

Nostradamovies

Groupe 2

Yoann Loial - Victor Mader - Tristan Chaput

19 juin 2022



Introduction

La reconnaissance d'image, sous-catégorie de la Computer Vision et de l'Intelligence Artificielle, représente un ensemble de méthodes de détection et d'analyse d'images pour permettre l'automatisation d'une tâche spécifique.

Il s'agit d'une technologie qui est capable d'identifier des lieux, des personnes, des objets et plusieurs autres types d'éléments au sein d'une image et d'en tirer des conclusions en les analysant.

Dans le cadre de notre projet, nous avons dû réaliser un algorithme de prédiction de genre de film avec comme unique donnée l'affiche du film.

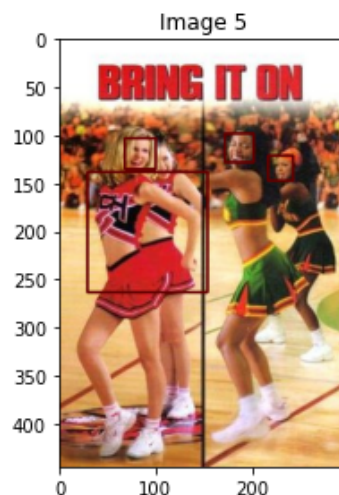
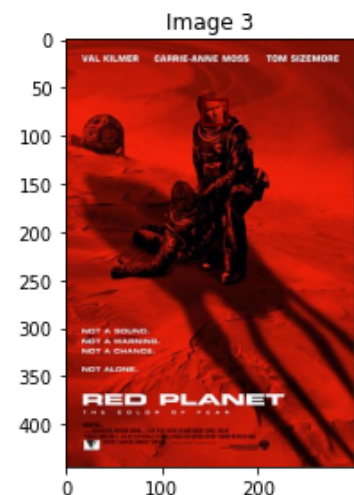
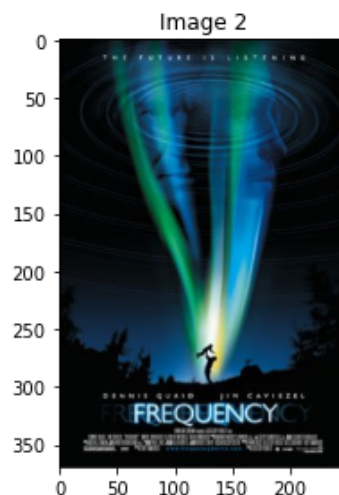
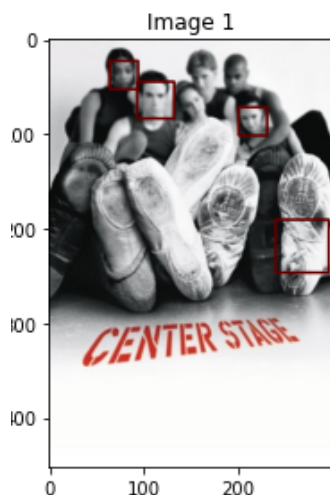
Nous avons utilisé différentes approches et techniques afin d'atteindre cet objectif notamment le *Deep Learning*, *Optical Character Recognition (OCR)* et *Natural Language Processing (NLP)*.

Reconnaissance des visages

Tout d'abord, nous sommes partis sur la librairie *opencv* pour la détection de visage qui utilise la technique de cascades de Haar.

Bien sûr, de nombreux algorithmes sont plus précis que les cascades de Haar mais ils sont toujours pertinents et utiles aujourd'hui.

L'un des principaux avantages des cascades de Haar est qu'elles sont si rapides qu'il est difficile de battre leur vitesse.



L'inconvénient des cascades de Haar est qu'elles ont tendance à être sujettes à des détections faussement positives, nécessitent un réglage des paramètres lorsqu'elles sont appliquées pour l'inférence/détection et, en général, ne sont pas aussi précises que les algorithmes plus "modernes" que nous avons aujourd'hui.

C'est pour cela que l'on a utilisé une autre méthode, MTCNN ou Multi-Task Cascaded Convolutional Neural Networks est un réseau de neurones qui détecte les visages et les repères faciaux sur les images. Il a été publié en 2016 par Zhang et al.

MTCNN est l'un des outils de détection de visage les plus populaires et les plus précis aujourd'hui. Il se compose de 3 réseaux de neurones connectés en cascade. Vous pouvez trouver un aperçu plus détaillé de MTCNN.

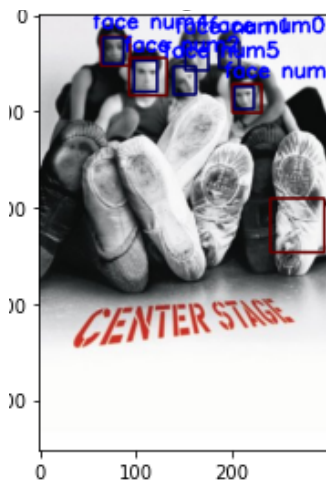


Image 4

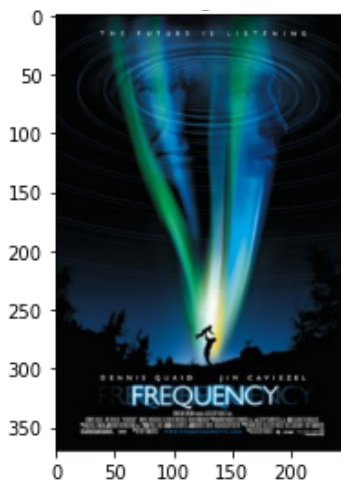


Image 5

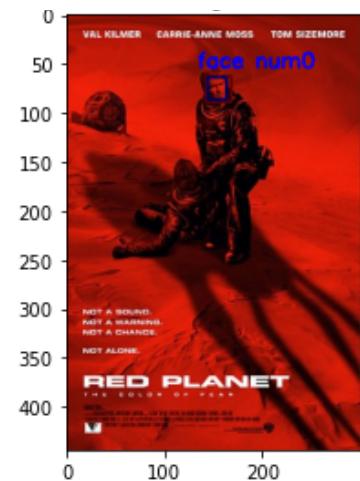
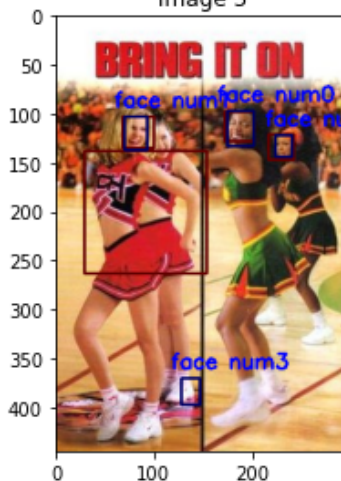



Image 6





Après plusieurs essais, la méthode du mtcnn nous paraît plus pertinente car on obtient de meilleurs résultats lors des tests.

Colorimétrie

vm:

Les espaces colorimétriques sont un moyen de représenter les canaux de couleur présents dans l'image qui donnent à l'image cette teinte particulière. Il existe plusieurs espaces de couleurs différentes et chacun a sa propre signification.

Certains des espaces colorimétriques populaires sont *RVB* (rouge, vert, bleu), *CMJN* (cyan, magenta, jaune, noir), *HSV* (teinte, saturation, valeur), etc.

Espace colorimétrique BGR : l'espace colorimétrique par défaut d'OpenCV est RVB. Cependant, il stocke en fait la couleur au format BGR. C'est un modèle de couleur additive où les différentes intensités de bleu, vert et rouge donnent différentes nuances de couleur.

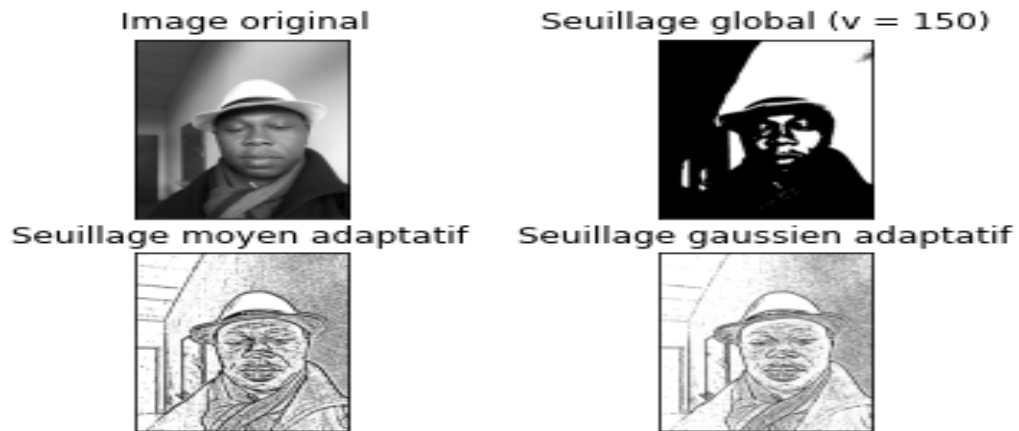
Espace colorimétrique HSV : Il stocke les informations de couleur dans une représentation cylindrique des points de couleur RVB. Il tente de représenter les couleurs telles qu'elles sont perçues par l'œil humain. La valeur de teinte varie de 0 à 179, la valeur de saturation varie de 0 à 255 et la valeur de valeur varie de 0 à 255. Il est principalement utilisé à des fins de segmentation des couleurs.

Espace colorimétrique CMJN : Contrairement au RVB, il s'agit d'un espace colorimétrique soustractif. Le modèle CMJN fonctionne en masquant partiellement ou entièrement les couleurs sur un fond plus clair, généralement blanc. L'encre réduit la lumière qui serait autrement réfléchi. Un tel modèle est dit soustractif car les encres « soustraient » les couleurs rouge, vert et bleu de la lumière blanche. Lumière blanche moins feuilles rouges cyan, lumière blanche moins feuilles vertes magenta et lumière blanche moins feuilles bleues jaunes.

Nous avons utilisé les bibliothèques opencv et numpy pour la détection de couleurs et des différentes intensités.

L'utilisation d'une valeur de seuil globale peut ne pas être un bon choix lorsque l'image a des conditions d'éclairage différentes dans différentes zones. Donc, dans ce cas, nous pouvons vouloir utiliser le seuillage adaptatif. Il utilise l'algorithme qui calcule le seuil pour de petites

régions de l'image afin d'obtenir différents seuils pour différentes régions de la même image et il nous donne de meilleurs résultats pour les images avec des conditions d'éclairage variables.

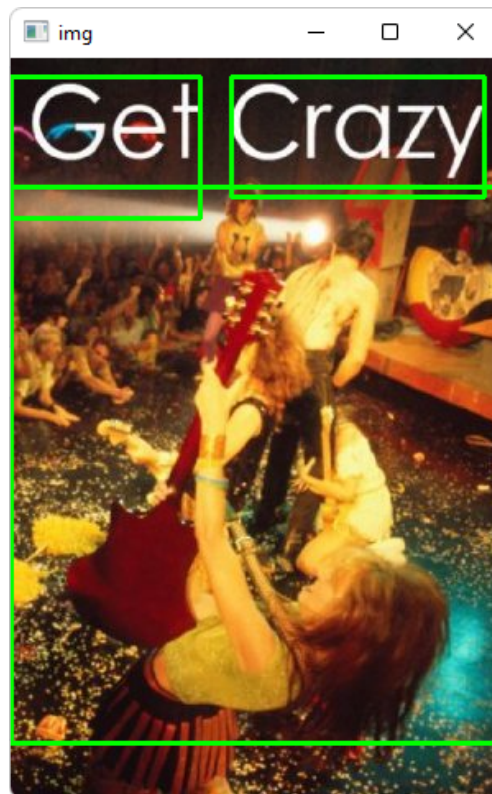


OCR

Pour l'extraction du texte présent dans les affiches, nous avons utilisé *pytesseract* et *opencv*, en utilisant ces 2 librairies nous pouvons dans un premier temps définir des rectangles de délimitation autour du texte avec *opencv* puis extraire le texte avec *tesseract*.

Étapes de l'OCR :

1. Transformation de l'image avec différents niveaux de gris.
2. Méthode d'Otsu pour effectuer un seuillage automatique qui sépare les deux classes pixels.
3. Dilatation de l'image, ce processus permet de supprimer le bruit blanc de l'image.
4. Création des rectangles autour du texte
5. Extraction du texte dans les boîtes précédemment créées.



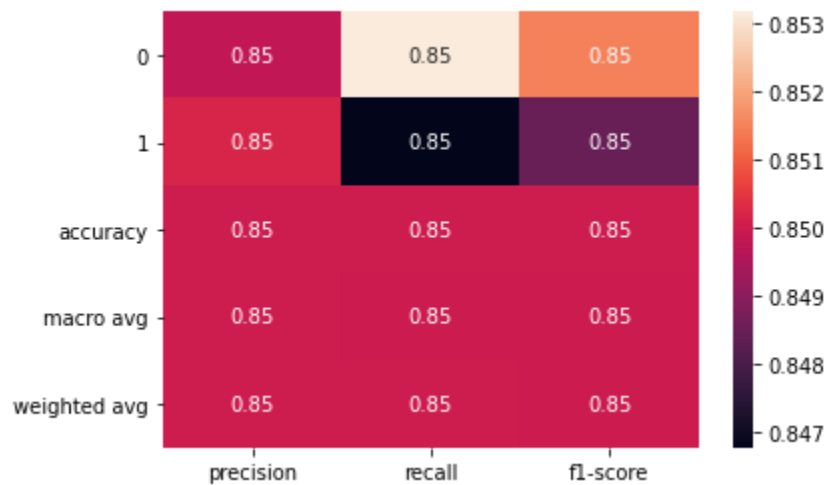
Benchmarking

Afin d'optimiser la vitesse de traitement et la précision pour la détection de langues nous avons effectué une comparaison entre les librairies *langid*, *fasttext* et *spaCy*.

Le dataset pour le benchmark est composé de tweets et il nous permet d'avoir les résultats suivants.

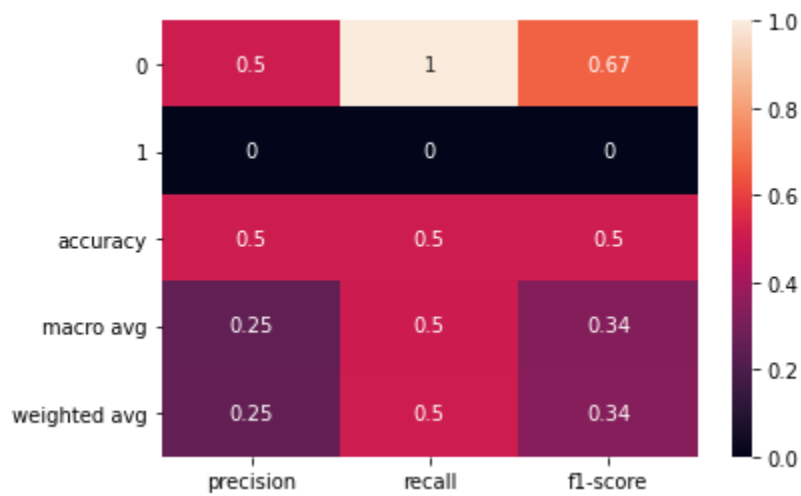
Pour *langid*, nous avons une précision de 85% et une vitesse de 934 ms

934 ms \pm 7.07 ms per loop (mean \pm std. dev. of 7 runs, 10 loops each)



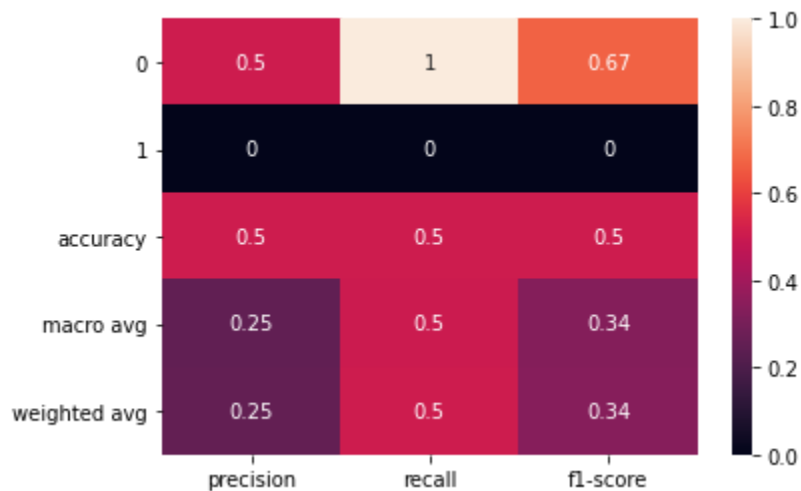
Pour *fasttext*, nous avons une précision de 50% et une vitesse de 7,4 ms

7.4 ms \pm 321 μ s per loop (mean \pm std. dev. of 7 runs, 10 loops each)



Pour spaCy, nous avons une précision de 50% et une vitesse de 7s

7 s \pm 97 ms per loop (mean \pm std. dev. of 7 runs, 10 loops each)



Conclusion

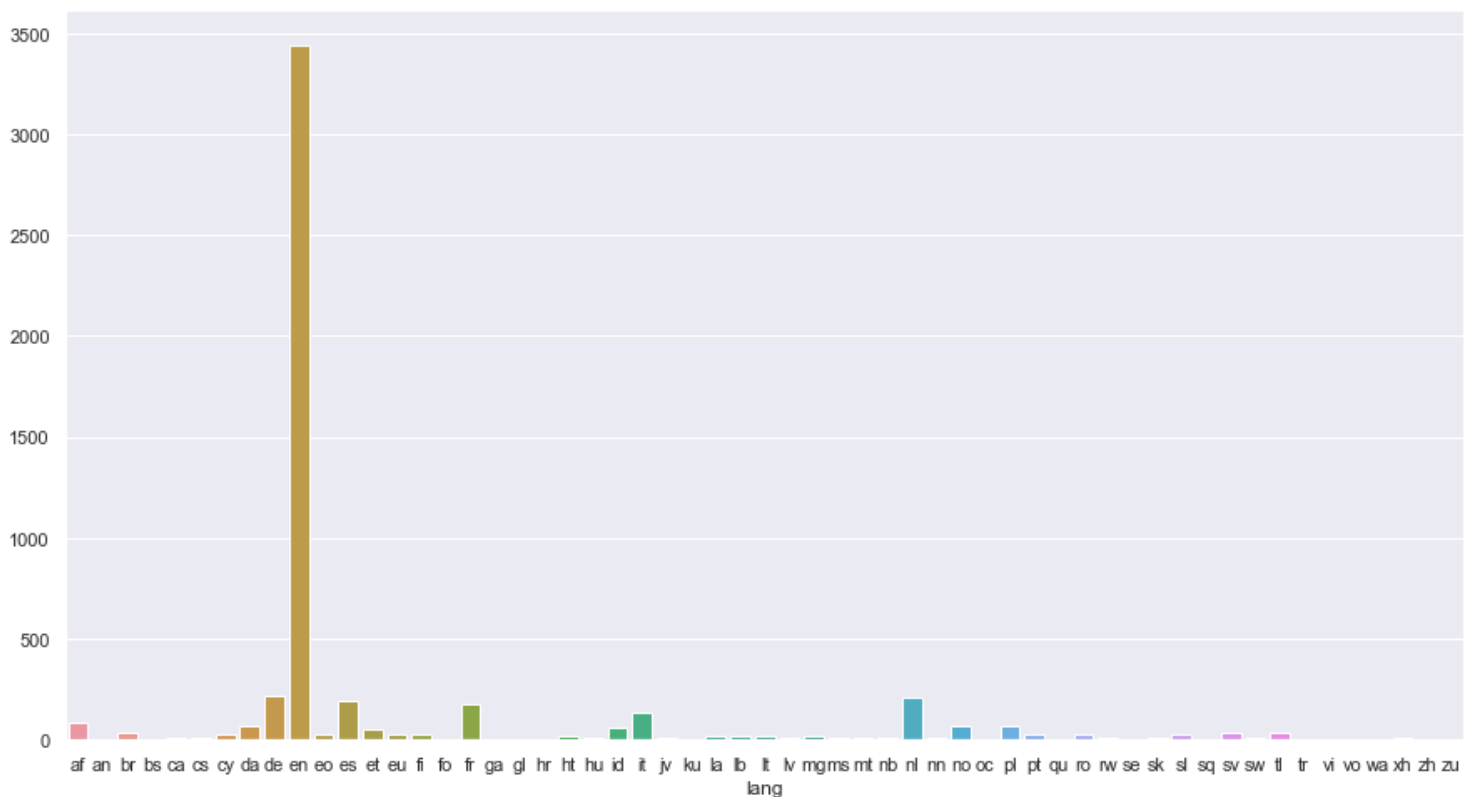
Nous pouvons observer une précision plus élevée pour *langid* et une plus grande rapidité de *fasttext*, *spaCy* est quant à lui hors compétition.

Nous avons donc fait un compromis en choisissant *langid* qui est plus précis que *fasttext* malgré sa rapidité.

Détection de langue

Précédemment, nous avons conclu que *langid* était la librairie la plus adaptée à notre problématique.

Ci-dessous, la répartition des langues en fonction du texte des affiches de films.

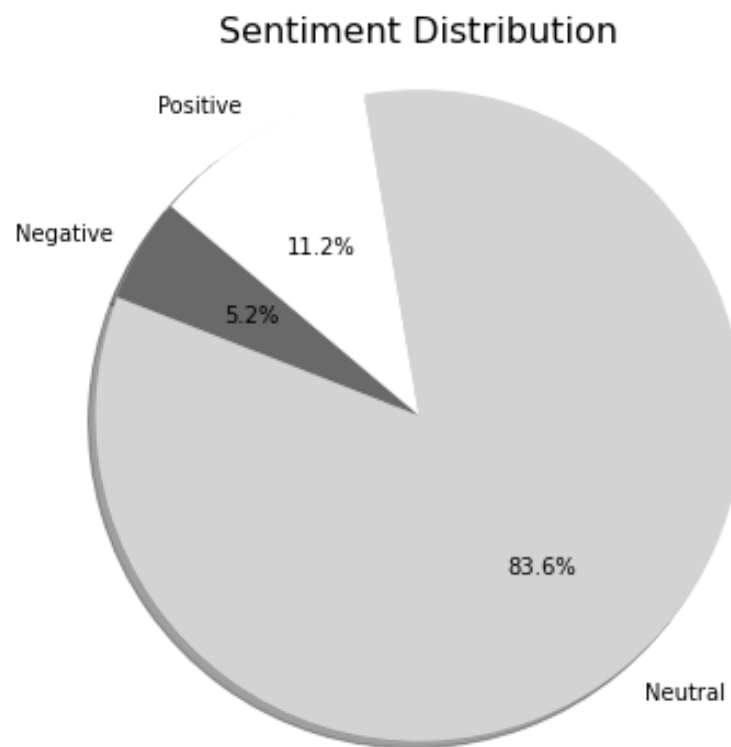


Sentiment distribution

Grâce à l'extraction de texte, nous avons pu explorer une feature intéressante, l'analyse de sentiments.

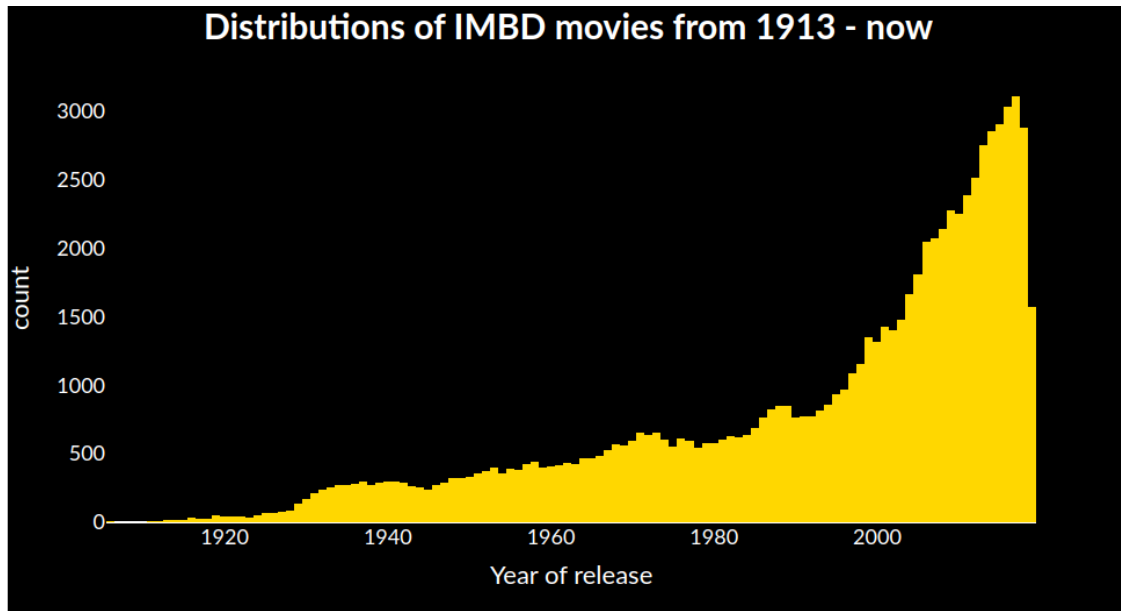
Nous effectuons une *lemmatisation* et nous retirons les *stopwords* afin que notre texte soit plus propre à analyser.

SpaCy permet de déterminer une polarité sur le texte. Si la polarité est positive, le sentiment est positif au contraire si la polarité est négative le sentiment est négatif, enfin si la polarité est nulle alors le sentiment est neutre.

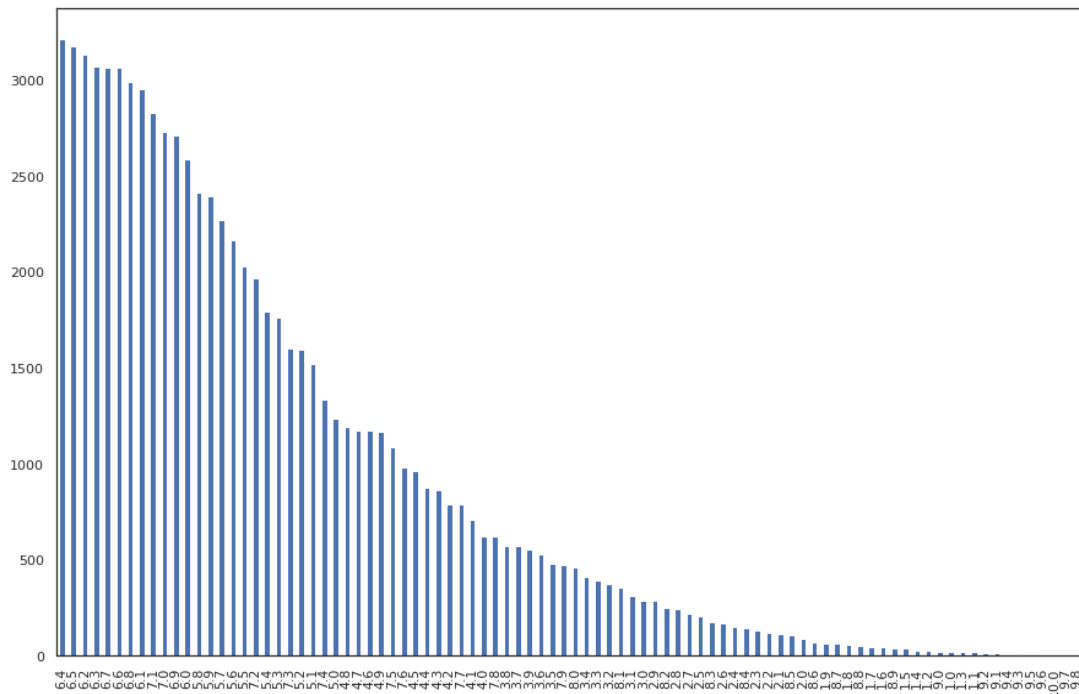


Graphiques

Nombre de films en fonction de l'année de sortie



Nombre de films par rapport aux avis des spectateurs

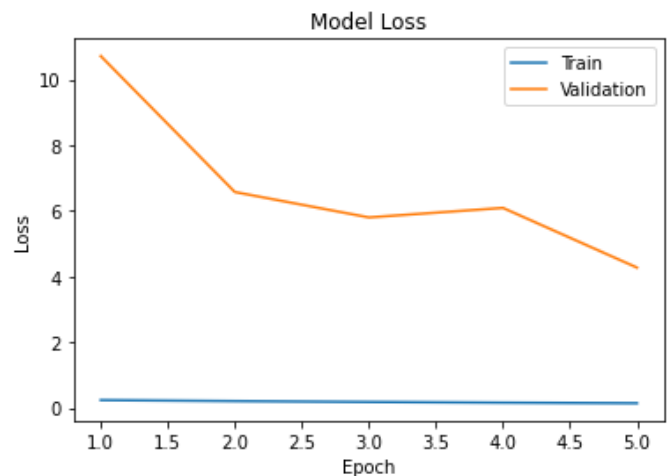
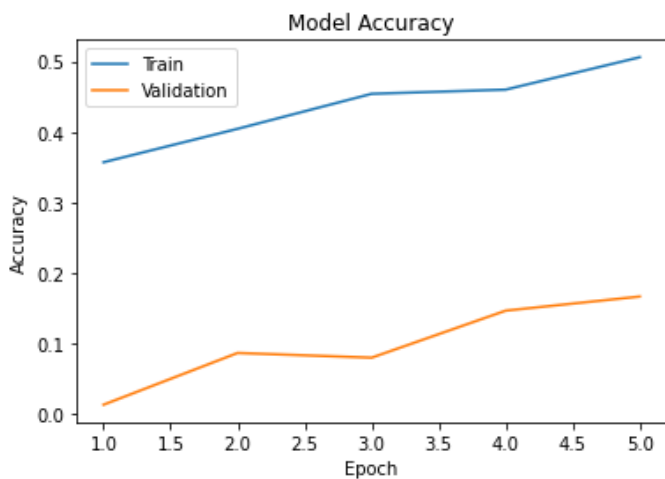


Algorithme de prédiction

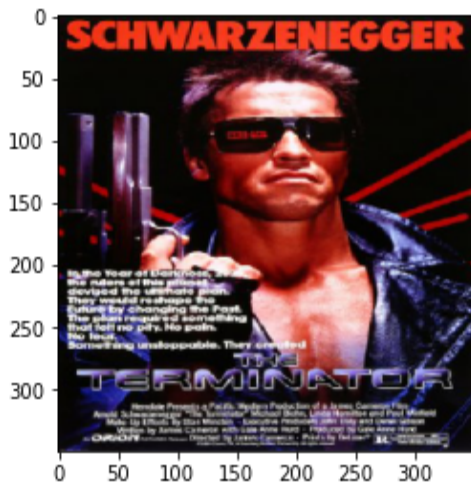
Pour notre algorithme de prédiction de genre nous avons réalisé un entraînement sur 1500 images avec 5 epoch pour une précision finale de 50%.

```
Epoch 1/5
43/43 [=====] - 56s 1s/step - loss: 0.2357 - accuracy: 0.3570 - val_loss: 10.7167 - val_accuracy: 0.0133
Epoch 2/5
43/43 [=====] - 58s 1s/step - loss: 0.2047 - accuracy: 0.4044 - val_loss: 6.5748 - val_accuracy: 0.0867
Epoch 3/5
43/43 [=====] - 57s 1s/step - loss: 0.1791 - accuracy: 0.4541 - val_loss: 5.8009 - val_accuracy: 0.0800
Epoch 4/5
43/43 [=====] - 57s 1s/step - loss: 0.1573 - accuracy: 0.4600 - val_loss: 6.0909 - val_accuracy: 0.1467
Epoch 5/5
43/43 [=====] - 58s 1s/step - loss: 0.1399 - accuracy: 0.5059 - val_loss: 4.2716 - val_accuracy: 0.1667
```

Ci-dessous les graphiques montrant l'évolution de la précision et des pertes en fonction du nombre d'époques.



Action
Drama
Adventure



Comme on peut l'observer, on a bien 1 chance sur 2 d'avoir la bonne prédiction de genre.

Ici, nous avons une accuracy de 50,59% donc nous avons 2 genres correctement prédits notamment Drama et Action mais pas Adventure.