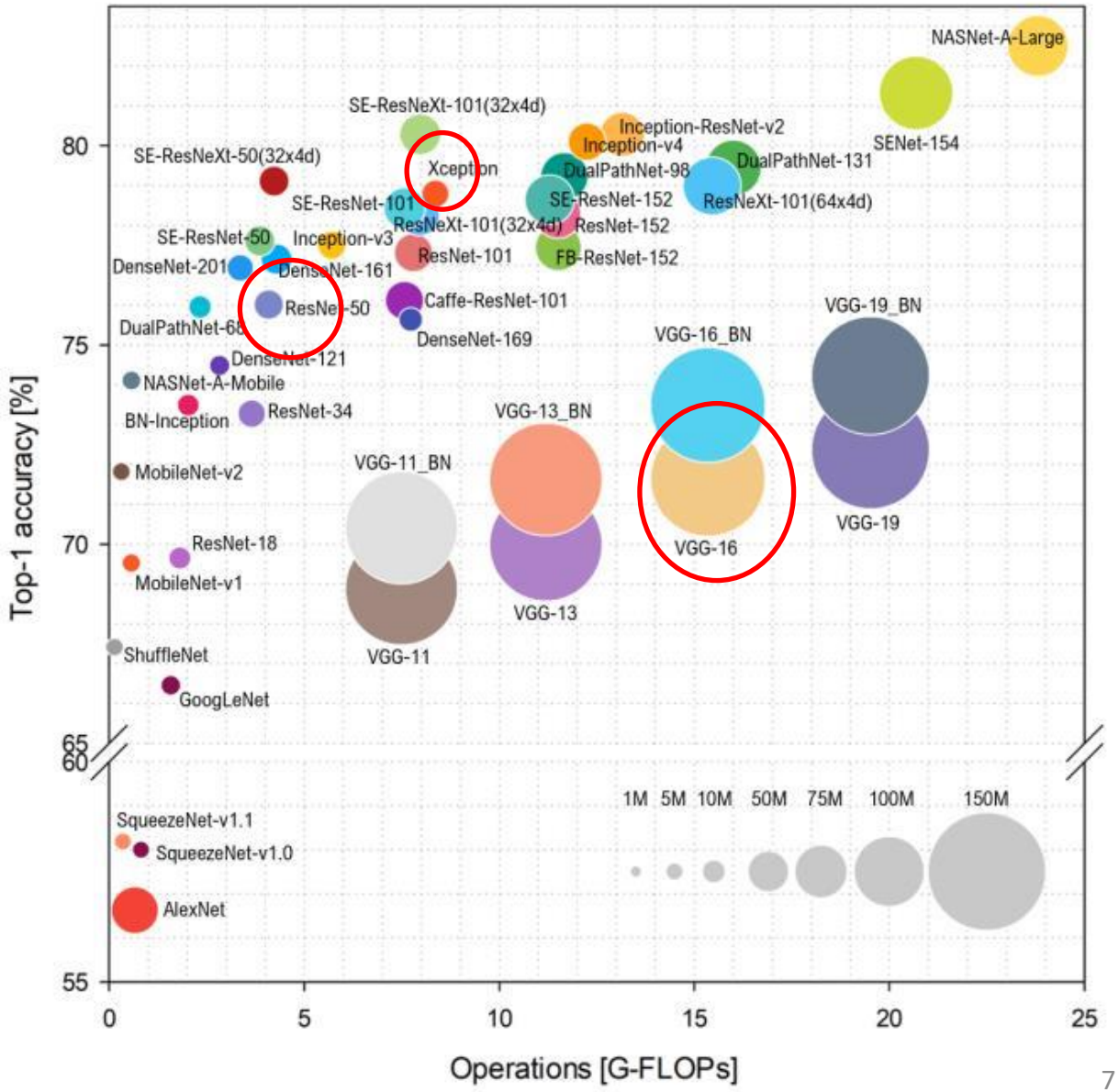
a. Images

<u>Choix du CNN pré-entraîné :</u>

1. Notoriété
2. Complexité
3. Performance

<u>Modèles retenus :</u>

- ResNet50 (2015)
- Xception (2016)
- VGG16 (2014)



4. Modèles et performances

a. Images

Choix des hyperparamètres :

- Couches convolutives gelées
- Taux de Dropout
- Fonction de perte
- Optimiseur
- Batch size et learning rate (LR)

4. Modèles et performances

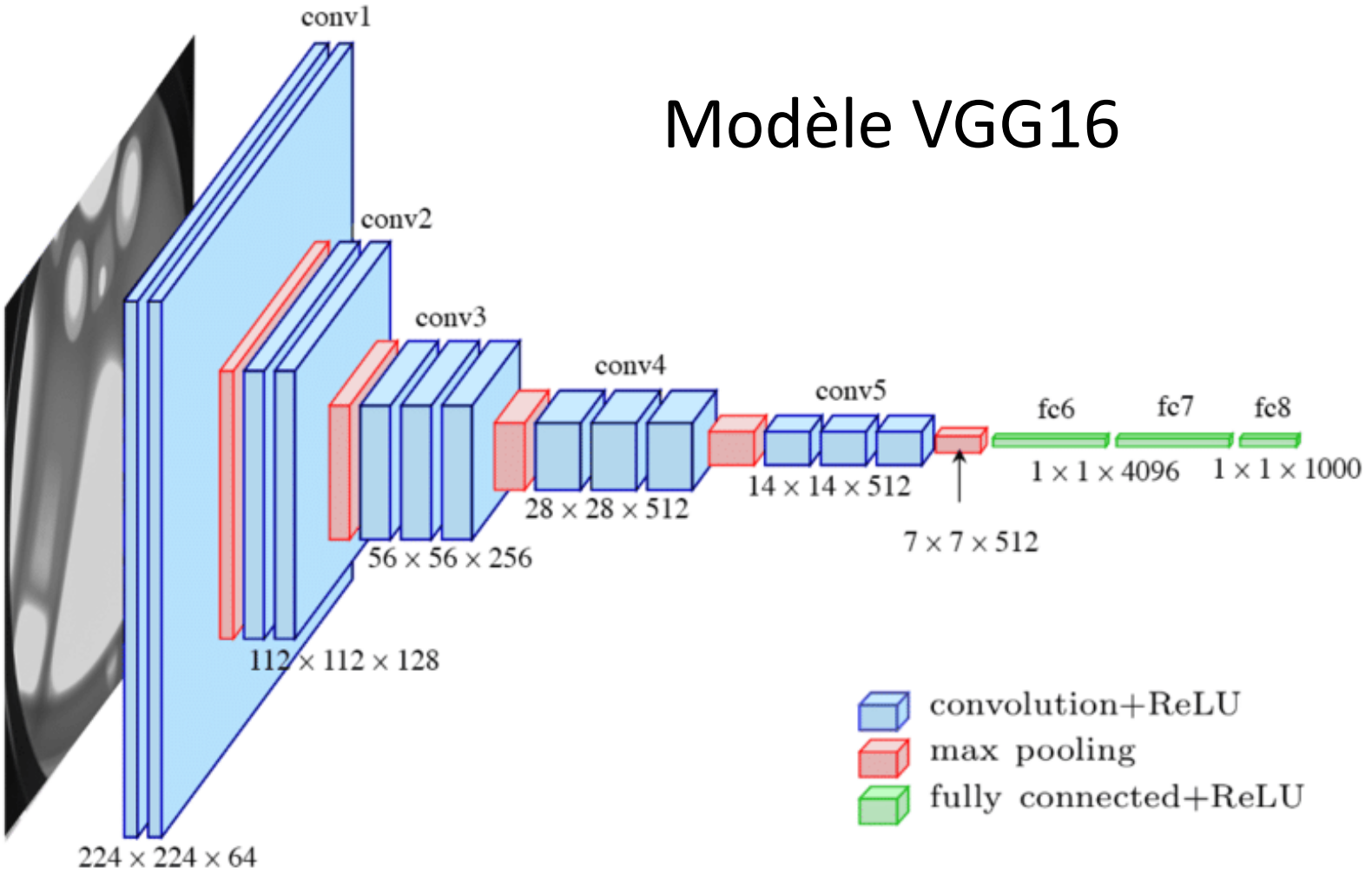
a. Images

Choix du backbone

CNN backbone	Hidden layers	Batch size	LR	Batch norm.	Drop. rate	Epochs	Valid. acc.	Valid. F1 macro	Valid. F1 weighted
Resnet50	[512, 256]	32	0.001	No	0	50	45.00%	42.00%	46.00%
Resnet50	[512, 256]	32	0.010	Yes	0.2	15	54.00%	50.00%	55.00%
VGG16	[256, 128]	32	0.001	Yes	0.2	20	49.00%	45.00%	50.00%
VGG16	[1024, 512]	32	0.001	Yes	0.2	15	52.00%	48.00%	53.00%
Xception	[2048, 1024]	128	0.001	Yes	0.375	20	53%	49.00%	53.00%

4. Modèles et performances

a. Images



4. Modèles et performances

a. Images

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, None, None, 512)   14714688
_________________________________________________________________
global_average_pooling2d (Gl (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 1024)              525312
_________________________________________________________________
batch_normalization (BatchNo (None, 1024)              4096
_________________________________________________________________
activation (Activation)      (None, 1024)              0
_________________________________________________________________
dropout (Dropout)            (None, 1024)              0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               524800
_________________________________________________________________
batch_normalization_1 (Batch (None, 512)               2048
_________________________________________________________________
activation_1 (Activation)    (None, 512)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 27)                13851
_________________________________________________________________
activation_2 (Activation)    (None, 27)                0
=================================================================
Total params: 15,784,795
Trainable params: 1,067,035
Non-trainable params: 14,717,760
```

4. Modèles et performances

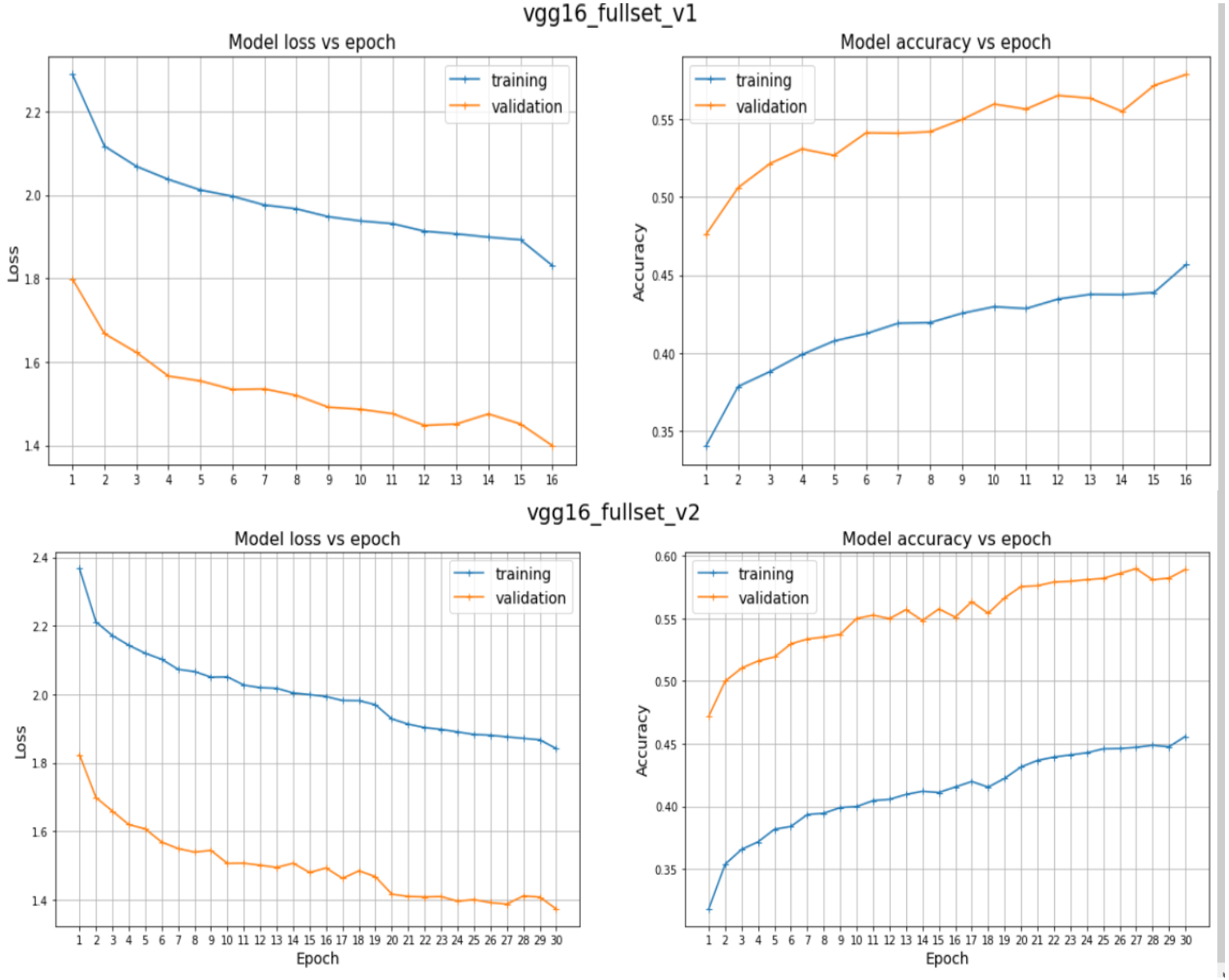
a. Images

<u>Premier essai:</u>

Batch = 64
LR = 0,01

<u>Deuxième essai:</u>

Batch = 32
LR = 0,01

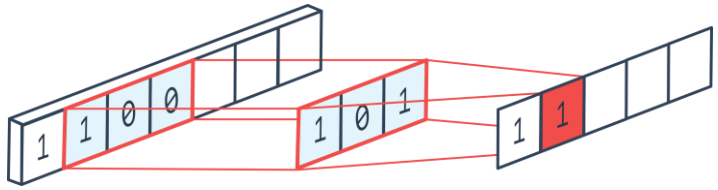


4. Modèles et performances

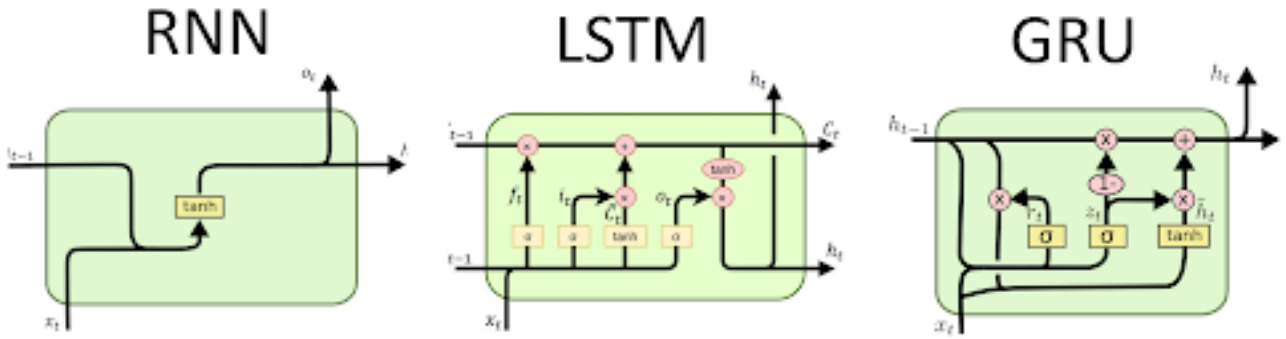
b. Textes

Modèles testés:

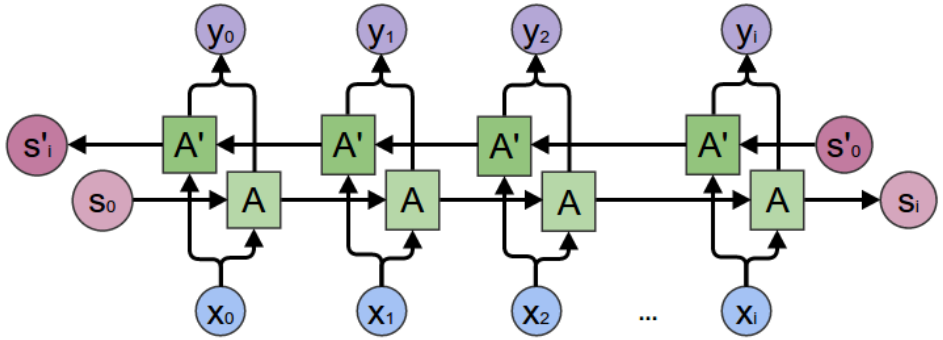
Convolution 1 dimension



RNNs unidirectionnels



RNNs Bidirectionnels



4. Modèles et performances

b. Textes

Choix des hyperparamètres :

- Batch size : 64, 128
- L'optimizer : “SDG”, “Adam” et “Nadam”
- Learning Rate : learning rate constant 10^-2
- Nombre d’epochs : entre 20 et 30
- Loss macro F1 customisée

4. Modèles et performances

b. Textes

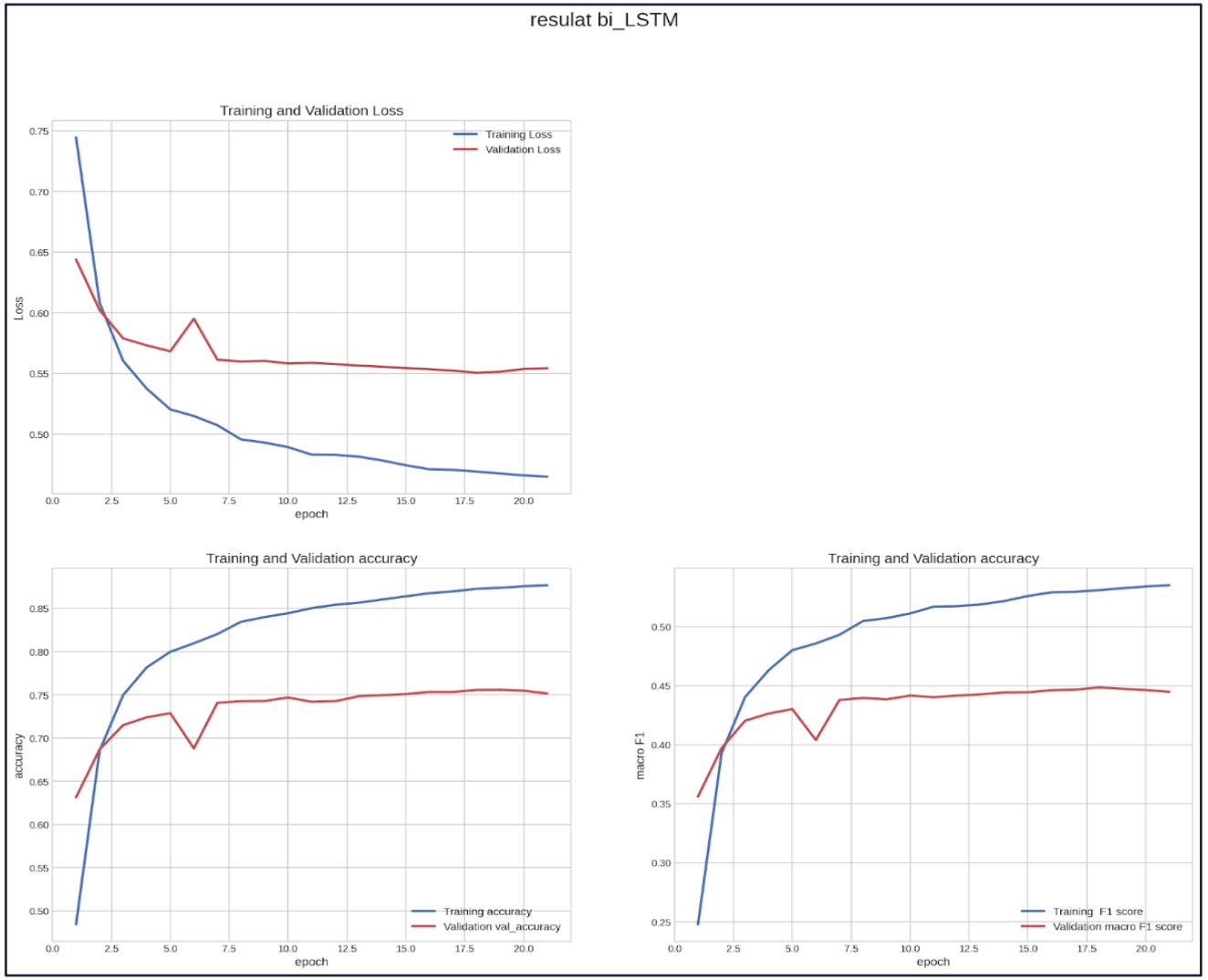
Résultats des quatre meilleurs modèles :

Models	macro_f1 train	weighted_f1 train	accuracy train	macro_f1 val	weighted_f1 val	accuracy val	macro_f1 test	weighted_f1 test	accuracy test
lstm	75,2%	83,1%	82,9%	64,2%	72,4%	71,8%	64,5%	72,8%	72,0%
bi_lstm 200 units	97,7%	97,8%	97,7%	75,8%	77,3%	77,3%	76,5%	77,7%	77,6%
bi_lstm 256 units	86,0%	88,5%	88,0%	72,5%	76,2%	75,6%	72,2%	75,8%	75,4%
gru	81,3%	84,4%	83,4%	69,4%	73,3%	72,0%	69,1%	73,2%	71,8%

4. Modèles et performances

b. Textes

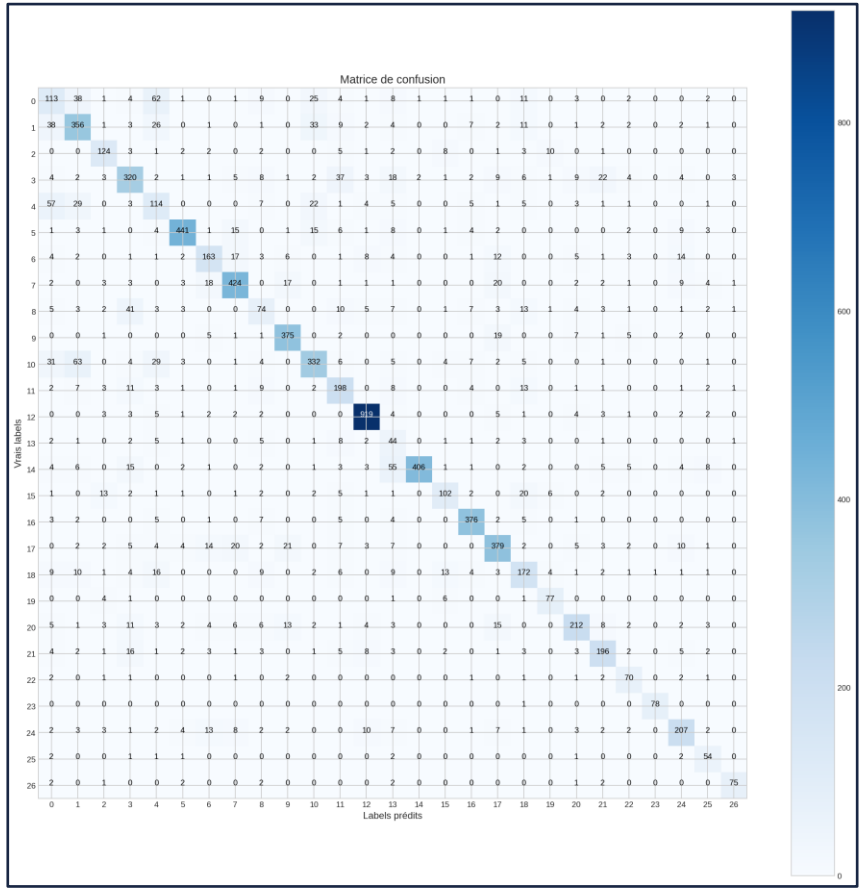
Performances du training :

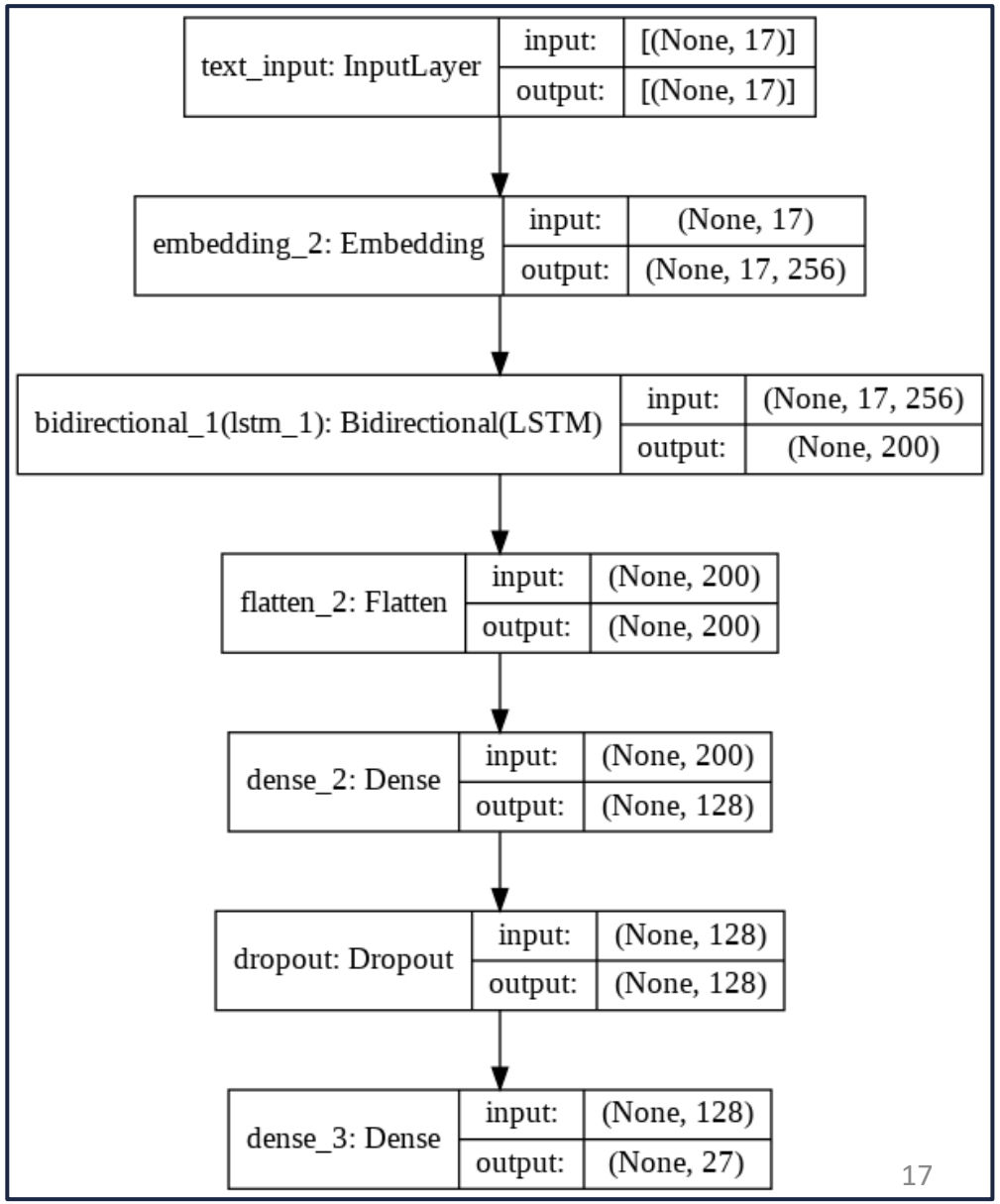


4. Modèles et performances

b. Textes

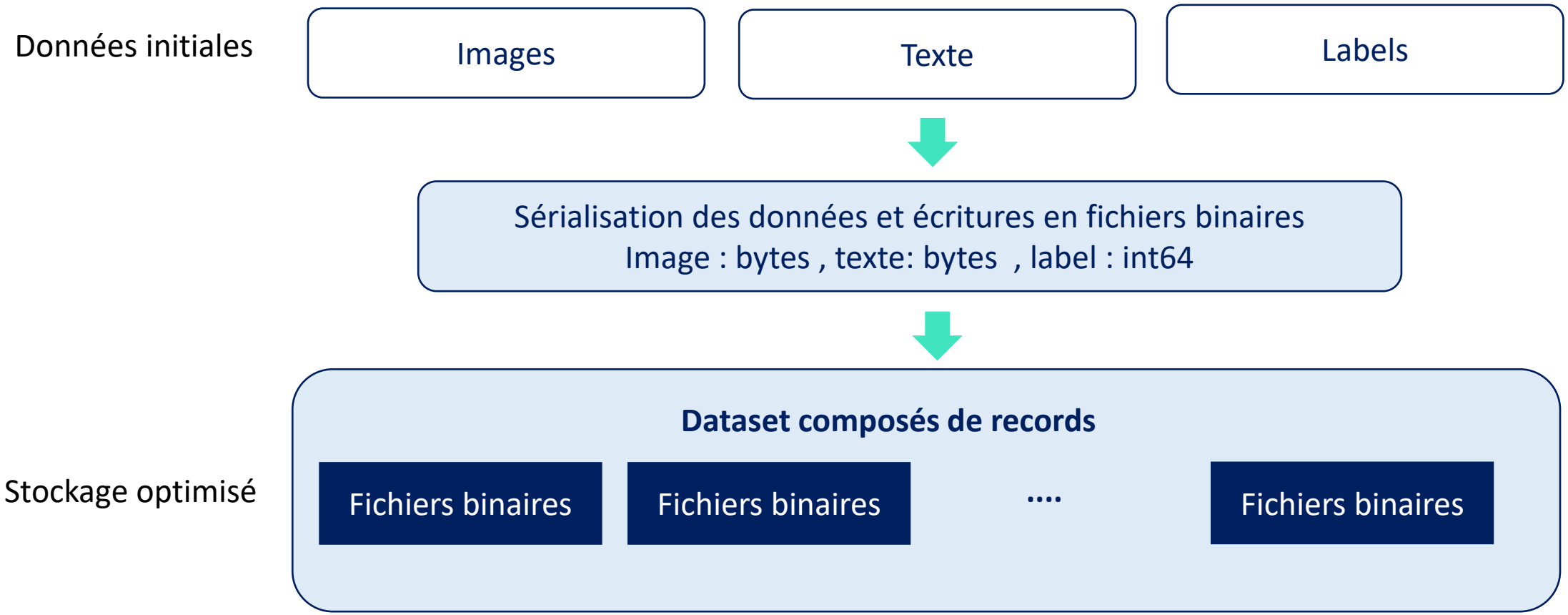
Architecture du meilleur modèle obtenu (bi_lstm) :





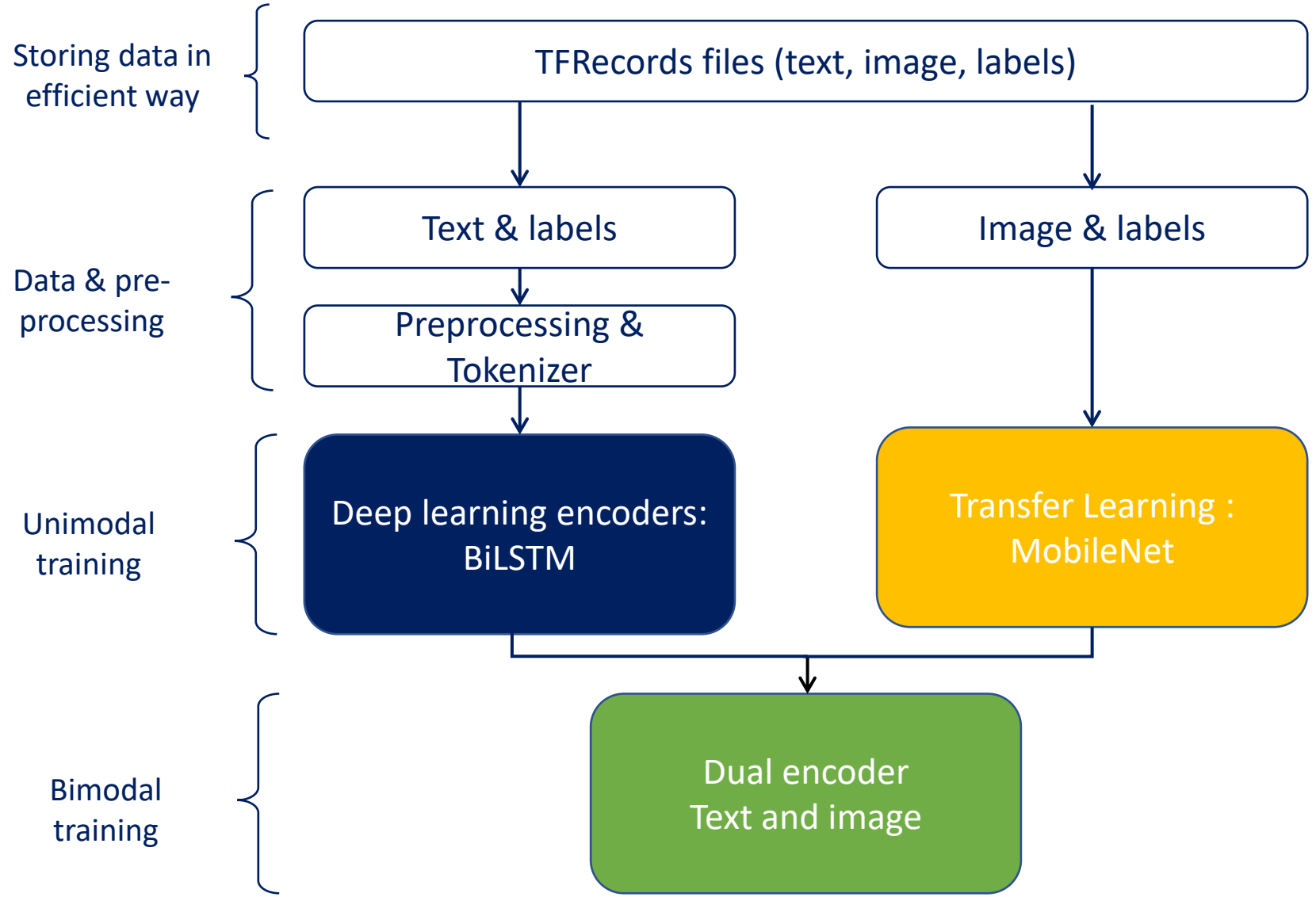
4. Modèles et performances

c. Bimodales



4. Modèles et performances

c. Bimodales

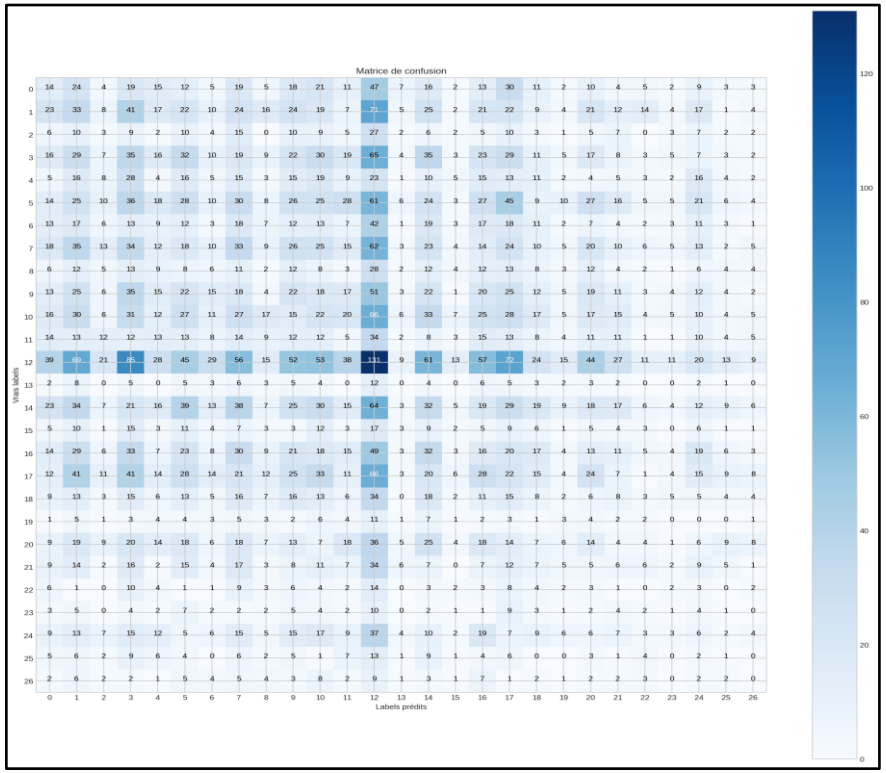


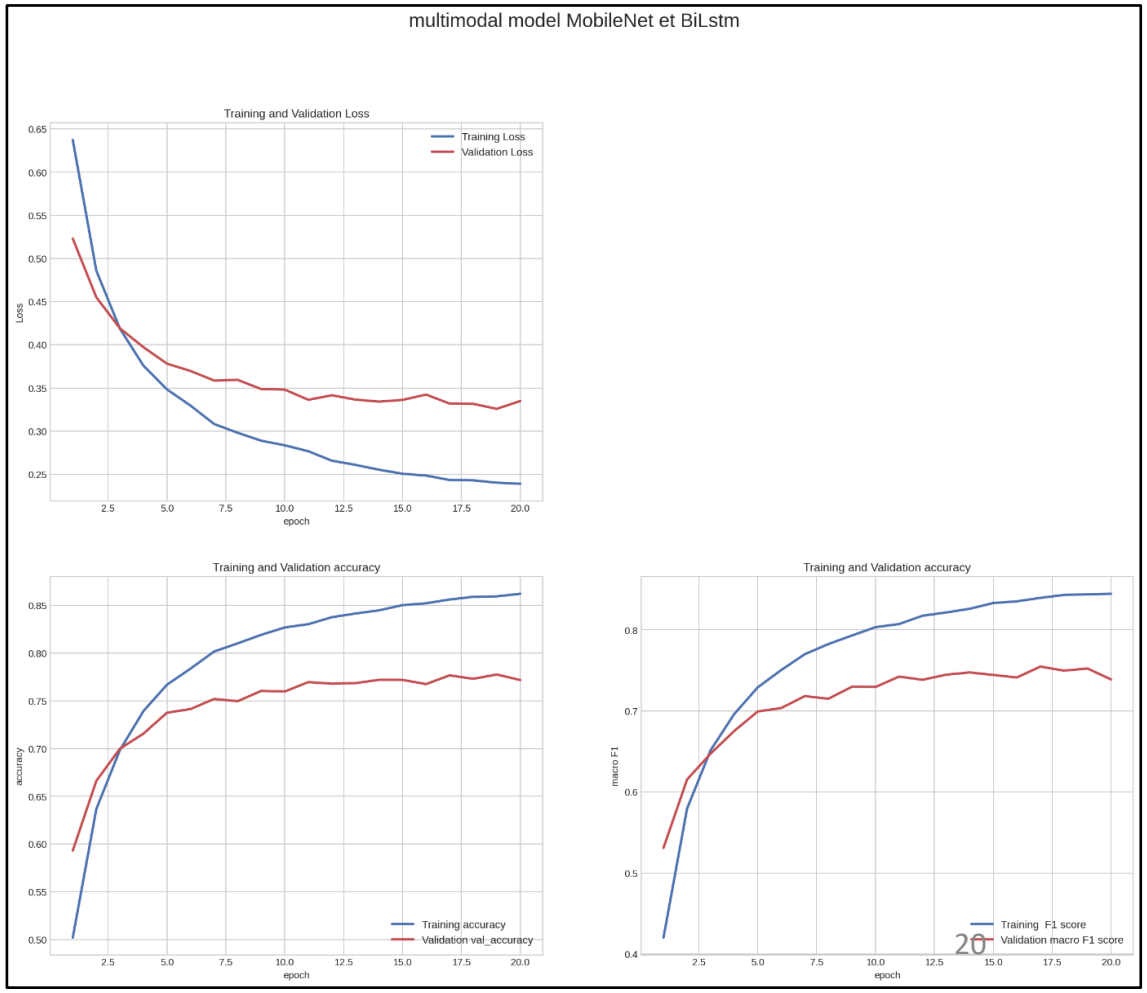
4. Modèles et performances

c. Bimodales

Résultats :

```
Training Accuracy: 0.8690 _ Training F1 macro : 0.8495
Validation Accuracy: 0.7718 _ Validation F1 macro : 0.7389
Testing Accuracy:  0.7687 _ Testing F1 macro : 0.7378
```





4. Modèles et performances

c. Bimodales

Pistes d'amélioration :

- Données rééquilibrées
- Représentation du texte : Glove
- Transfert Text learning
- Plus d'épochs.
- Autres optimizers

5. Conclusion projet

- *Acquis de connaissances :*

DataViz / NLP / CV / Deep Learning (RNN et CNN)

- *Environnement de travail:*

Anaconda : Jupyter et Spyder (streamlit)

Versioning GitHub

Google Colab

- *Métier Data Science:*

Concevoir, implémenter et évaluer des prototypes de modèle de classification

- Performances satisfaisantes et pistes d'amélioration nombreuses

Démo Streamlit