

Annexe

I - Module de reconnaissance des fonctions mathématiques

1 - Description

Dans le cadre de ce projet, la reconnaissance des fonctions mathématiques se fait par trois étapes :

1. Identification et découpage des caractères présents dans l'image (Réalisée avec OpenCV)
2. Reconnaissance de chaque caractère a part (Image to String)
3. Tracer la fonction mathématique (Réalisée avec Matplotlib)

Cette partie de l'annexe traite majoritairement la deuxième étape, à savoir la reconnaissance des caractères (digits), qui repose sur un réseau neuronal.

2 - Extraction des caractéristiques (features)

Prétraitement

Tout d'abord, l'image contenant la fonction mathématique subit un pré-traitement afin de l'adapter au passage dans le modèle. Ce traitement suit les étapes suivantes :

- Seuillage pour éliminer les nuances de gris
- Premier Inversement des couleurs
- Déterminer des contours en délimitant les zones de couleur blanche (digits)
- Isoler le contenu de l'image entouré par les contours (crop)
- Second Inversement des couleurs
- Dilater le contenu de l'image

L'image qui suit montre l'input et l'output du prétraitement.

$$x \times x \times 2 + 2 \times x + 178$$



x

x

x

2

+

2

x

x

+

1

Extraction des caractéristiques

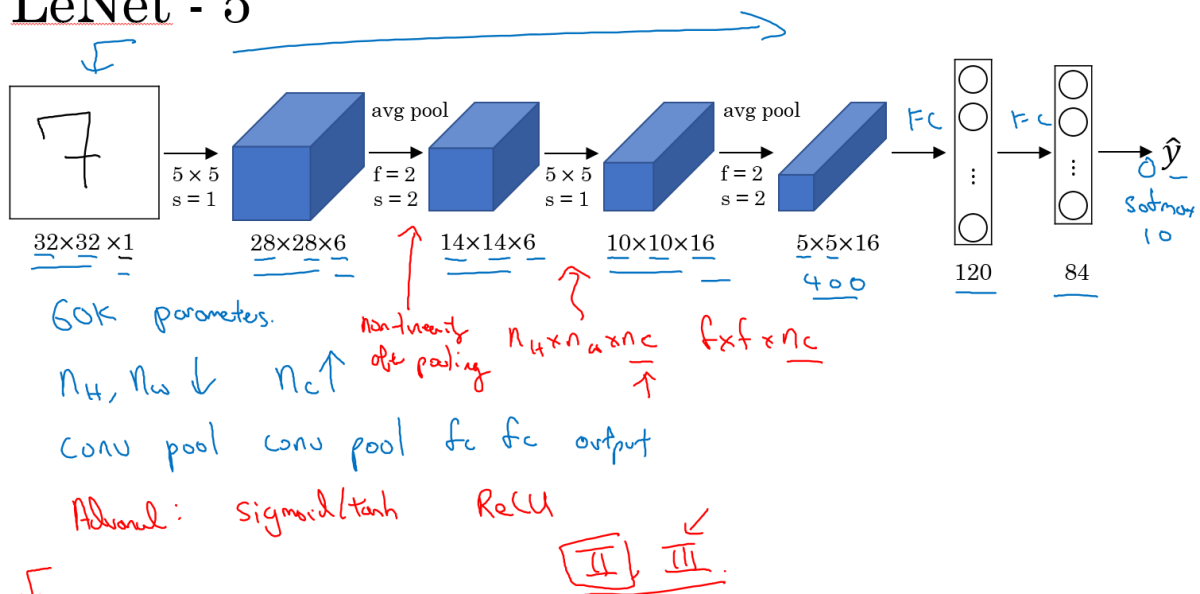
Le modèle étant un réseau neuronal, l'extraction des caractéristiques se fait implicitement dans les couches du réseau. La partie suivante détaille ces dernières.

3 - Entraînement du modèle

Architecture du réseau neuronal

Le modèle utilisé pour la reconnaissance des caractères (digits) s'inspire largement de l'architecture du fameux réseau neuronal « LeNet-5 » du Pr. Yann LeCun.

LeNet - 5



[LeCun et al., 1998. Gradient-based learning applied to document recognition]

Andrew Ng

Figure 1: LeNet-5

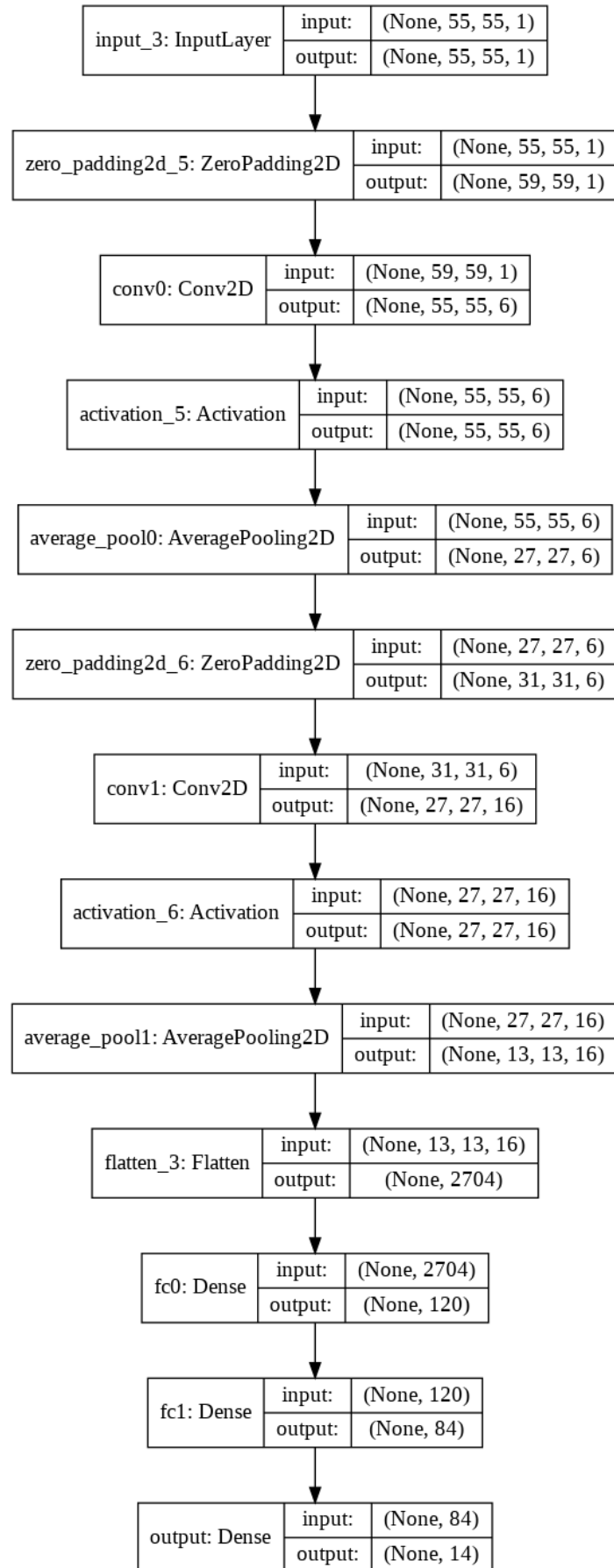


Figure 2: Le modèle utilisé dans le projet

L'image en entrée est de taille 55*55 en noir et blanc.

L'image passe dans une combinaison composée d'un zero-padding de taille (2,2), une couche de 6 filtres convolutionnels de taille (5,5), une fonction d'activation RELU puis d'un average pooling divisant en 2 la hauteur et la largeur de l'image.

Ensuite, l'image passe dans une combinaison composée d'un zero-padding de taille (2,2), une couche de 16 filtres convolutionnels de taille (5,5), une fonction d'activation RELU puis d'un average pooling de pas (2,2) divisant en 2 la hauteur et la largeur de l'image.

Finalement, l'image est aplatie en vecteur puis est passée à une succession de 2 couches neuronales denses.

La dernière couche représente la matrice de sortie montrant la probabilité d'appartenance de l'image à l'une des 14 classes utilisées pour classer les images en entrée.

CLASSES = ("+", "-", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "*", "x")

Training set et test set

Nous avons construit notre training set à partir d'une base de données de différents symboles mathématiques répartis en fichiers, chacun contenant de 3000 à 15000 images.

Lien : <https://www.kaggle.com/xainano/handwrittenmathsymbols/home>

Nous avons extrait 5000 images de la base de données par classe pour construire le training set. Pour pallier au problème de pénurie d'images dans la base de données, nous avons mis en œuvre des stratagèmes différents selon le digit concerné. Par exemple, dans le cas du digit « 8 », qui ne contient qu'environ 3000 images dans la base de données d'origine, nous avons copié une partie de la base de données, puis appliqué une rotation de 180 degrés à chaque copie afin d'atteindre 5000 exemples sans redondance de digits « 8 » déjà existants.

Les images de la base de données ayant la même maigre épaisseur, nous avons appliqué des degrés de dilatation différents à chaque image du training set pour prendre en considération l'épaisseur du digit dans la prédiction et éviter que des digits écrits avec une trop grosse épaisseur ne soient mal/pas reconnus.

Le test set, quand à lui, se compose d'images piochées dans la base de données ne se trouvant pas déjà dans le training set.

4 – Performance

Outils utilisés pour l'implémentation

Langage : Python

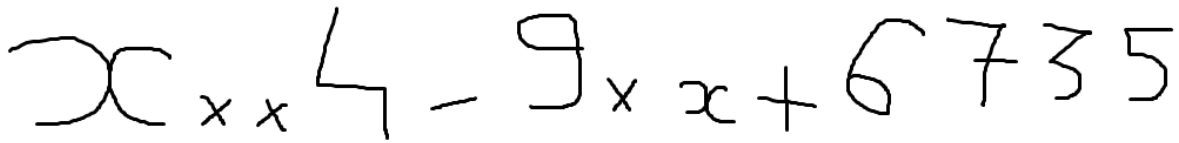
Librairies Python utilisées :

- OpenCV, pour toute opération impliquant le traitement d'image.
- Glob, pour la lecture optimisée de groupes de fichiers.
- Keras, pour la création, l'entraînement et la sauvegarde du modèle.
- Numpy, pour le traitement numérique de vecteurs et matrices.
- Matplotlib, pour tracer la fonction mathématique.

IDE : Jupyter Notebook

Résultats

Voici des screenshots illustrant les étapes du processus décrit dans la partie « Description »



A handwritten mathematical expression in black ink on a white background. The expression is $x^4 - 9x + 6735$. The 'x' is written as a simple curve, the '4' is a simple '4', the minus sign is a short horizontal line, the '9' is a simple '9', the plus sign is a simple '+', and the numbers '6735' are written in a simple, slightly irregular font.

Figure 3: Fonction 1

Après pré-traitement et passage dans le modèle, on obtient le résultat suivant :

L'equation a afficher : $x^{**4}-9*x+6735$

Predicted : x

Predicted : *

Predicted : *

Predicted : 4

Predicted : -

Predicted : 9

Predicted : *

Predicted : x

Predicted : +

Predicted : 6

Predicted : 7

Predicted : 3

Predicted : 5

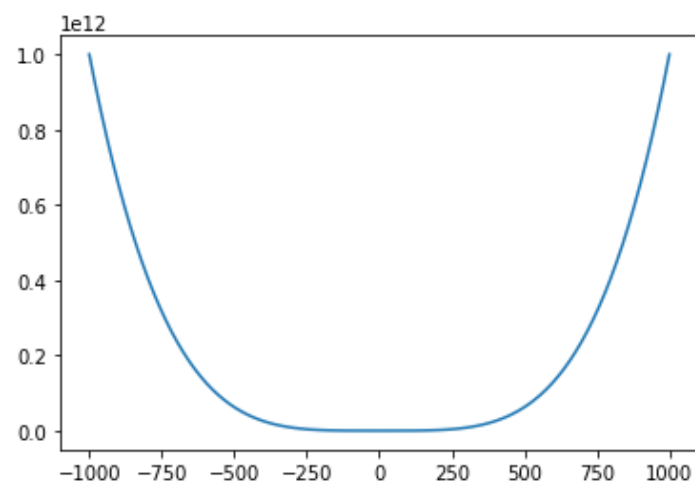


Figure 4: Reconnaissance et traçage de la fonction 1

$$x \times x 2 + 2 \times x + 178$$

Figure 5: Fonction 2

```
L'equation a afficher :x**2+2*x+178
Predicted : x
Predicted : *
Predicted : *
Predicted : 2
Predicted : +
Predicted : 2
Predicted : *
Predicted : x
Predicted : +
Predicted : 1
Predicted : 7
Predicted : 8
```

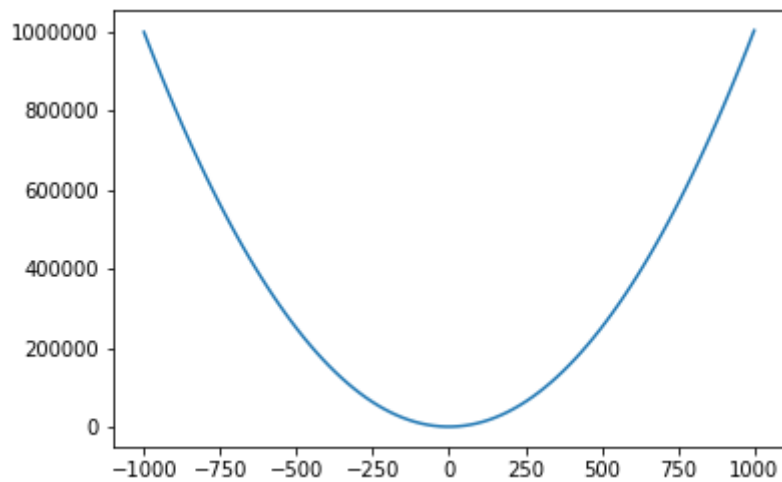


Figure 6: Reconnaissance et traçage de la fonction 2

II - Module de reconnaissance de commentaires haineux/toxiques

1 – Description

Ce modèle consiste en l'analyse du ton d'un commentaire (normal, toxique, haineux) et du degré d'intensité de sa toxicité (0 = normal, 1 = toxique).

2 - Extraction des caractéristiques (features)

Prétraitement

Les étapes que nous avons effectuées pour nettoyer les données du training set consistent, dans cet ordre :

- Éliminer la ponctuation et les chiffres
- Convertir le texte en minuscule
- Supprimer les mots rares

Extraction des caractéristiques

Nous avons décidé d'utiliser la représentation «Bag of Words», car elle est la plus efficace et la plus utilisée dans le domaine du NLP pour vectoriser les mots.

Le principe du « Bag Of Words » repose sur la constitution d'un dictionnaire de mots basé sur l'occurrence de ces derniers dans les différentes phrases du training set.

Exemple : Le training set se compose de ces deux phrases

John likes to watch movies. Mary likes movies too.

John also likes to watch football games.

La représentation BoW de ces deux phrases sera comme suit :

[1, 2, 1, 1, 2, 1, 1, 0, 0, 0]

[1, 1, 1, 1, 0, 0, 0, 1, 1, 1]

Où chaque chiffre représente le nombre d'occurrences de chaque mot dans un dictionnaire composé des deux phrases du training set. On obtient alors une matrice que l'on peut utiliser pour entraîner un modèle basé sur un algorithme de machine learning.

Optimisation

Afin de maximiser la qualité du processus d'extraction des caractéristiques, nous pouvons nettoyer davantage le vocabulaire en lui appliquant un algorithme TF-IDF (Term Frequency

times inverse document frequency), qui normalise la matrice, ce qui donne d'excellents résultats avec des textes longs, car se débarrasser des mots et les expressions récurrentes du training set entraîneront inévitablement une performance accrue.

$$\text{TF}(\text{word}, \text{text}) = \frac{\text{number of times the word occurs in the text}}{\text{number of words in the text}}$$

$$\text{IDF}(\text{word}) = \log \left[\frac{\text{number of texts}}{\text{number of texts where the word occurs}} \right]$$

$$\text{TF-IDF}(\text{word}, \text{text}) = \text{TF}(\text{word}, \text{text}) \times \text{IDF}(\text{word})$$

Figure 7: Principe de TF-IDF

3 - Entrainement du modèle

Algorithme utilisé

Un algorithme d'apprentissage automatique approprié pour la classification du texte est le « Naive Bayes ». Bien que simple dans son aspect mathématique, il s'est avéré plus performant, plus accessible et plus simple à comprendre que d'autres algorithmes bien connus de ML (tels que SVM - Support Vector Machines) après plusieurs essais. Nous avons utilisé la variante nommée « Multinomial Naïve Bayes », qui convient mieux à la classification de texte.

Training set et test set

Nous avons téléchargé le training et test set de la compétition Kaggle suivante :

Lien : <https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data>

4 – Performance

Outils utilisés pour l'implémentation

Langage : Python

Librairies Python utilisées :

- Scikit-learn, pour la création, l'entraînement et la sauvegarde du modèle,
- Pandas, pour la manipulation efficace et efficiente de DataFrames,
- Re, pour l'utilisation d'expressions regex afin de manipuler efficacement les chaînes de caractères,
- Csv, pour écrire dans des fichiers csv le résultat de nos prédictions.

IDE : Jupyter Notebook

Résultats

Après avoir lancé l'algorithme de détection de toxicité de commentaires sur le test set fourni par la compétition Kaggle, et après avoir enregistré les résultats des prédictions sur un fichier .csv, on obtient ceci :

comment_text	toxic
from rfc the title is fine as it is imo	0.0
sources zawe ashton on lapland	0.0
if you have look back at the source the information updated was the c	0.0
don anonymously edit articles at all	0.0
thank you for understanding think very highly of you and would not re	0.0
please do not add nonsense to wikipedia such edits are considered va	0.0
dear god this site is horrible	0.0
only fool can believe in such numbers the correct number lies between	0.0
double redirects when fixing double redirects don just blank the oute	0.0
think its crap that the link to roggienbier is to this article somebody tha	0.0
somebody will invariably try to add religion really you mean the way	0.0
february utc looking it over it clear that banned sockpuppet of ignore	0.0
it says it right there that it is type the type of institution is needed in	0.0
before adding new product to the list make sure it relevant before ad	0.0
current position anyone have confirmation that sir alfred is no longer	0.0
this other one from	0.0
reason for banning throwing this article needs section on why throwi	0.0
wallamoose was changing the cited material to say things the original	0.0
block from editing wikipedia	0.0
indefinitely block have indefinitely block this account	0.0
arabs are committing genocide in iraq but no protests in europe may	0.0
please stop if vou continue to vandalize wikipedia as vou did to homc	0.0

Figure 8: Exemples de commentaires non-toxiques

[illegible]

Figure 9: Exemples de commentaires toxiques