



Rapport de projet:

Sign language translation

membres:

- Elkanouni Samir
- Amadar Abdessamad
- El Hanafi Oussama
- Benmessaoud Aymane
- Belhaj Mohamed
- Benbouz Yassine

Encadrants:

- Madame Zrikem
- Monsieur El Bied
- Monsieur Hatim

Sommaire :

- Problématique
- Cahier de charge
- Analyse des besoins
- Outils & bibliothèques utilisées
- Réalisation du projet
- Les difficultés rencontrées et les solutions proposées.
- Remerciement.

Problématique :

La communication est l'une des choses les plus nécessaires dans la vie de l'être humain, tant qu'elle nous permet d'exprimer et communiquer avec l'autre, or il existe plusieurs gens qui n'arrivent pas à exprimer leurs idées et besoins, spécifiquement les mutismes, vu l'ignorance du langage de signe par la majorité des gens. Ce qui nous a amené à proposer une solution réalisable, abordable, convenable, afin de communiquer facilement avec eux. En effet, la communication avec les mutismes ne se fait que par des signes ou des gestes communs qui diffèrent selon le contexte et la région.

Alors, il faut un canal intermédiaire entre les mutismes et les gens qui n'arrivent pas à bien maîtriser leurs signes.

Notre idée est de réaliser des scripts en python qui peuvent gérer cette problématique.

Cahier de charge :

Pour réaliser notre projet, on propose de traduire le signe en texte, puis le convertir en audio (text-to-speech).

Pour se faire, on a besoin d'exploiter quelques notions d'intelligence artificielle, à savoir le Machine Learning et précisément le deep Learning, qui nous permettra de développer un modèle à base de l'intelligence artificielle.

La réalisation se décompose en plusieurs parties :

- Collection du data.
- Préparation du data.
- Création d'un modèle à base d'un réseau de neurones convolutifs (CNN).
- L'entraînement et le test du modèle.
- L'évaluation de la précision du modèle.
- La détection en temps réel.
- Assemblage des lettres et mots et construction d'une phrase qui va être affichée sur l'écran.
- Émettre le message sous forme audio.

Analyse des besoins :

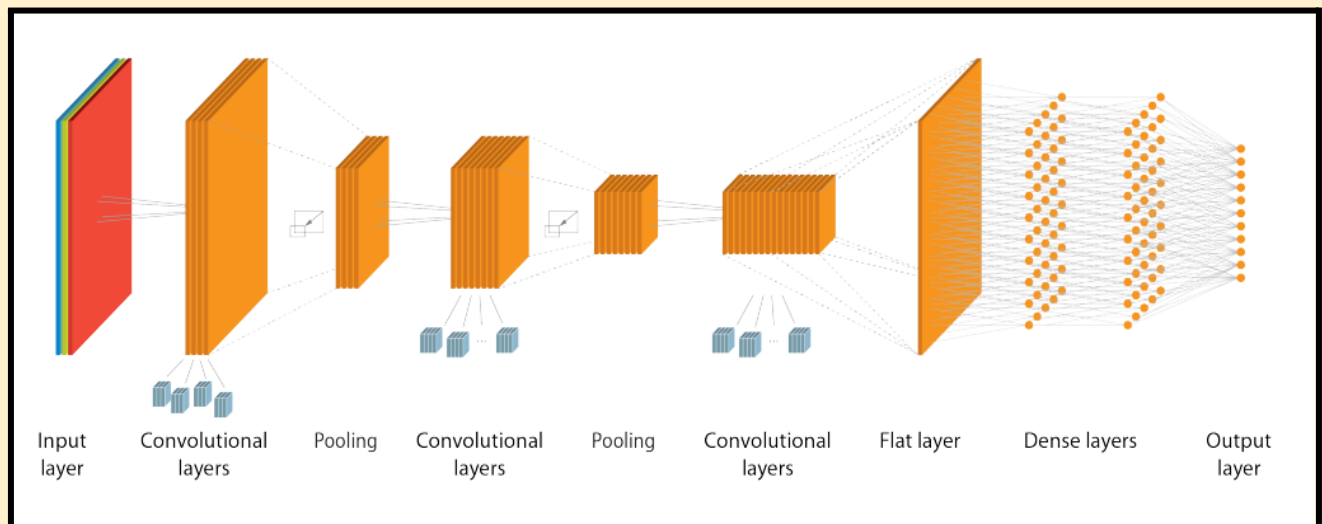
Notre projet doit être capable de distinguer entre les différents signes qui sont représentés sous forme d'images et qui dit image dit « pixels », il est difficile de réaliser ce projet avec la programmation classique, car cela nécessite des milliers de structures de contrôle. D'où la nécessité d'une autre méthode de réalisation, on parle donc de l'intelligence artificielle.

L'intelligence artificielle (AI) est un processus d'imitation de l'intelligence humaine qui repose sur la création et l'application d'algorithmes exécutés dans un environnement informatique dynamique.

Dans notre cas, il faut enseigner le système et le rendre capable de classifier des images selon le signe mentionné l'application de l'intelligence artificielle qui permet de réaliser ce processus est l'apprentissage automatique ou le Machine Learning (ML) ,ce dernier permet à l'ordinateur d'apprendre d'une manière implicite sans l'intervention de l'être humain ou une programmation explicite.

Le deep learning (DL), l'une des nombreuses approches du machine learning, utilise des réseaux de neurones profonds pour identifier des structures dans des volumes considérables de données. Dans le contexte informatique, les « réseaux neuronaux » sont des ensembles d'algorithmes modélisés sur la structure biologique du cerveau humain. Chaque réseau neuronal se concentre sur une couche spécifique de la tâche à apprendre (par exemple savoir si une caractéristique est présente ou non). C'est la superposition de ces couches de neurones qui permet au système de classifier une image à partir de ces caractéristiques.

Un réseau de neurone convolutif peut être représenté sous la forme suivante :

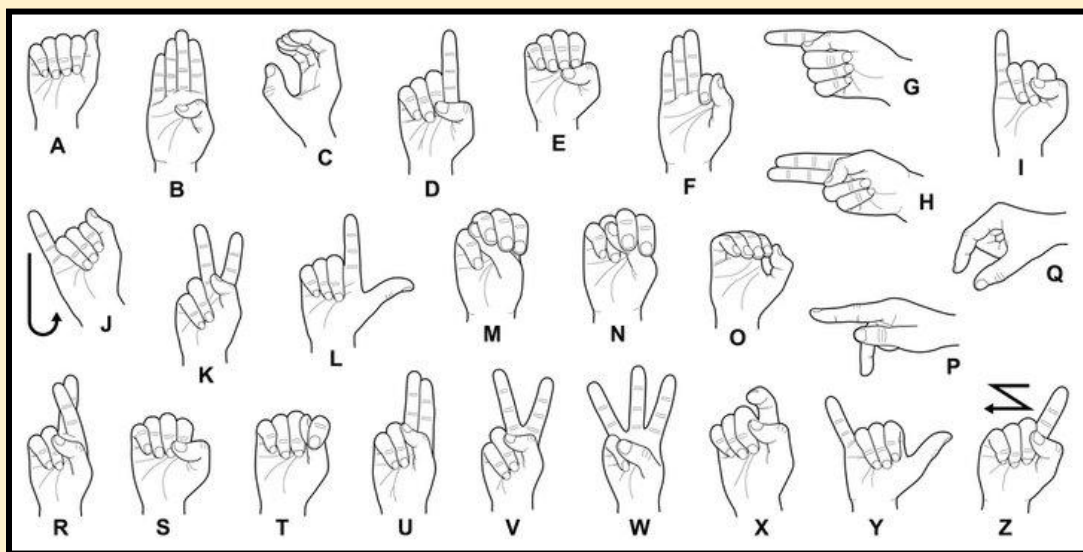


tous ces traitements nécessitent une machine puissante qui peut les gérer.

Le volume de données collectées pour entraîner et tester le modèle est très important pour augmenter le plus possible la précision du système (besoin des milliers d'images), or il est difficile de collecter ce nombre de données image par image, d'où la nécessité d'introduire un processus qui facilite la collection.

Pour réaliser chaque partie de notre projet, on a le choix entre plusieurs bibliothèques qui sont dédiées aux mêmes tâches, le choix va être effectué selon les bibliothèques les plus simples à apprendre.

il faut déterminer les signes, sur lesquels notre projet va se baser:



Outils & bibliothèques utilisées :

Python est déjà présent sous Linux ou Mac OS X et s'installe très facilement sous Windows. Toutefois, nous décrivons dans cet ouvrage l'utilisation de modules supplémentaires qui sont très utiles en Data Science (NumPy, tensorflow.keras, opencv, matplotlib, pandas, seaborn, os.system, sklearn), mais également les notebooks Jupyter.

On va donc utiliser un gestionnaire de paquets qui va installer ces modules supplémentaires. On souhaite également que ce gestionnaire de paquets soit disponible pour Windows.

Il y a deux grandes alternatives :

Anaconda: Anaconda est une distribution complète open source destiné à la programmation. Il est souvent utilisé en science de données, machine learning et l'intelligence artificielle car il peut contenir plusieurs packages nécessaires dans ce domaine notamment Numpy , tensorflow.keras, opencv, matplotlib,..

Pip : Pip est le gestionnaire de paquets de Python et qui est systématiquement présent depuis la version 3.4.

La plupart du travail de programmation est partagé sous la forme de code source brut ou bien en tant qu'exécutable compilé. Le code source fournit une information complète mais sous une forme qui « dit » davantage qu'elle ne montre. Quant à l'exécutable, il nous montre ce que fait le logiciel, mais même lorsqu'il est livré avec son code source, il peut être difficile de comprendre exactement comment il fonctionne.

Imaginez être capable de voir le code et de l'exécuter dans la même interface utilisateur, de telle façon qu'on peut modifier le code et voir les résultats de ces changements instantanément en temps réel.

C'est pourquoi on a choisi d'utiliser l'application web open source **Jupyter Notebook**. Cette dernière offre plusieurs bénéfices tel que :

- Le partage de code.
- La documentation de code.
- La visualisation de données.
- Les interactions en direct avec le code.


On doit mentionner aussi l'utilisation de la service cloud offert par Google appelé **Google Colab** ou **Colaboratory** qui est basé sur Jupyter Notebook.



Colab permet d'écrire et d'exécuter le code Python de notre choix par le biais du navigateur. C'est un environnement particulièrement adapté au machine learning, à l'analyse de données et à l'éducation. En termes plus techniques, Colab est un service hébergé de notebooks Jupyter qui ne nécessite aucune configuration et permet d'accéder **sans frais** à des ressources informatiques, dont des GPU.

En effet, les notebooks Colab sont stockés dans Google Drive. On peut aussi les charger depuis GitHub et les partager comme nous le ferions avec des documents Google Docs ou Sheets.

On va parler maintenant des modules utilisés, et on vous propose de lire le tableau ci- dessous:

bibliothèques	définition et utilisation
<p>NumPy</p> <div data-bbox="50 577 440 772">  NumPy </div>	<p>NumPy est une bibliothèque pour langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux.</p> <p>Plus précisément, cette bibliothèque logicielle libre et open source fournit de multiples fonctions permettant notamment de créer directement un tableau depuis un fichier ou au contraire de sauvegarder un tableau dans un fichier, et manipuler des vecteurs, matrices et polynômes.</p> <p>NumPy est la base de SciPy, regroupement de bibliothèques Python autour du calcul scientifique.</p>

Pandas



Le fonctionnement de Pandas repose sur les “DataFrames”: des tableaux de données en deux dimensions, dont chaque colonne contient les valeurs d’une variable et chaque ligne contient un ensemble de valeurs de chaque colonne. Les données stockées dans un DataFrame peuvent être des nombres ou des caractères.

os



Le module **os** de Python permet d'effectuer des opérations courantes liées au système d'exploitation. Il est indépendant par rapport au système d'exploitation de la machine. Ce qui signifie que ce module peut fonctionner sur n'importe quel système d'exploitation.

C'est est un module fournit par Python dont le but d'interagir avec le système d'exploitation, il permet ainsi de gérer l'arborescence des fichiers, de fournir des informations sur le système d'exploitation processus, variables systèmes, ainsi que de nombreuses fonctionnalités du systèmes

tensorflow.keras



TensorFlow est une plate-forme open source de bout en bout, une bibliothèque pour plusieurs tâches d'apprentissage automatique, tandis que Keras est une bibliothèque de réseau neuronal de haut niveau qui s'exécute au-dessus de :

TensorFlow. Les deux fournissent des API de haut niveau utilisées pour créer et former facilement des modèles, mais Keras est plus convivial car il est intégré à Python.

opencv



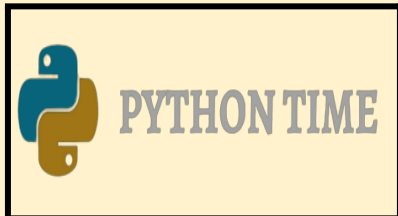
OpenCV (pour Open Computer Vision) est une bibliothèque graphique libre, initialement développée par Intel, spécialisée dans le traitement d'images en temps réel. Une image peut être mémorisée à l'intérieur d'une structure en C du type Cvmatrix ou IplImage. Ces structures sont issues des versions 1.x de Opencv. La structure IplImage est un vieux format d'image original compatible intel IPP.

sklearn



Scikit-learn est une bibliothèque clé pour le langage de programmation Python qui est généralement utilisé dans les projets d'apprentissage automatique. Scikit-learn se concentre sur les outils d'apprentissage automatique, y compris les algorithmes mathématiques, statistiques et à usage général qui constituent la base de nombreuses technologies d'apprentissage automatique.

time



Le module time en python est une des façons les plus simples de manipuler le temps dans un programme. Un temps en python est, par défaut, un nombre représentant des secondes.

Ce module fournit diverses fonctions liées au temps. Pour les fonctionnalités associées, voir aussi les modules datetime et calendrier.

Bien que ce module soit toujours disponible, toutes les fonctions ne sont pas disponibles sur toutes les plateformes. La plupart des fonctions définies dans ce module appellent les fonctions de la bibliothèque C de la plate-forme avec le même nom.

Réalisation du projet:

la réalisation de notre projet se décompose en trois parties intéressantes:

1-Collection du data:

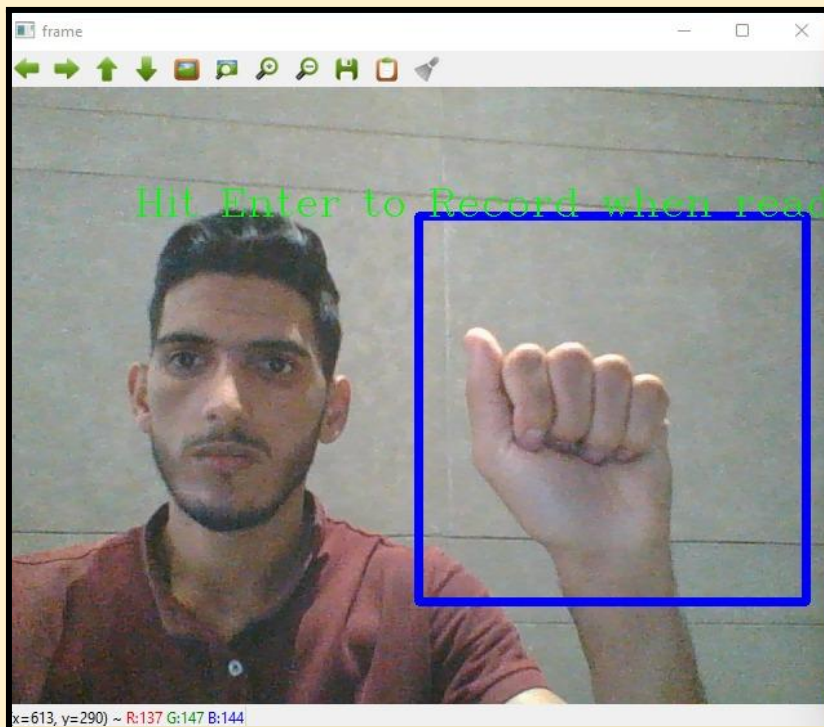
On a proposé de développer un script pour collecter les données pour chaque classe (les classes sont les lettres représentées dans notre cas: num_classes=25) selon le nombre suffisant pour un système précis (num_train_images=5000,num_test_images=200).

les données collectées (les images) se décomposent en deux parties, une pour les données spécifiées pour l'entraînement du modèle ,et l'autre pour les données spécifiées pour le test.

l'idée est de capturer plusieurs images en quelques secondes , pour faciliter la collection.

On a créé un processus à l'aide du librairie OpenCV qui peut être associée soit à la caméra de notre machine, soit à celle du téléphone portable pour une meilleure qualité.

Puisque la partie intéressante de la fenêtre est celle qui contient la main, on fait extraire juste cette partie par la création d'un roi dans la fenêtre, cet roi est placé par suite dans une autre fenêtre:



un ligne de code change la taille du roi en 28x28, ainsi son format en "gray scale" (blanc noir) et le place dans une fenetre pour visualiser les changements effectuees:



Pour faciliter l'utilisation du processus de la collection, puisqu'on a besoin de milliers d'images, la collection se fait classe par classe.

Les images collectées sont donc de taille 28x28 (les images extraites à partir du roi), elles sont stockées dans des dossiers nommés de 0 à 24 (25 classes/lettres), selon la catégorie (train/test) et la classe correspondante:

0:	'A',
1:	'B',
2:	'C',
3:	'D',
4:	'E',
5:	'F',
6:	'G',
7:	'H',
8:	'I',
9:	'K',
10:	'L',
11:	'M',

12:	'N',
13:	'O',
14:	'P',
15:	'Q',
16:	'R',
17:	'S',
18:	'T',
19:	'U',
20:	'V',
21:	'W',
22:	'X',
23:	'Y' }

La classe numéro 24 représente l'espace.

Pour préparer ces images, on les transforme premièrement en une matrice en une seule ligne qui contient tous les pixels d'une seule image, et on les rassemble ligne par ligne dans un tableau qui contient en colonne le numéro des pixels et en ligne le numéro de classe correspondante, ces étapes sont effectués à l'aide de la librairie pandas :

	label	pixel0	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	...	pixel774	pixel775	pixel776	pixel777	pixel778	pixel779
0	0	210.0	210.0	211.0	212.0	209.0	215.0	213.0	213.0	214.0	...	206.0	207.0	209.0	211.0	211.0	211.0
1	0	210.0	210.0	209.0	212.0	213.0	210.0	214.0	214.0	214.0	...	206.0	207.0	209.0	210.0	210.0	212.0
2	0	210.0	211.0	208.0	214.0	212.0	210.0	216.0	213.0	215.0	...	207.0	209.0	209.0	210.0	211.0	211.0
3	0	209.0	210.0	211.0	211.0	215.0	210.0	214.0	214.0	215.0	...	206.0	207.0	210.0	211.0	210.0	211.0
4	0	210.0	208.0	210.0	214.0	213.0	211.0	213.0	214.0	215.0	...	207.0	207.0	210.0	210.0	210.0	211.0
...
2495	4	208.0	208.0	211.0	210.0	210.0	213.0	212.0	214.0	213.0	...	204.0	206.0	207.0	208.0	210.0	209.0
2496	4	210.0	208.0	210.0	209.0	211.0	211.0	211.0	215.0	214.0	...	209.0	206.0	207.0	209.0	210.0	209.0
2497	4	207.0	208.0	207.0	209.0	212.0	213.0	211.0	213.0	213.0	...	203.0	205.0	207.0	207.0	209.0	212.0
2498	4	209.0	209.0	208.0	209.0	212.0	211.0	210.0	213.0	213.0	...	196.0	206.0	207.0	208.0	209.0	210.0
2499	4	207.0	210.0	210.0	210.0	211.0	213.0	210.0	213.0	213.0	...	170.0	196.0	206.0	208.0	208.0	210.0

2500 rows × 785 columns

Pour enregistrer ces données, on les place dans un fichier excel (.csv) .

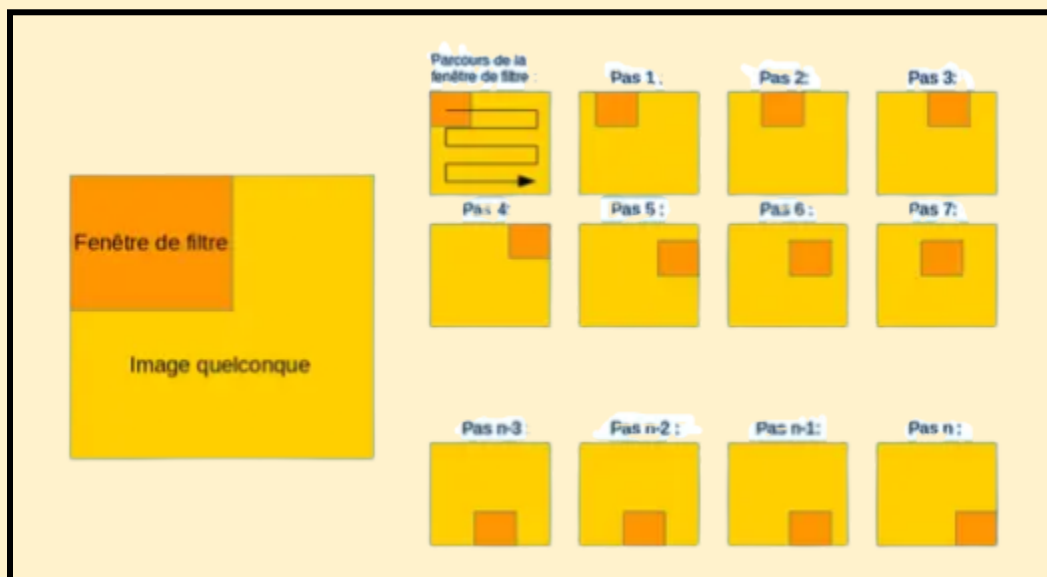
2-Création d'un modèle à base d'un réseau de neurones convolutifs (CNN):

cette partie se fait sur la plateforme de google colab ,il faut d'abord apporter nos fichiers excel et l'uploader sur

le plateforme après l'installation d'une connexion avec une hardware de google:

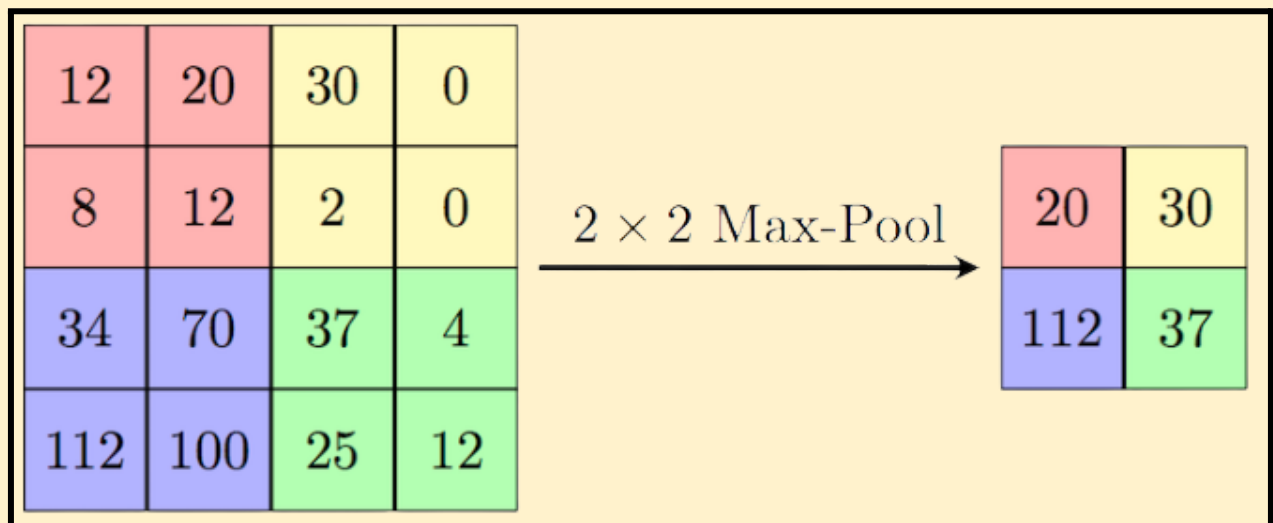
La création de ce modèle se fait en utilisant la librairie keras,et suite à plusieurs critères que l'on suit,on construire ce modèle qui se décompose en plusieurs parties (couches):

Couche de convolution (CONV) : Le rôle de cette première couche est d'analyser les images fournies en entrée et de détecter la présence d'un ensemble de features. On obtient en sortie de cette couche un ensemble de features maps.



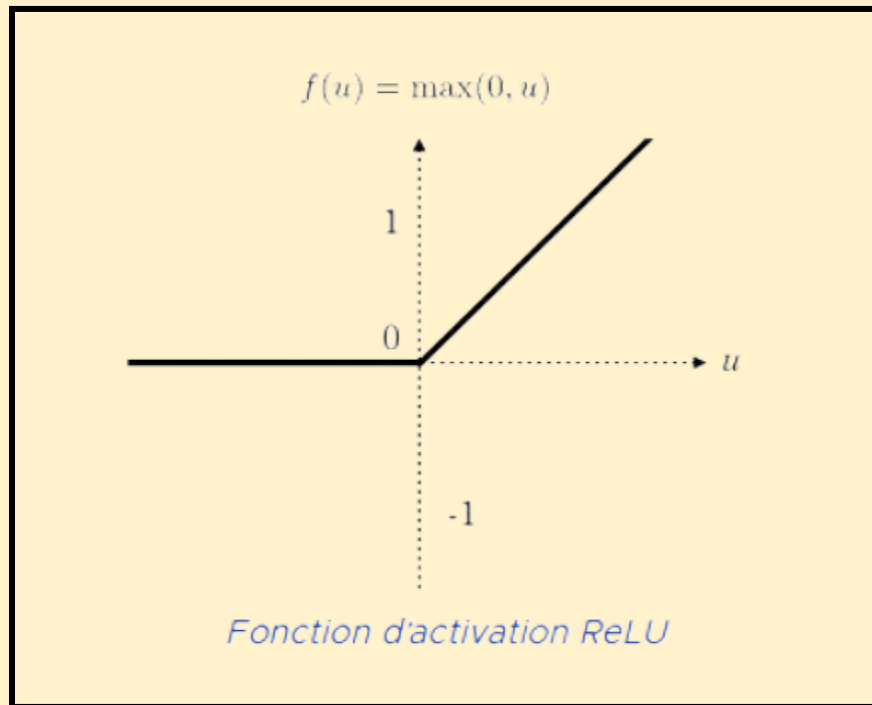
Couche de Pooling (POOL) : La couche de Pooling est une opération généralement appliquée entre deux couches de

convolution. Celle-ci reçoit en entrée les features maps formées en sortie de la couche de convolution et son rôle est de réduire la taille des images, tout en préservant leurs caractéristiques les plus essentielles. Parmi les plus utilisés, on retrouve le max-pooling mentionné précédemment ou encore l'average pooling dont l'opération consiste à conserver à chaque pas, la valeur moyenne de la fenêtre de filtre.

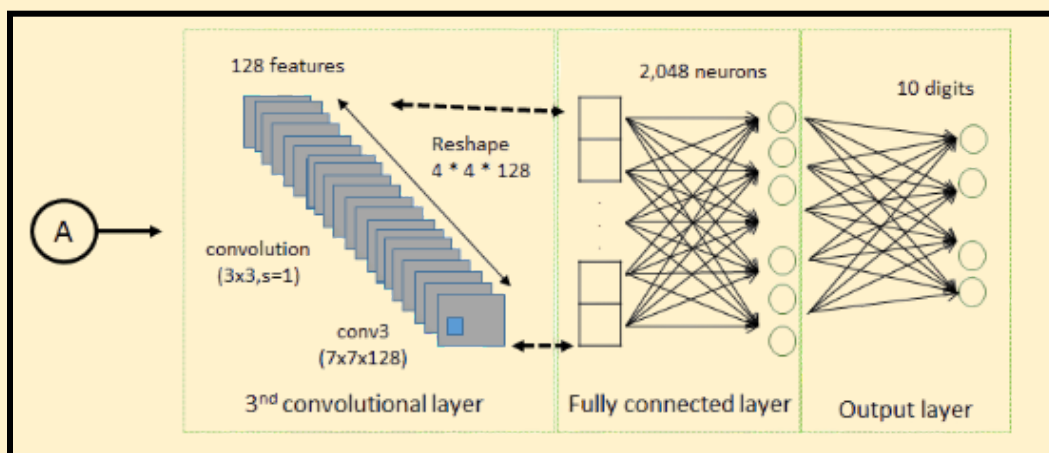


La couche d'activation ReLU (Rectified Linear Units) :

Cette couche remplace toutes les valeurs négatives reçues en entrées par des zéros. L'intérêt de ces couches d'activation est de rendre le modèle non linéaire et de ce fait plus complexe.



Couche Fully Connected (FC) : Ces couches sont placées en fin d'architecture de CNN et sont entièrement connectées à tous les neurones de sorties (d'où le terme fully-connected)



dans notre modèle nous avons créé trois filtres de convolution 3×3 dans la première couche, chaque filtre contient 64 noeuds (chaque filtre va effectuer 64 passage

sur l'image) avec une activation Relu, on effectue un max pool 2x2 sur la sortie de chaque filtre alors le sommaire de notre modèle de la forme:

Model: "sequential"		
Layer (type)	Output Shape	Param #
=====		
conv2d (Conv2D)	(None, 26, 26, 64)	640
max_pooling2d (MaxPooling2D)	(None, 13, 13, 64)	0
conv2d_1 (Conv2D)	(None, 11, 11, 64)	36928
max_pooling2d_1 (MaxPooling2D)	(None, 5, 5, 64)	0
conv2d_2 (Conv2D)	(None, 3, 3, 64)	36928
max_pooling2d_2 (MaxPooling2D)	(None, 1, 1, 64)	0
flatten (Flatten)	(None, 64)	0
dense (Dense)	(None, 128)	8320
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 24)	3096

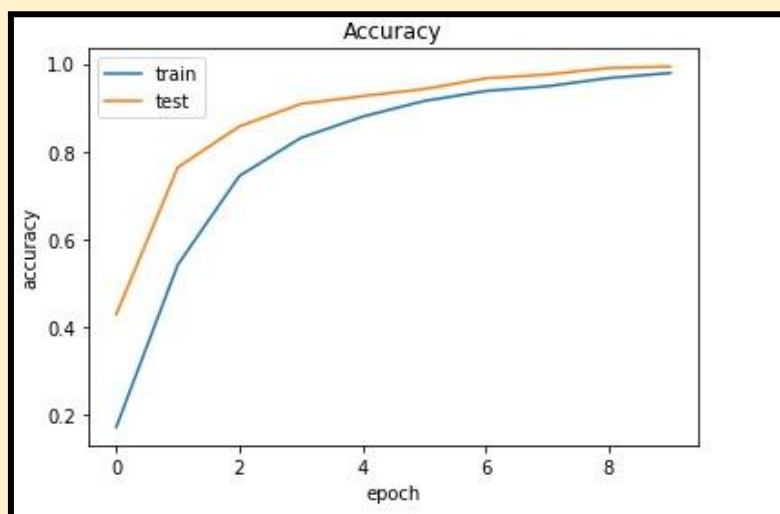
3-l'entraînement et le teste du modèle:

pour entraîner le modèle, on le compile en précisant le nombre de fois que l'on veut faire l'entrainement de notre modèle (epochs=10),le nombre d'images par chaque epoch(batch_size=128), la précision augmente epoch par epoch,par contre l'erreur diminue:

```
# Train our Model
history = model.fit(x_train, y_train, validation_data = (x_test, y_test), epochs=epochs, batch_size=batch_size)
```

```
Epoch 1/10
151/151 [=====] - 30s 194ms/step - loss: 2.6938 - accuracy: 0.1713 - val_loss: 1.8042 - val_accuracy: 0.4284
Epoch 2/10
151/151 [=====] - 28s 185ms/step - loss: 1.3626 - accuracy: 0.5416 - val_loss: 0.8202 - val_accuracy: 0.7635
Epoch 3/10
151/151 [=====] - 28s 189ms/step - loss: 0.7679 - accuracy: 0.7439 - val_loss: 0.4852 - val_accuracy: 0.8569
Epoch 4/10
151/151 [=====] - 28s 184ms/step - loss: 0.5093 - accuracy: 0.8308 - val_loss: 0.3215 - val_accuracy: 0.9081
Epoch 5/10
151/151 [=====] - 28s 184ms/step - loss: 0.3629 - accuracy: 0.8792 - val_loss: 0.2319 - val_accuracy: 0.9258
Epoch 6/10
151/151 [=====] - 28s 183ms/step - loss: 0.2580 - accuracy: 0.9148 - val_loss: 0.1798 - val_accuracy: 0.9420
Epoch 7/10
151/151 [=====] - 28s 184ms/step - loss: 0.1920 - accuracy: 0.9376 - val_loss: 0.1198 - val_accuracy: 0.9665
Epoch 8/10
151/151 [=====] - 28s 185ms/step - loss: 0.1535 - accuracy: 0.9483 - val_loss: 0.0843 - val_accuracy: 0.9751
Epoch 9/10
151/151 [=====] - 28s 184ms/step - loss: 0.1094 - accuracy: 0.9671 - val_loss: 0.0501 - val_accuracy: 0.9900
Epoch 10/10
151/151 [=====] - 28s 185ms/step - loss: 0.0766 - accuracy: 0.9786 - val_loss: 0.0405 - val_accuracy: 0.9931
```

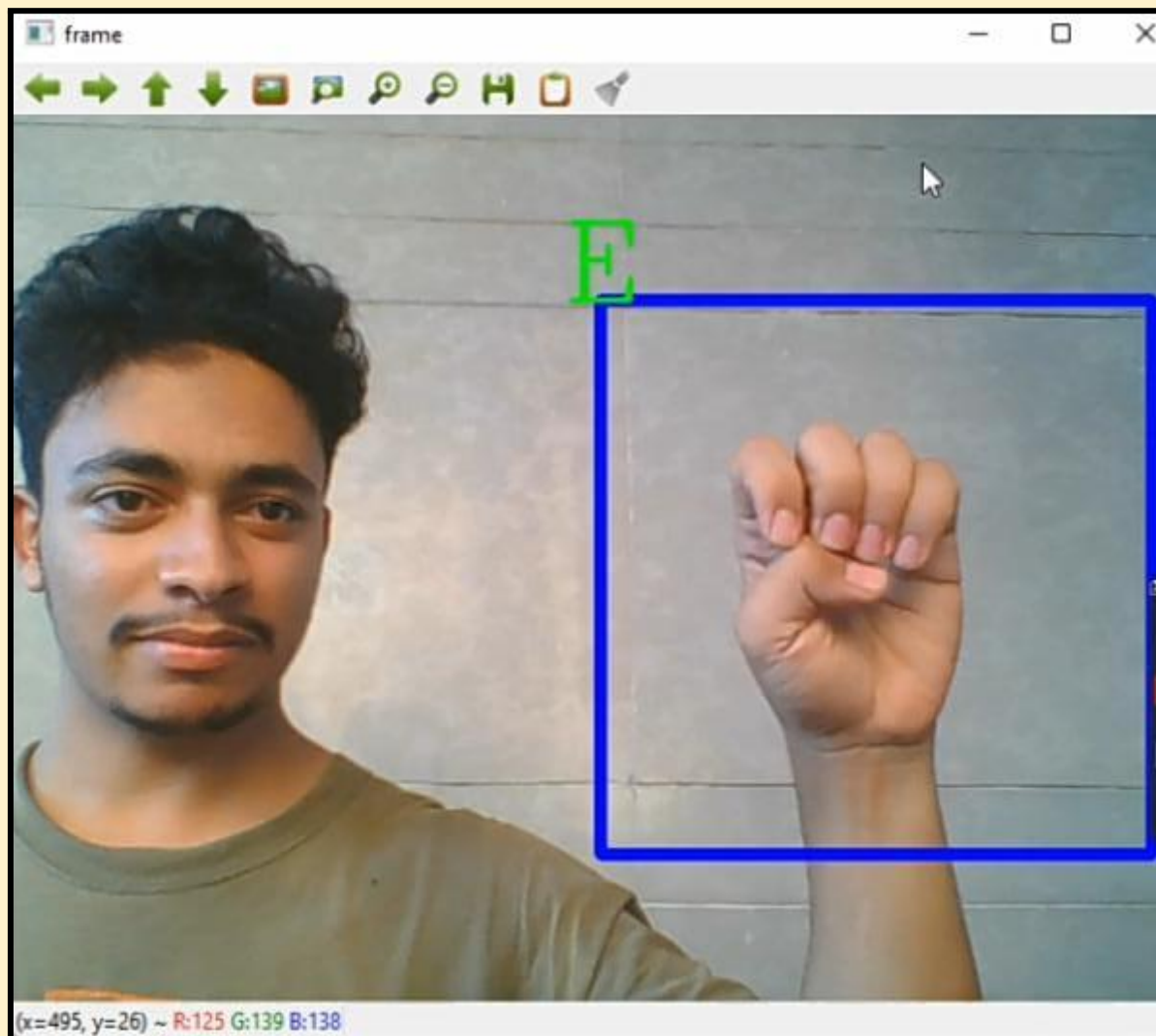
on visualise la variation de la précision:



Après avoir entraîné notre modèle correctement, l'étape qui suit est le test, en passant les données collectées précisément pour le test , le modèle doit faire la prédiction de la classe correspondante et on mesure l'erreur entre ces prédictions et la valeur exacte pour évaluer la précision finale du modèle. dans notre cas la précision est 0.876 (87,6%)

3-détection en temps réel:

Après l'affectation de chaque classe (nombre entre 0 et 24) avec sa lettre correspondante à l'aide d'un dictionnaire, on passe au chargement de notre modèle qui a été entraîné sur google colab et ensuite on commence la détection en temps réel par le lancement d'une vidéo qui capte notre signes et les traduise en même temps en lettre correspondante, la lettre capter est affichée en haut du roi. NB:il faut placer la main dans le roi , pour saisir la lettre trouvée, on clique sur "Entrer" pour construire la phrase voulue.



l'émission de la phrase constituée se fait à l'aide des bibliothèques de gtts (google text to speech) et OS.

Les difficultés rencontrées et les solutions proposées :

Durant la réalisation de notre projet on a rencontré plusieurs problèmes et difficultés, ce qui a arrêté le déroulement du travail dans certaines périodes. Parmi ces difficultés nous citons :

- la mise à jour continue de certaines bibliothèques, ce qui nécessite à chaque fois la recherche des dernières versions de celle-ci puis modifier notre code.

- nous avons trouvé des difficultés en ce qui concerne la compatibilité entre les bibliothèques, et pour résoudre ce problème, on a créé des environnements virtuels pour rassembler les bibliothèques compatibles entre eux.

- on n'a pas pu inclure les lettres J et Z, puisqu'ils sont représentés par des mouvements.

- nos machines n'ont pas supporté la partie de l'entraînement et le teste du modèle, donc on a proposé d'utiliser l'espace de google colab, d'une autre part les librairies utilisées pour faire la détection en temps réel ne fonctionnent pas correctement dans cet espace, alors l'idée est de faire l'entraînement et le teste du système

dans google colab, puis la détection en temps réel dans l'environnement virtuel créer dans la machine.

-l'apprentissage des notions de base du Machine Learning nous a pris beaucoup de temps avant le début de la réalisation du projet.

-manque de stabilité du système.

-la ressemblance entre les signes de certaines lettres nous a posé des problèmes au niveau de la précision du système.

Remerciement :

Nous tenons à remercier sincèrement nos professeurs encadrants , madame ZRIKEM ,monsieur El Bied ,et monsieur Anas Hatim pour leurs efforts ,et leur disponibilité,ainsi que leurs conseils, durant toutes les périodes de travail. Grâce à eux on a pu résoudre plusieurs problèmes afin de bien finaliser notre projet.