Classifiez automatiquement des biens de consommation

Projet n°6 - Parcours Data Scientist

Jérémy Vangansberg - 12/08/21

Introduction

- L’entreprise **Place de marché** est une marketplace e-commerce
- Fonctionnement actuel :
  - Les vendeurs proposent des articles en postant une photo et une description
  - **L’attribution de la catégorie de l’article est effectuée manuellement par les vendeurs**
- Objectif :
  - **Etudier la faisabilité d’un moteur de classification** qui utiliserait les images et la description pour classifier automatiquement les articles

Sommaire

1. Présentation des données
2. Modélisations : approches non-supervisées
    a. Texte
    b. Images
3. Modélisations : approches supervisées
    a. Texte
    b. Images
    c. Texte et images
4. Conclusion et pistes d’amélioration

Présentation des données

Le jeu de données

- 1050 enregistrements indexés par ‘uniq_id’
- 13 features
- Les variables sélectionnées pour les modélisations sont :
  - ‘**description**’ : texte décrivant le produit (572 mots maximum)
  - ‘**Image**’ : identifiant de l’image lié à l’enregistrement. Les images sont stockées dans le dossier ‘Flipkart/Images
  - ‘**product_category_tree**’ : labels
- Il n’y a aucune valeur manquante dans les variables utilisées

Les labels

- Les valeurs de la variable ‘**product_category_tree**’ sont enregistrées de la manière suivante : catégorie / sous-catégorie 1 / sous-catégorie 2 / sous-catégorie 3 / …
- Exemple :
  - ["Home Furnishing >> Curtains & Accessories >> Curtains >> Elegance Polyester Multicolor Abstract Eyelet Do… »
- J’ai retenu la **catégorie racine** en tant que label car cela permet de découper le dataset en **7 classes égales** (150 enregistrements par label)
- L’objectif de mes modèles sera de prédire la catégorie racine

Les labels

Nom des classes avec leurs traductions

Nom de classe	Traduction
Watches	Montres
Baby Care	Soins bébé
Computers	Ordinateurs
Kitchen & Dining	Cuisine et salle à manger
Home Furnishing	Ameublement
Home Decor & Festive Needs	Décoration d'intérieur et besoins festifs
Beauty and Personal Care	Beauté et soins personnels

Aperçu des variables utilisées

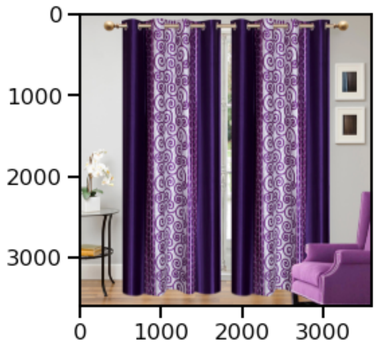
- ‘description’ : texte décrivant le produit

Key Features of Elegance Polyester Multicolor Abstract Eyelet Door Curtain Floral Curtain,Elegance Polyester Multicolor Abstract Eyelet Door Curtain (213 cm in Height, Pack of 2) Price: Rs. 899 This curtain enhances the look of the interiors.This curtain is made from 100% high quality polyester fabric.It features an eyelet style stitch with Metal Ring.It makes the room environment romantic and loving.This curtain is ant- wrinkle and anti shrinkage and have elegant apparance.Give your home a bright and modernistic appeal with these designs. The surreal attention is sure to steal hearts. These contemporary eyelet and valance curtains slide smoothly so when you draw them apart first thing in the morning to welcome the bright sun rays you want to wish good morning to the whole world and when you draw them close in the evening, you create the most special moments of joyous beauty given by the soothing prints. Bring home the elegant curtain that softly filters light in your room so that you get the right amount of sunlight.,Specifications of Elegance Polyester Multicolor Abstract Eyelet Door Curtain (213 cm in Height, Pack of 2) General Brand Elegance Designed For Door Type Eyelet Model Name Abstract Polyester Door Curtain Set Of 2 Model ID Duster25 Color Multicolor Dimensions Length 213 cm In the Box Number of Contents in Sales Package Pack of 2 Sales Package 2 Curtains Body & Design Material Polyester

---

Specifications of Sathiyas Cotton Bath Towel (3 Bath Towel, Red, Yellow, Blue) Bath Towel Features Machine Washable Yes Material Cotton Design Self Design General Brand Sathiyas Type Bath Towel GSM 500 Model Name Sathiyas cotton bath towel Ideal For Men, Women, Boys, Girls Model ID asvtwl322 Color Red, Yellow, Blue Size Mediam Dimensions Length 30 inch Width 60 inch In the Box Number of Contents in Sales Package 3 Sales Package 3 Bath Towel

- ‘image’ : images en couleur et de tailles variables

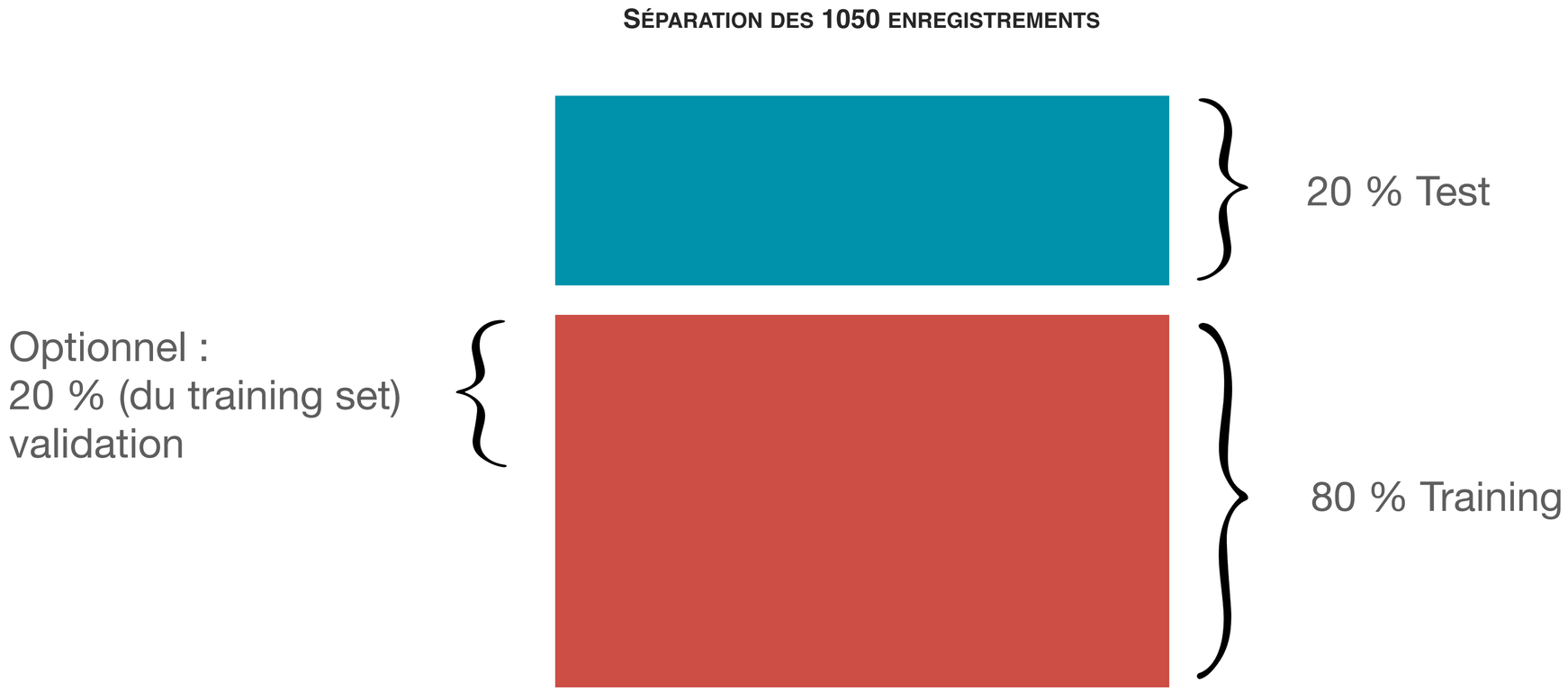


Démarche

- Dans un premier temps, je vais explorer des **méthodes non-supervisées** de modélisation. Ces méthodes sont **rapidement déployables** dans la mesure où elle ne nécessite pas de labels

- Ensuite je comparerai ces résultats avec des **modélisations supervisées** et notamment en effectuant du ***transfer learning.*** En effet, cette approche est pertinente dans ce cas

Séparation des données pour les approches supervisées



Modélisations : approches non-supervisées

Texte

NLP : preprocessing

- Modifications apportées au texte:
  1. Suppression de la casse
  2. Tokenization (Regex : [a-zA-Z]+)
  3. Stemming
  4. Création du corpus (chaque mot est relié à un index)
  5. Padding des documents

TF-IDF (1)

- Méthode de pondération qui permet d'évaluer l'importance d'un terme dans un document
- L'objectif de cette méthode est d'**accorder moins d'importance aux termes qui apparaissent souvent** dans les documents qu'aux termes apparaissant plus rarement
- Composantes :
  - TF ou Term Frequency
    - nombre d'occurrences du terme / nombre de termes dans le document
  - IDF ou Inverse Document Frequency
    - Log(nombre total de documents dans le corpus / nombre de documents où le terme apparaît)
- Formule finale : TF x IDF

TF-IDF (2)

- Cette algorithme produit une matrice de taille 1050x4215 (nombre de *training examples* par nombre de mots dans mon corpus)
- On peut extraire les mots ayant le score le plus élevé par document

CRÉATION DES VISUAL WORDS PAR CLUSTERING DES DESCRIPTEURS

Index	Top	Traduction	label
0	curtain	rideaux	Home Furnishing
1	bath	bain	Baby Care
2	towel	serviette	Baby Care
3	bedsheet	drap	Home Furnishing
4	sheet	drap	Home Furnishing

LDA : Latent Dirichlet Allocation

- C’est un modèle probabiliste génératif permettant, entre autre, de **découvrir des sujets abstraits dans une collection de documents**
- Cette algorithme dispose d’un **hyperparamètre *n_components***, qui fixe le nombre de sujets à définir. Dans ce cas, je vais fixer cette variable à 7 (nombre de labels différents)
- En effet, l’objectif est que les **7 sujets correspondent aux 7 labels**
- On peut explorer les sujets trouvés par cette modélisation non-supervisée en inspectant les mots les plus représentatifs de chaque sujet

LDA : Latent Dirichlet Allocation

TOP 10 DES MOTS PAR SUJET ET LABEL CORRESPONDANT

Index du sujet	Top 10 des mots	Label le plus proche*
0	home hair price showpiece rs cm art towel beautiful brass	Beauty and personal care
1	warranty usb adapter power laptop battery replacement light product quality	Computers
2	mug ceramic coffee perfect mugs gift material safe loved rockmantra	Kitchen and Dining
3	skin oil traits shampoo wall ml soap type used applied	Beauty and personal care
4	products free rs delivery genuine shipping cash buy 30 day	/
5	cm pack baby features color specifications general cotton package number	Baby care
6	laptop skin print shapes pad set mouse warranty combo multicolor	Computers

* Rapprochement sémantique selon mon interprétation

Modélisations : approches non-supervisées

Images

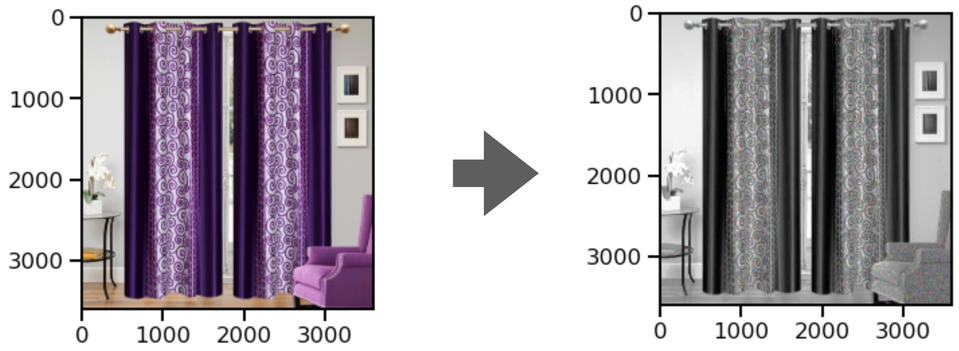
Preprocessing des images pour le clustering

- Etapes :

  1. Égalisation de l’histogramme

  2. Transformation en nuances de gris

IMAGE ORIGINALE ET APRÈS TRANSFORMATION



SIFT

- Le sift ou *scale-invariant feature transform* est un algorithme utilisé en computer vision qui **permet de détecter et d'identifier des éléments similaires entre différentes photos**

- Cette algorithme peut être utilisé pour regrouper des images semblables entre elles (clustering)

SIFT : descripteurs

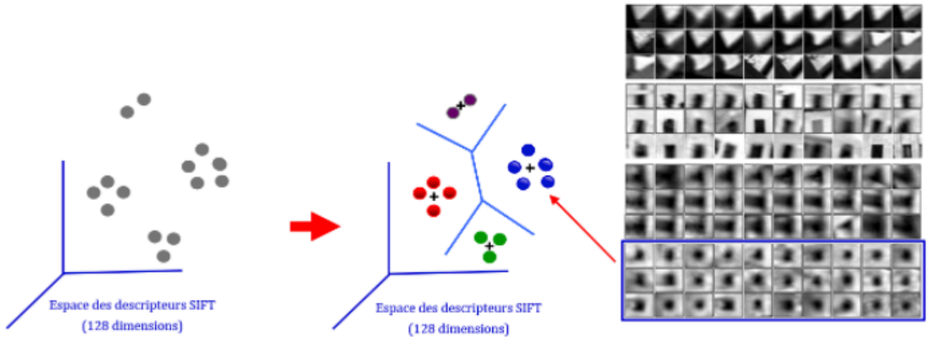
- **Chaque image contient des descripteurs** ou points remarquables (vecteur d'une longueur 128)

source : opencv.org

- Une image peut contenir plusieurs centaines de descripteurs
- En fonction de la longueur de notre dataset, on peut avoir plusieurs milliers de descripteurs

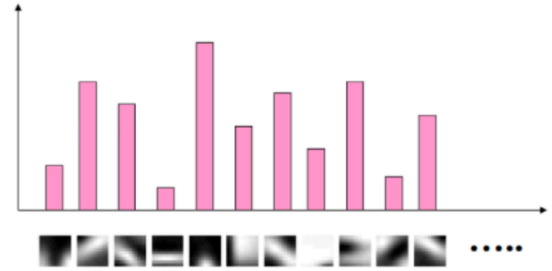
SIFT : features des images

CRÉATION DES VISUAL WORDS PAR CLUSTERING DES DESCRIPTEURS



- A l'aide du **k-means**, on va regrouper les descripteurs similaires entre eux afin de créer des **visuals bag of words**

UNE IMAGE ET SON HISTOGRAMME INDIQUANT LA FRÉQUENCE D'APPARITION DE CHAQUE VISUAL WORD



- Pour chaque image, on crée un **histogramme** qui indique la fréquence d'apparition de chaque visual bag of word (ou cluster) dans l'image

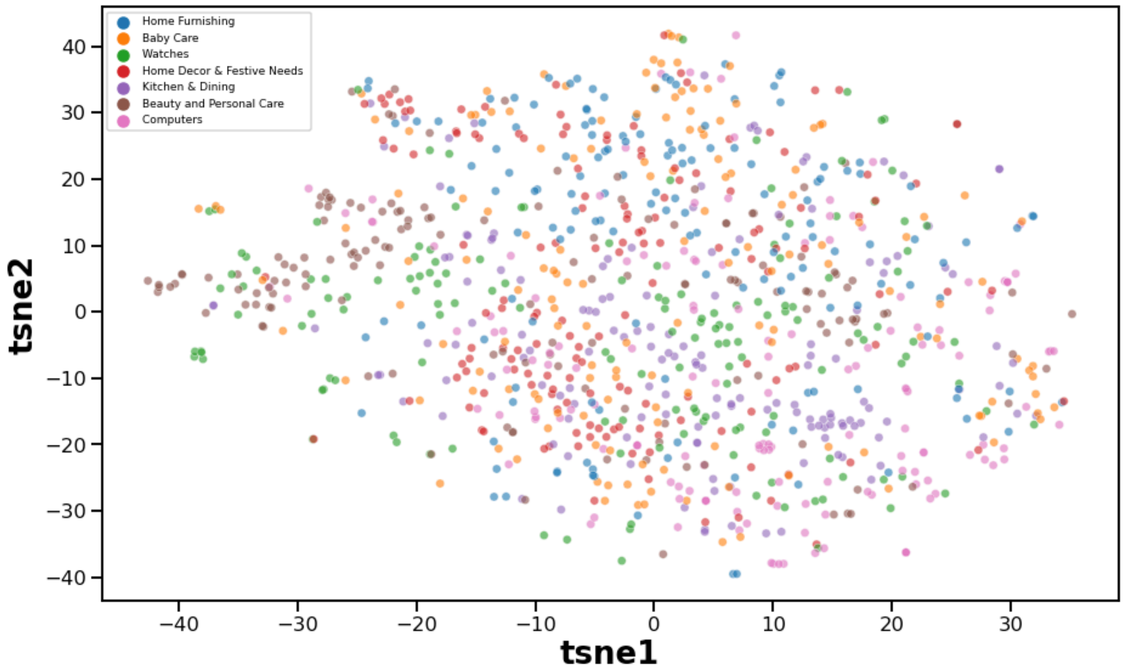
SIFT : optimisation

- Le nombre de clusters est un **hyperparamètre** du K-means
- Une des formules à notre disposition pour définir cette variable est de prendre la **racine carré du nombre total de descripteurs**
- Dans notre cas, *n_clusters* = 719. Je dispose donc d'un dataset de dimensions 1050 x 719 x 128.
- Il est possible de **réduire les dimensions du dataset** en perdant un minimum d'informations en effectuant une **PCA** avec *n_components* = 0.99. Ce qui revient à effectuer une PCA avec la contrainte de garder **99% de la variance expliquée**
- Suite à cette opération, la deuxième dimension du dataset passe de 719 à 505
- Cette étape permet d'accélérer le temps d'exécution du T-SNE

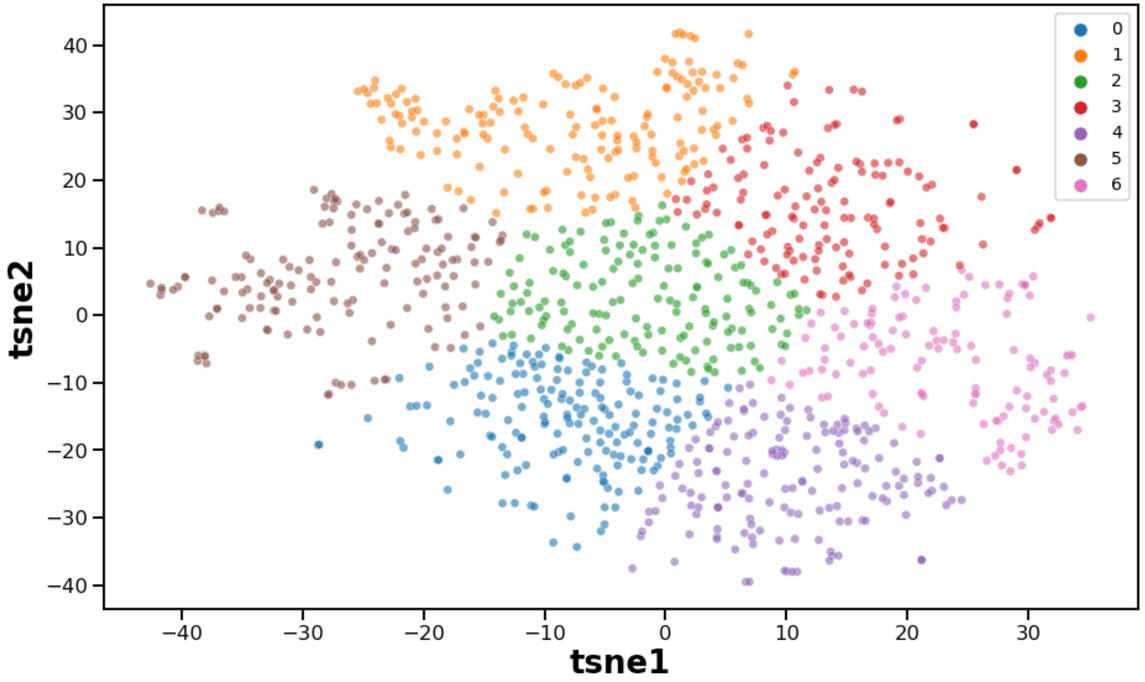
SIFT : T-SNE

- Le T-SNE est un algorithme qui permet de **réduire la dimension** d'un dataset à 2 ou 3 dimensions afin d'**effectuer des visualisations**

VISUALISATION DES DONNÉES SELON LES VRAIS CLASSES



VISUALISATION DES DONNÉES SELON LES CLUSTERS



- On peut utiliser le **Rand Index** (indice compris entre 0 et 1) pour évaluer la qualité d'un clustering, ici il est 0.06, ce qui caractérise un **clustering peu performant**

Les approches non-supervisées : conclusion

- Les différentes modélisations non-supervisées effectuées sont **peu performantes**
- Ces approches **ne pourraient pas être utilisées pour réaliser un moteur de classification automatique**

Modélisations : approches supervisées

Texte

Optimisation du temps d’entraînement

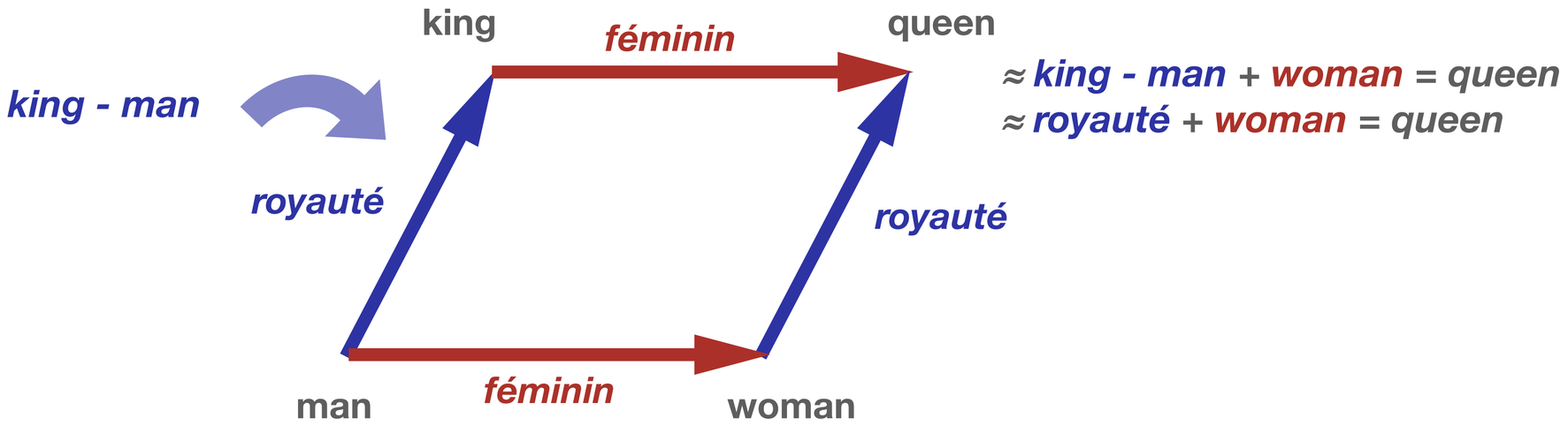
- La partie Deep learning a été réalisé avec keras/tensorflow 2.0
- Le kernel colab a une **accélération matériel TPU** qui permet **d’accélérer** le **temps d’exécution** des projets réalisés avec **keras/tensorflow 2.0**
- Résultats des tests effectués :
  - Durée d’une epoch sans accélération matérielle : 725 secondes
  - Durée d’une epoch avec TPU : 25 secondes
  - Soit un gain de facteur 29

Le transfer learning

- Cette méthode très utile en deep learning, permet de **tirer profit de modélisations complexes** qui ont été entraîné sur des jeux données très importants en ré-utilisant une partie du modèle pour une tâche annexe
- Cette technique est particulièrement **utile** quand notre **dataset est petit**
- Texte
  - Glove embeddings en dimensions 100
- Images
  - VGG16

Transfer learning de texte

WORD EMBEDDING : REPRÉSENTATION SÉMANTIQUE DES MOTS DANS L'ESPACE



≈ *king - man + woman = queen*
≈ *royauté + woman = queen*

≈ *woman + doctor = nurse*

- Glove, word2vec
- Méthode plus avancée : Transformers (BERT)

Régression logistique

- Transfer learning : **Glove embedding à 100 dimensions**
- Approche : chaque texte est représenté sous la forme d'un vecteur de dimension 100 qui est la somme des vecteurs mots qui le compose
- Le modèle va utiliser ces features pour effectuer des prédictions
- Taux de précision sur le jeu de données test : 0,88
- Malgré une méthode assez simpliste, cette approche est relativement efficace

Réseau de neurones

ARCHITECTURE DU RÉSEAU DE NEURONES

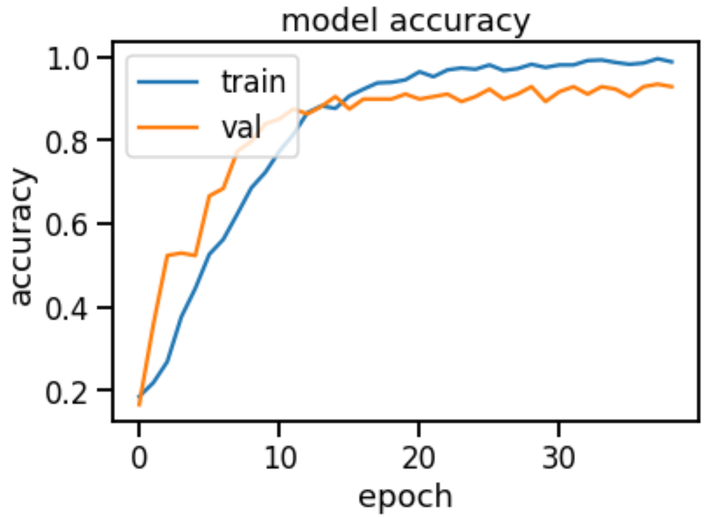
Layer (type)	Output Shape	Param #
input_74 (InputLayer)	[(None, None)]	0
embedding_10 (Embedding)	(None, None, 100)	573900
batch_normalization_14 (Batc	(None, None, 100)	400
conv1d_58 (Conv1D)	(None, None, 128)	64128
dropout_64 (Dropout)	(None, None, 128)	0
max_pooling1d_38 (MaxPooling	(None, None, 128)	0
batch_normalization_15 (Batc	(None, None, 128)	512
conv1d_59 (Conv1D)	(None, None, 128)	82048
max_pooling1d_39 (MaxPooling	(None, None, 128)	0
conv1d_60 (Conv1D)	(None, None, 128)	82048
global_max_pooling1d_19 (Glo	(None, 128)	0
dense_198 (Dense)	(None, 128)	16512
dropout_65 (Dropout)	(None, 128)	0
dense_199 (Dense)	(None, 7)	903

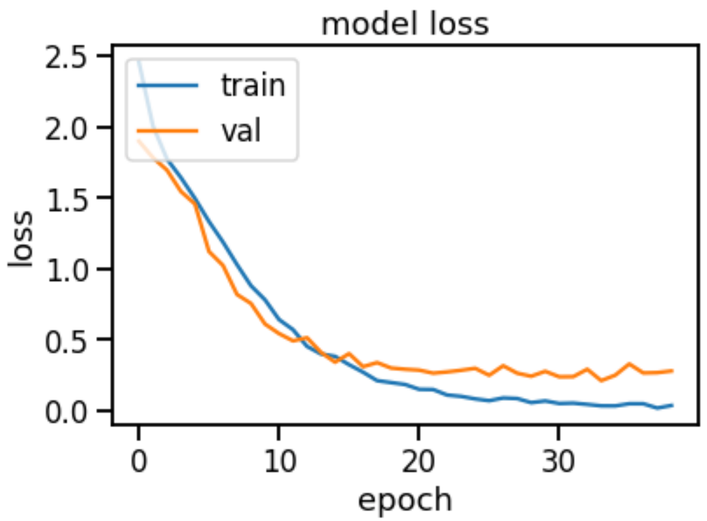
Total params: 820,451
Trainable params: 246,095
Non-trainable params: 574,356

- Paramètres :
  - optimiseur : adam
  - batch : 128
  - epochs : 300
  - early stopping : patience 3 sur la fonction coût
- Contrairement à l'approche précédente, chaque input à une dimension 572 x 100

Réseau de neurones

Graphique d'entraînement du modèle





- Les courbes d'apprentissage sont très satisfaisantes, on peut constater que le modèle apprend correctement sans overfitting

Accuracy sur les 3 différents splits

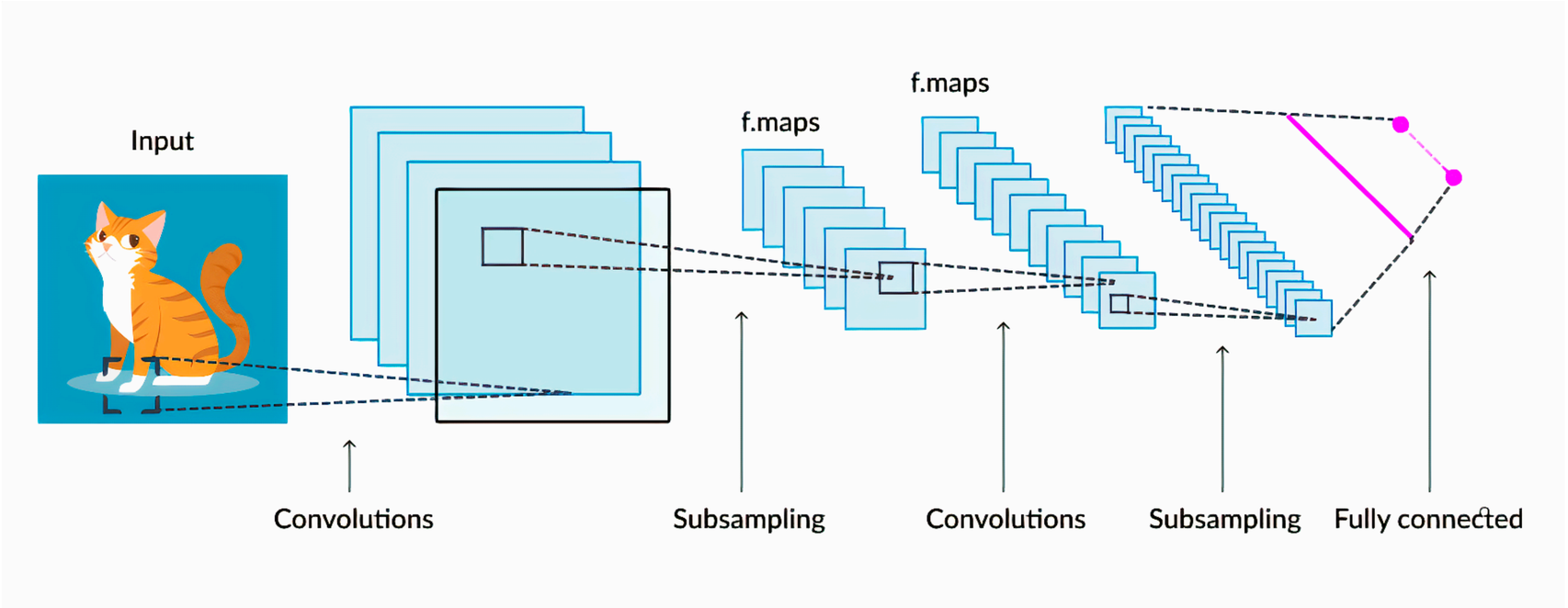
train	val	**test**
0,98	0,93	**0,85**

Modélisations : approches supervisées

Images

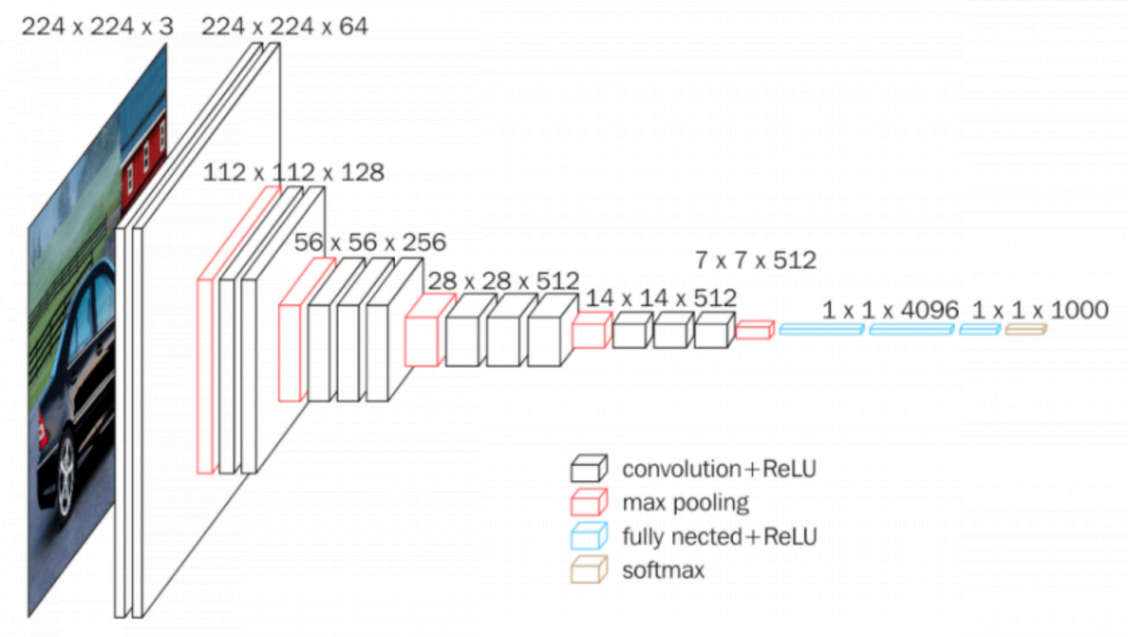
Convolutional Neural Network

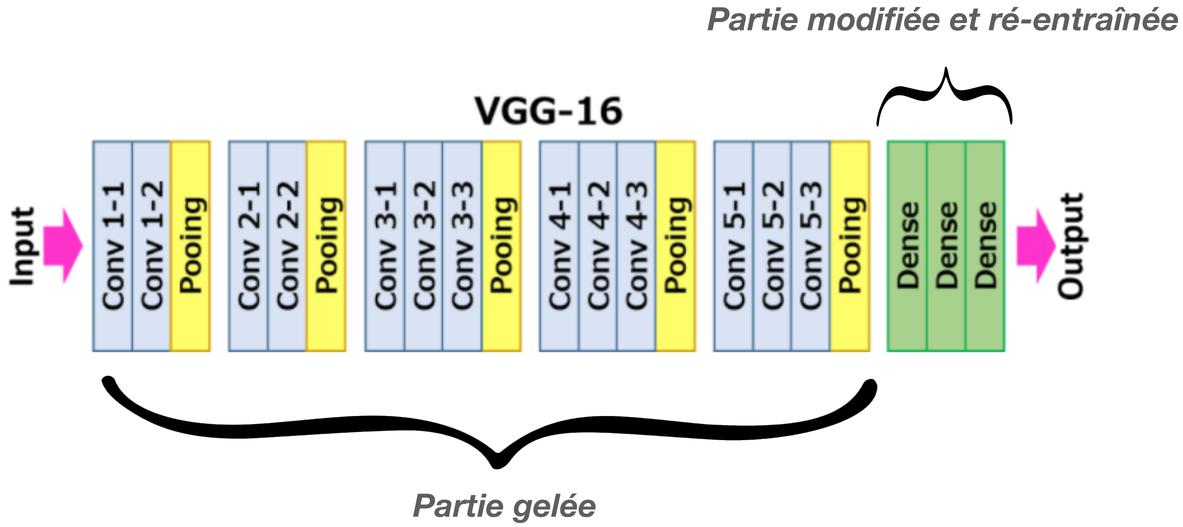
SCHÉMATISATION D'UN CONVNET



- ImageNet Large Scale Visual Recognition Challenge :
  - Compétition entre différents modèles sur une tâche de classification
  - Je vais utiliser les poids d'un modèle ayant atteint 92,7% à la compétition : VGG16

VGG16 : schématisations





Transfer learning : VGG16

ARCHITECTURE DU MODÈLE

_

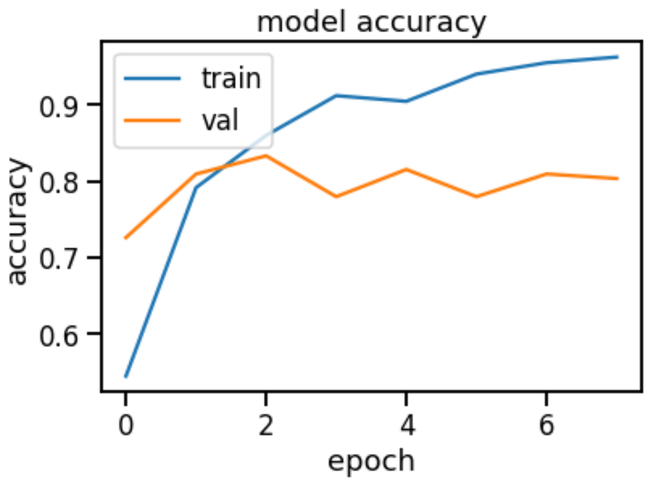
Layer (type)	Output Shape	Param #
input_75 (InputLayer)	[(None, 300, 300, 3)]	0
block1_conv1 (Conv2D)	(None, 300, 300, 64)	1792
block1_conv2 (Conv2D)	(None, 300, 300, 64)	36928
block1_pool (MaxPooling2D)	(None, 150, 150, 64)	0
block2_conv1 (Conv2D)	(None, 150, 150, 128)	73856
block2_conv2 (Conv2D)	(None, 150, 150, 128)	147584
block2_pool (MaxPooling2D)	(None, 75, 75, 128)	0
block3_conv1 (Conv2D)	(None, 75, 75, 256)	295168
block3_conv2 (Conv2D)	(None, 75, 75, 256)	590080
block3_conv3 (Conv2D)	(None, 75, 75, 256)	590080
block3_pool (MaxPooling2D)	(None, 37, 37, 256)	0
block4_conv1 (Conv2D)	(None, 37, 37, 512)	1180160
block4_conv2 (Conv2D)	(None, 37, 37, 512)	2359808
block4_conv3 (Conv2D)	(None, 37, 37, 512)	2359808
block4_pool (MaxPooling2D)	(None, 18, 18, 512)	0
block5_conv1 (Conv2D)	(None, 18, 18, 512)	2359808
block5_conv2 (Conv2D)	(None, 18, 18, 512)	2359808
block5_conv3 (Conv2D)	(None, 18, 18, 512)	2359808
block5_pool (MaxPooling2D)	(None, 9, 9, 512)	0
global_average_pooling2d_6 (	(None, 512)	0
dense_200 (Dense)	(None, 1024)	525312
dropout_66 (Dropout)	(None, 1024)	0
dense_201 (Dense)	(None, 1024)	1049600
dense_202 (Dense)	(None, 7)	7175

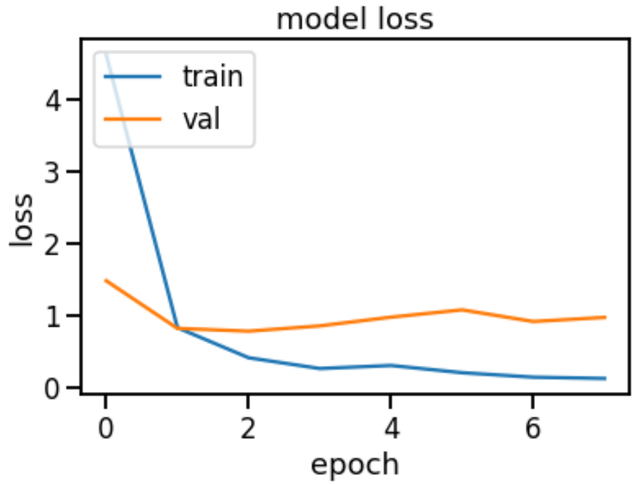
Total params: 16,296,775
Trainable params: 1,582,087
Non-trainable params: 14,714,688

- Preprocessing des images :
  - mode de couleurs : rgb
  - taille : 300 x 300
  - interpolation : bilinéaire

VGG16 : entraînement et performance

GRAPHIQUE D'ENTRAÎNEMENT DU MODÈLE





- Nombre d'Epochs : 50 (avec early stopping)
- Les performances sur le jeu de données d'entraînement s'améliorent significativement au fur et à mesure des epochs, tandis que les scores sur la progression est faible

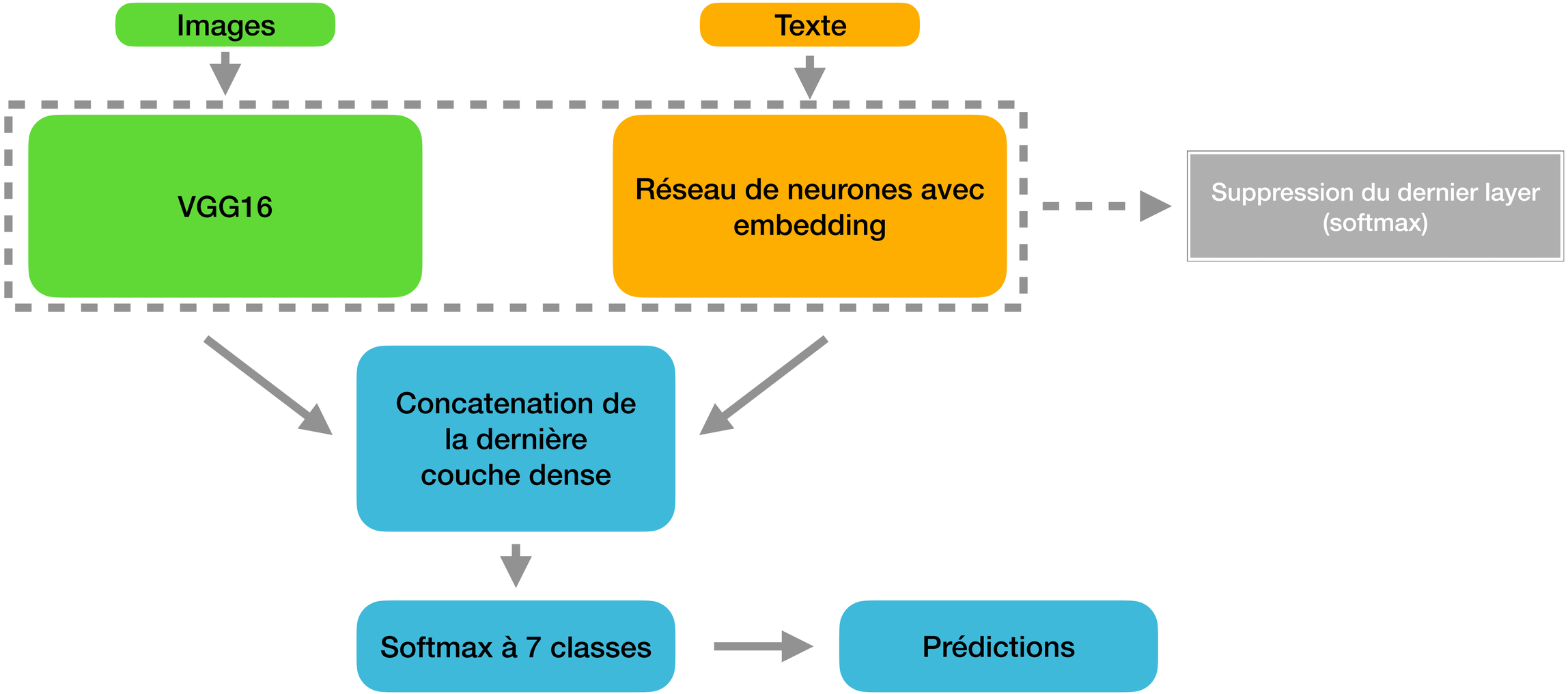
ACCURACY SUR LES 3 DIFFÉRENTS SPLITS

train	val	test
0,96	0,80	0,80

Modélisations : approches supervisées

Texte + Images

Schématisation de l’architecture du modèle multi-inputs



Transfer learning : multi-inputs

Architecture du modèle

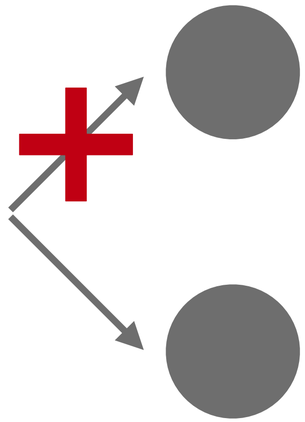
Layer (type)	Output Shape	Param #	Connected to
input_77 (InputLayer)	[(None, 300, 300, 3)	0	
block1_conv1 (Conv2D)	(None, 300, 300, 64)	1792	input_77[0][0]
block1_conv2 (Conv2D)	(None, 300, 300, 64)	36928	block1_conv1[0][0]
block1_pool (MaxPooling2D)	(None, 150, 150, 64)	0	block1_conv2[0][0]
block2_conv1 (Conv2D)	(None, 150, 150, 128	73856	block1_pool[0][0]
block2_conv2 (Conv2D)	(None, 150, 150, 128	147584	block2_conv1[0][0]
block2_pool (MaxPooling2D)	(None, 75, 75, 128)	0	block2_conv2[0][0]
block3_conv1 (Conv2D)	(None, 75, 75, 256)	295168	block2_pool[0][0]
block3_conv2 (Conv2D)	(None, 75, 75, 256)	590080	block3_conv1[0][0]
input_76 (InputLayer)	[(None, None)]	0	
block3_conv3 (Conv2D)	(None, 75, 75, 256)	590080	block3_conv2[0][0]
embedding_11 (Embedding)	(None, None, 100)	670600	input_76[0][0]
block3_pool (MaxPooling2D)	(None, 37, 37, 256)	0	block3_conv3[0][0]
batch_normalization_16 (BatchNo	(None, None, 100)	400	embedding_11[0][0]
block4_conv1 (Conv2D)	(None, 37, 37, 512)	1180160	block3_pool[0][0]
conv1d_61 (Conv1D)	(None, None, 128)	64128	batch_normalization_16[0][0]
block4_conv2 (Conv2D)	(None, 37, 37, 512)	2359808	block4_conv1[0][0]
max_pooling1d_40 (MaxPooling1D)	(None, None, 128)	0	conv1d_61[0][0]
block4_conv3 (Conv2D)	(None, 37, 37, 512)	2359808	block4_conv2[0][0]
conv1d_62 (Conv1D)	(None, None, 128)	82048	max_pooling1d_40[0][0]
block4_pool (MaxPooling2D)	(None, 18, 18, 512)	0	block4_conv3[0][0]
max_pooling1d_41 (MaxPooling1D)	(None, None, 128)	0	conv1d_62[0][0]
block5_conv1 (Conv2D)	(None, 18, 18, 512)	2359808	block4_pool[0][0]
conv1d_63 (Conv1D)	(None, None, 128)	82048	max_pooling1d_41[0][0]
block5_conv2 (Conv2D)	(None, 18, 18, 512)	2359808	block5_conv1[0][0]
global_max_pooling1d_20 (Global	(None, 128)	0	conv1d_63[0][0]
block5_conv3 (Conv2D)	(None, 18, 18, 512)	2359808	block5_conv2[0][0]
dense_203 (Dense)	(None, 128)	16512	global_max_pooling1d_20[0][0]
block5_pool (MaxPooling2D)	(None, 9, 9, 512)	0	block5_conv3[0][0]
dropout_67 (Dropout)	(None, 128)	0	dense_203[0][0]
flatten_50 (Flatten)	(None, 41472)	0	block5_pool[0][0]
dense_204 (Dense)	(None, 64)	8256	dropout_67[0][0]
dense_205 (Dense)	(None, 1024)	42468352	flatten_50[0][0]
concatenate_14 (Concatenate)	(None, 1088)	0	dense_204[0][0] dense_205[0][0]
dense_206 (Dense)	(None, 1024)	1115136	concatenate_14[0][0]
dropout_68 (Dropout)	(None, 1024)	0	dense_206[0][0]
dense_207 (Dense)	(None, 7)	7175	dropout_68[0][0]

Total params: 59,229,343
Trainable params: 43,843,855
Non-trainable params: 15,385,488

Enjeux et difficultés rencontrées

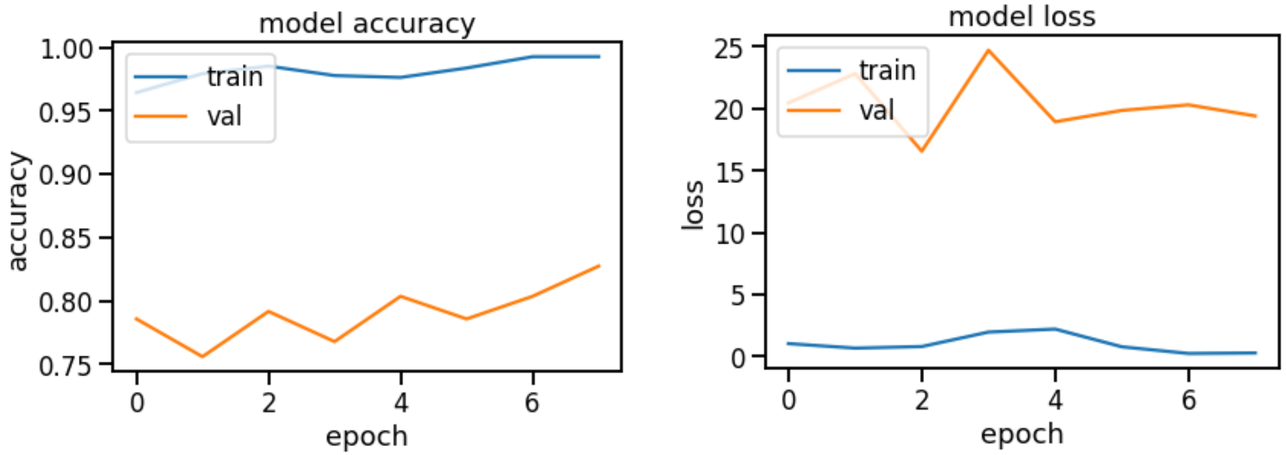
- Synchronisé les inputs (*random_state*)
- Le format des inputs (BatchDataset, np.array)
- Limiter l’overfitting (dropout)

SCHÉMATISATION DU DROPOUT



Multi-inputs : entraînement et performance

Graphique d'entraînement du modèle



- Le preprocessing du texte et des images est identique aux 2 architectures de réseaux de neurones précédentes
- Nombre d'Epochs : 8

Accuracy sur les 3 différents splits

train	val	test
0,99	0,82	**0,79**

Conclusion (1)

Modéles	Taux de précision (Train)	Taux de précision (Val)	Taux de précision (Test)
Régression logistique			**0,88**
Réseau de neurones (texte)	0,98	0,93	**0,85**
Réseau de neurones (images)	0,96	0,80	**0,80**
Réseau de neurones (multi-inputs)	0,99	0,82	**0,79**

Conclusion (2)

- Les approches **supervisées sont plus performantes** que les méthodes non-supervisées.
- Les performances de la régression logistique, des réseaux de neurones avec *images/ textes* et *images + textes sont assez similaires*
- Cependant, malgré le transfer learning, le jeu de données est petit et nous sommes confrontés à des **problèmes d'overfitting** quand les inputs ont des dimensions trop importantes (**fléau de la dimensionnalité**)
- Toutefois, il existe différentes stratégies pour réduire l'overfitting (régularisation, data augmentation, etc.)
- **Ces différentes modélisations démontrent la faisabilité d'un moteur de classification efficace**

Pistes d’amélioration : réduire l’overfitting

- Réaliser du data augmentation avec les images :
  - Flou, rotation, translation, miroir, luminosité
- Essayer un input texte semblable à celui de la régression logistique dans les réseaux de neurones (*document = somme des vecteurs mots*)