Chapitre 1

Manupilaction de la base

1.1 Pourquoi on à choisie cette base de donées

Représentation complète de l'alphabet : Cette base de données offre une représentation complète de l'alphabet, ce qui permet de couvrir l'ensemble des gestes et signes nécessaires pour la communication en langue des signes. Chaque fichier représente un caractère spécifique, ce qui facilite l'apprentissage et la prédiction des gestes correspondants.

Diversité des gestes : Les images de mains représentant les caractères dans la base de données offrent une diversité de gestes. Les positions, les mouvements et les configurations des doigts peuvent varier d'une image à l'autre, ce qui permet de capturer la variabilité naturelle des gestes dans la langue des signes.

Apprentissage des caractéristiques visuelles : En utilisant cette base de données, le modèle de prédiction peut apprendre à reconnaître les caractéristiques visuelles spécifiques à chaque geste. Les différentes images de mains représentant le même caractère permettent de capturer les variations et d'apprendre à généraliser ces caractéristiques pour une prédiction précise.

Classification précise des gestes : Avec une base de données organisée par caractère, il est plus facile d'effectuer une classification précise des gestes. Chaque fichier représente un geste spécifique, ce qui facilite l'étiquetage et l'entraînement du modèle pour reconnaître chaque caractère de manière distincte.

1.2 Extraction des features :

1.2.1 HOG

L'extraction des caractéristiques HOG (Histogram of Oriented Gradients) est une technique largement utilisée pour la détection d'objets et de motifs dans les images. Dans le contexte de la détection de la langue des signes, elle permet de capturer les informations importantes des gestes des mains. Voici comment se déroule le processus d'extraction des caractéristiques HOG :

1. Prétraitement de l'image : Tout d'abord, l'image contenant le geste de la main est prétraitée pour améliorer la qualité et réduire les variations indésirables. Cela peut inclure des opérations telles que la normalisation de la luminosité, la réduction du bruit ou l'égalisation de l'histogramme.
2. Calcul des gradients : Les gradients de l'image sont calculés pour capturer les variations de luminosité et les contours. Cela se fait généralement en utilisant des opérateurs de détection de gradients tels que le gradient de Sobel ou le gradient de Prewitt. Les gradients fournissent des informations sur la direction et l'intensité des changements de couleur ou de luminosité dans l'image.
3. Division de l'image en cellules : L'image est ensuite divisée en petites cellules carrées ou rectangulaires. Chaque cellule contient un certain nombre de pixels et servira à calculer les histogrammes de gradients locaux.
4. Calcul des histogrammes de gradients : Pour chaque cellule, les orientations des gradients calculés précédemment sont utilisées pour construire un histogramme. Les orientations des gradients sont regroupées en plusieurs "bins" (intervalles) pour former l'histogramme. La magnitude des gradients peut également être prise en compte pour pondérer les contributions des gradients.
5. Normalisation des blocs : Les cellules sont regroupées en blocs pour capturer les relations spatiales entre les différentes parties de l'image. Les blocs se chevauchent généralement pour couvrir l'ensemble de l'image. Les histogrammes de gradients des cellules à l'intérieur de chaque bloc sont normalisés pour réduire les variations d'éclairage locales.
6. Concaténation des vecteurs de caractéristiques : Les histogrammes de gradients normalisés sont concaténés pour former un vecteur de caractéristiques global représentant le geste de la main dans l'image.

1.2.2 LBP

LBP fonctionne en examinant chaque pixel d'une image et en comparant son intensité à celle de ses voisins. Pour chaque pixel, une valeur binaire est calculée en fonction de la relation entre les intensités des pixels voisins et celle du pixel central. Ces valeurs binaires sont ensuite combinées pour former un motif local, qui représente les variations locales de l'intensité de l'image.LBP présente plusieurs avantages dans le contexte de l'analyse des images de mains pour la reconnaissance des gestes dans la langue des signes :

Robustesse aux variations d'éclairage : LBP est relativement robuste aux variations d'éclairage, ce qui en fait une méthode adaptée à l'analyse d'images de mains qui peuvent présenter des variations d'intensité en raison de conditions d'éclairage changeantes.

Calcul efficace : Le calcul des motifs LBP est rapide et efficace, ce qui le rend adapté à l'analyse en temps réel des gestes de la langue des signes.

Représentation locale des caractéristiques : LBP capture les variations locales de l'intensité de l'image, ce qui permet de représenter les caractéristiques importantes des gestes dans une région spécifique de la main.

Représentation locale des caractéristiques :LBP capture les variations locales de l'intensité de l'image, ce qui permet de représenter les caractéristiques importantes des gestes dans une région spécifique de la main.

1.3 Les algorithmes utilisés

1.3.1 SVM

SVM (Support Vector Machine) est un algorithme d'apprentissage supervisé utilisé dans le domaine de l'apprentissage automatique pour la classification et la régression.SVM fonctionne en construisant un hyperplan optimal qui sépare les données en différentes classes. L'objectif est de maximiser la marge, c'est-à-dire la distance entre l'hyperplan et les échantillons les plus proches de chaque classe. Les échantillons les plus proches de l'hyperplan sont appelés vecteurs de support, d'où le nom de l'algorithme.Lorsqu'il est utilisé pour la reconnaissance de gestes dans la langue des signes, SVM peut être entraîné à reconnaître différents gestes en utilisant des caractéristiques extraites des images de mains.

1.3.2 KNN

L'algorithme des k plus proches voisins (KNN - k-nearest neighbors) est un algorithme de classification supervisée largement utilisé en apprentissage automatique. Voici comment fonctionne l'algorithme de KNN : Collecte des données (par exemple, dans le contexte de la langue des signes, un exemple pourrait être un geste de la main avec sa signification associée), Calcul de la similarité (La distance euclidienne est souvent utilisée pour calculer la distance entre les caractéristiques des exemples), Sélection des k plus proches voisins, Vote majoritaire, Classification.

1.3.3 Decision tree

L'arbre de décision (Decision Tree) est un algorithme d'apprentissage automatique supervisé utilisé pour la classification et la régression.Un arbre de décision est une structure en forme d'arbre composée de nœuds de décision et de feuilles. Chaque nœud de décision représente un test sur une caractéristique spécifique, et chaque branche de l'arbre correspond à un résultat possible du test. Les feuilles de l'arbre contiennent les étiquettes de classe ou les valeurs de sortie.

Dans le contexte de la reconnaissance de gestes dans la langue des signes, un arbre de décision peut être utilisé pour apprendre à classer différents gestes en fonction de caractéristiques extraites des images de mains. Les caractéristiques peuvent inclure des informations sur la position des doigts, les mouvements ou la forme générale de la main.

1.3.4 Perceptron

L'algorithme du Perceptron est un algorithme d'apprentissage automatique supervisé utilisé pour la classification binaire. Il est considéré comme l'un des premiers algorithmes d'apprentissage automatique. Voici comment fonctionne l'algorithme du Perceptron : Collecte des données : (par exemple, dans le contexte de la détection de la langue des signes, un exemple pourrait être un geste de la main étiqueté comme "positif" ou "négatif"). Initialisation des poids et du biais, Calcul de la sortie, Mise à jour des poids, après les étapes précédentes sont répétées pour chaque exemple d'entraînement jusqu'à ce que le Perceptron atteigne une convergence et enfin Classification.

1.3.5 Régression Logistique

La régression logistique est un algorithme d'apprentissage supervisé utilisé pour la classification binaire, c'est-à-dire pour prédire des valeurs appartenant à deux catégories distinctes.L'algorithme de régression logistique modélise la relation entre les variables indépendantes (caractéristiques) et la variable cible en utilisant une fonction logistique. Cette fonction logistique comprime les valeurs continues en une plage de probabilités entre 0 et 1. Le modèle utilise ensuite ces probabilités pour prédire la classe à laquelle chaque observation appartient.Dans le contexte de la reconnaissance de gestes dans la langue des signes, la régression logistique peut être utilisée pour prédire la classe d'un geste donné en fonction des caractéristiques extraites des images de mains.

1.3.6 Random Forest

L'idée principale derrière Random Forest est de construire un grand nombre d'arbres de décision indépendants et de les combiner pour obtenir une prédiction finale. Chaque arbre est construit sur un sous-ensemble aléatoire des données d'entraînement, ainsi que sur un sous-ensemble aléatoire des caractéristiques. Cette aléatorisation permet d'introduire de la diversité dans les arbres et de réduire le risque de surapprentissage.

Lorsqu'il s'agit de classification, chaque arbre de décision dans la forêt aléatoire prédit la classe à laquelle une observation appartient, et la classe prédite est déterminée par un vote majoritaire parmi tous les arbres. Pour la régression, la prédiction finale est souvent obtenue en prenant la moyenne des prédictions de tous les arbres.

1.4 Optimisation des algorithmes

1.4.1 Grid Search

La recherche par grille (Grid Search) est une technique utilisée pour trouver les meilleurs hyperparamètres pour un algorithme d'apprentissage automatique. Les hyperparamètres sont des paramètres qui ne sont pas appris directement à partir des données, mais qui déterminent la configuration du modèle.

La recherche par grille consiste à définir une grille de valeurs possibles pour chaque hyperparamètre, puis à évaluer le modèle pour chaque combinaison de valeurs dans la grille. Cela permet

de déterminer quelle combinaison d'hyperparamètres donne les meilleures performances du modèle sur un jeu de données de validation ou de test.

1.4.2 Random Search

La recherche aléatoire (Random Search) est une technique utilisée pour trouver les meilleurs hyperparamètres pour un algorithme d'apprentissage automatique. Elle diffère de la recherche par grille en ce sens qu'elle explore de manière aléatoire l'espace des hyperparamètres au lieu d'évaluer toutes les combinaisons possibles.

1.4.3 Elbow method

La méthode du coude, également connue sous le nom de "elbow method" en anglais, est une technique utilisée pour déterminer le nombre optimal de clusters dans une analyse de regroupement (clustering). Elle est souvent utilisée en conjonction avec l'algorithme du k-NN, bien que la méthode puisse être appliquée à d'autres méthodes de regroupement.

1.4.4 cross validation

La validation croisée (cross-validation en anglais) est une technique utilisée pour évaluer les performances d'un modèle d'apprentissage automatique et est particulièrement utile lorsque les données sont limitées. Elle permet d'estimer la capacité de généralisation d'un modèle en utilisant des échantillons de données d'entraînement et de test.

Chapitre 2

Implémentation

2.1 comparaison des performances des modéls

2.1.1 HOG

2.1.1.1 Accuaracy

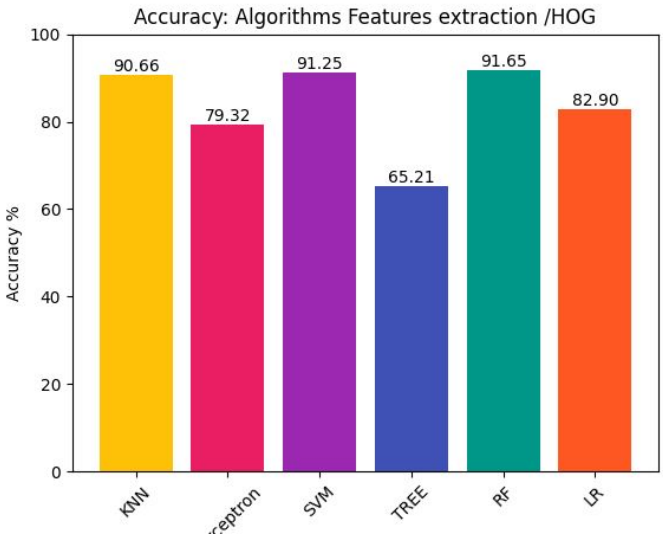

FIGURE 2.1 – Models accuaracy via HOG

2.1.1.2 Temps d'execution

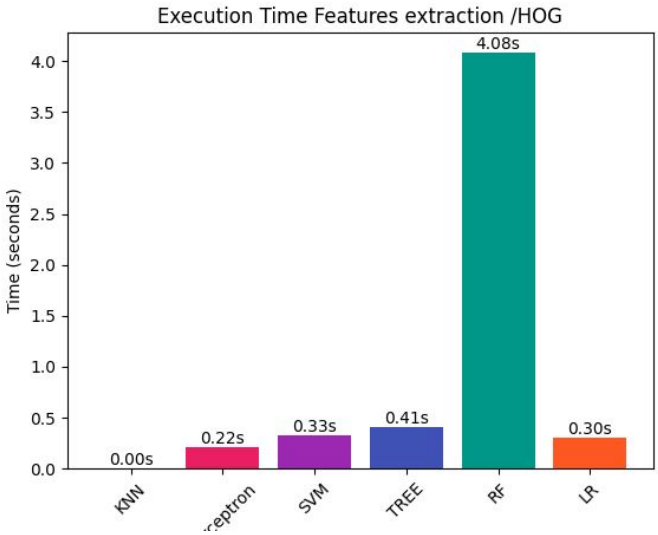

FIGURE 2.2 – Models execution time

2.1.2 Via HOG Grid Search

2.1.2.1 Accuaracy

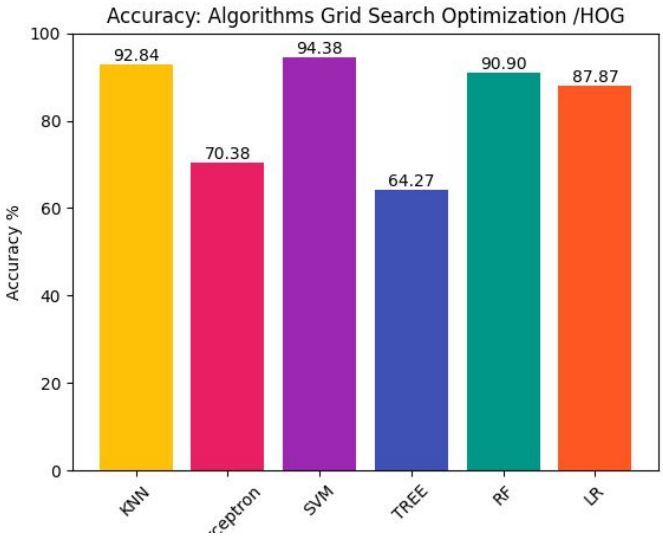

FIGURE 2.3 – Models accuaracy

2.1.2.2 Temps d'execution

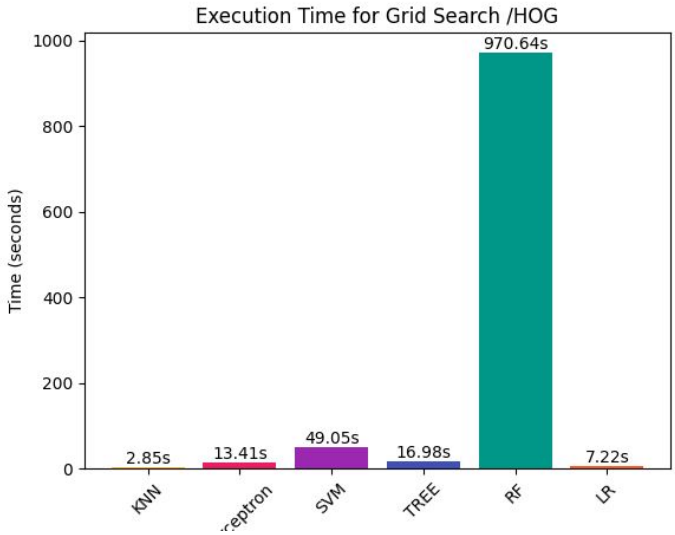

FIGURE 2.4 – Models execution time

2.1.3 Via HOG Random Search

2.1.3.1 Accuaracy

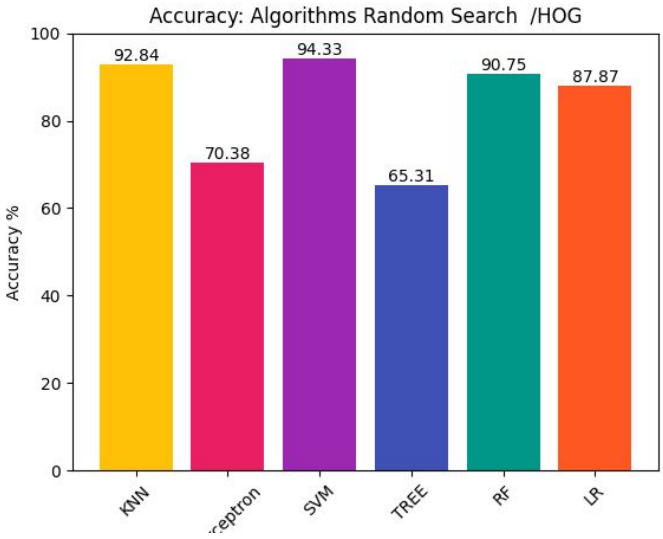

FIGURE 2.5 – Models accuaracy

2.1.3.2 Temps d'execution

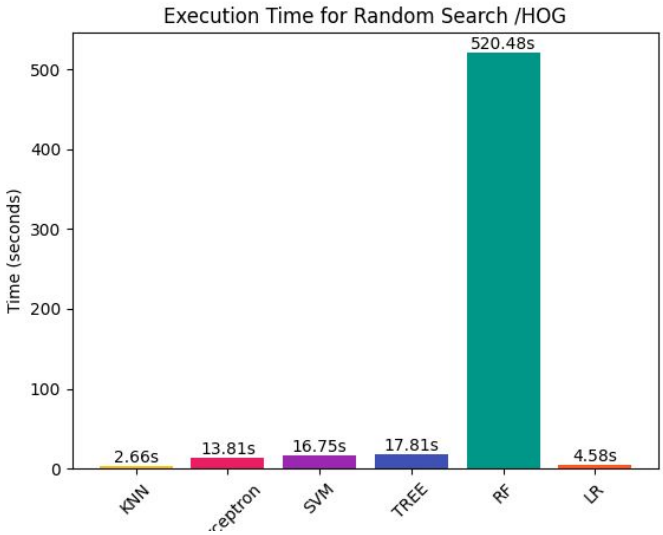

FIGURE 2.6 – Models execution time

2.1.4 Via LBP

2.1.4.1 Accuaracy

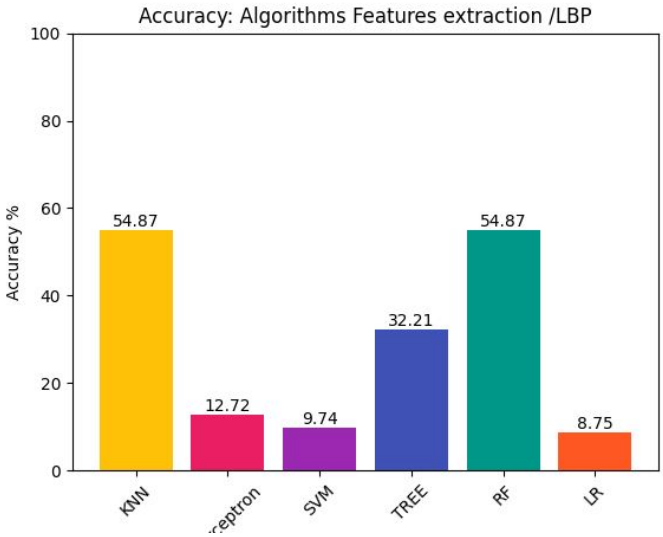

FIGURE 2.7 – Models accuaracy via LBP

2.1.4.2 Temps d'execution

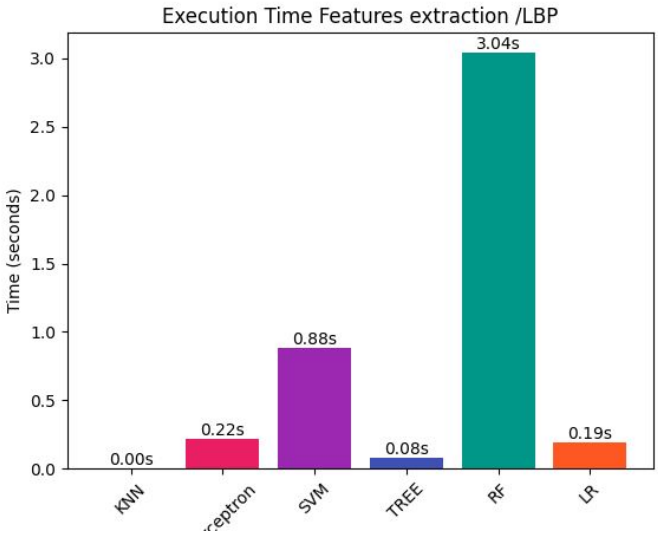

FIGURE 2.8 – Temps d'execution

2.1.5 Via LBP Grid Search

2.1.5.1 Accuracy

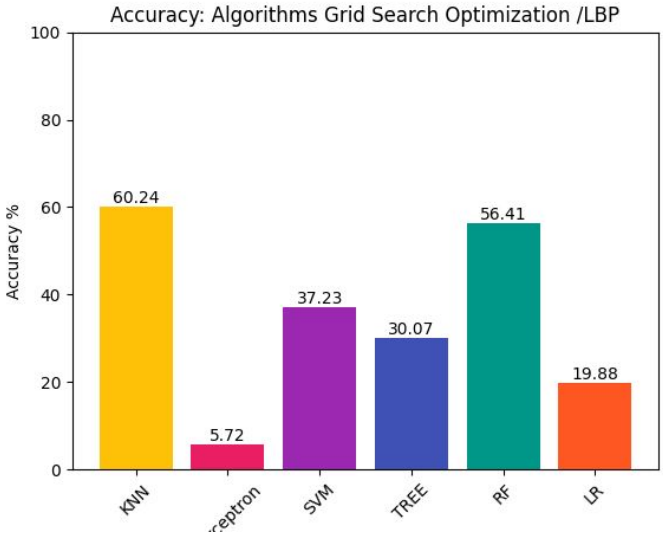

FIGURE 2.9 – Models accuaracy

2.1.6 Temps d'execution

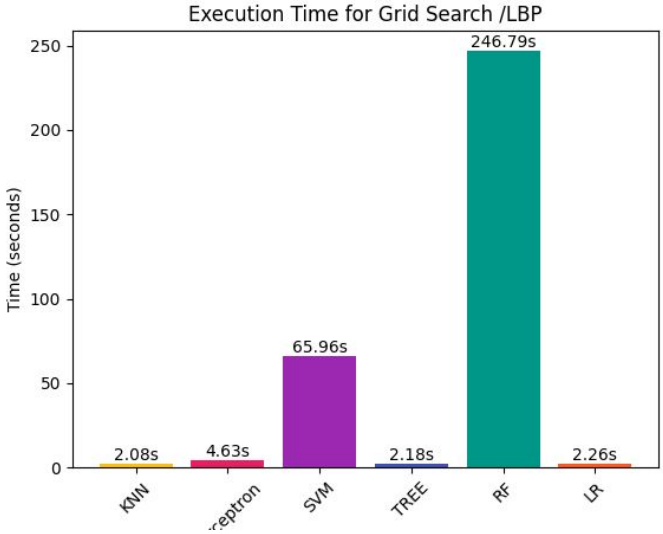

FIGURE 2.10 – Models execution time

2.1.7 Via LBP Random Search

2.1.7.1 Accuracy

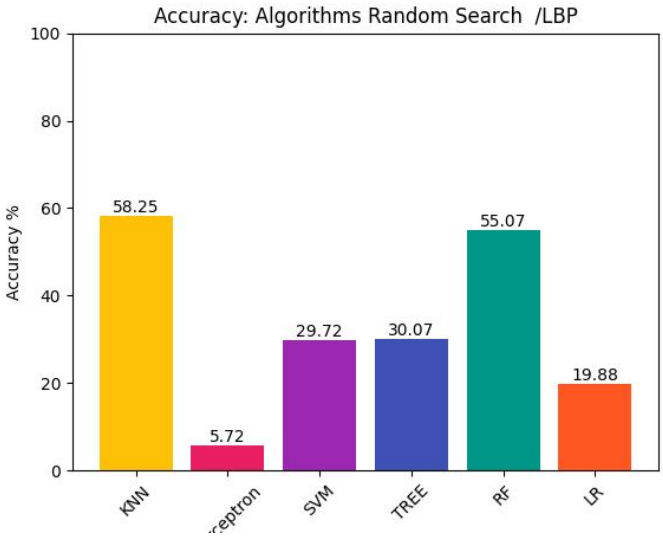

FIGURE 2.11 – Models accuaracy

2.1.8 Temps d'execution

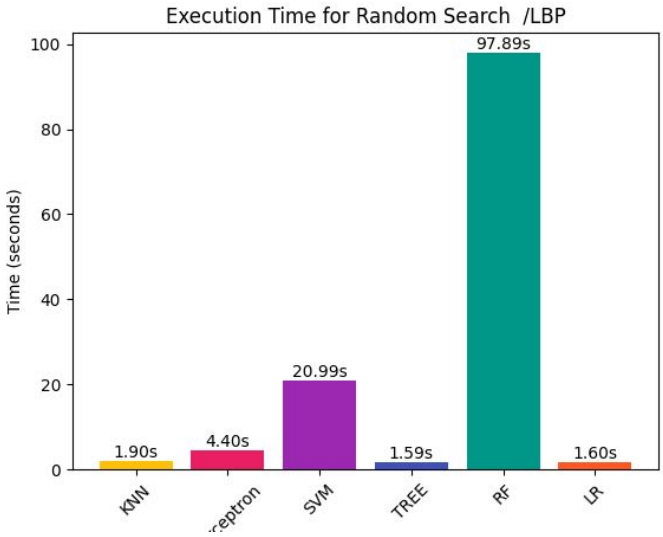

FIGURE 2.12 – Models execution time

2.2 Interface de l'application

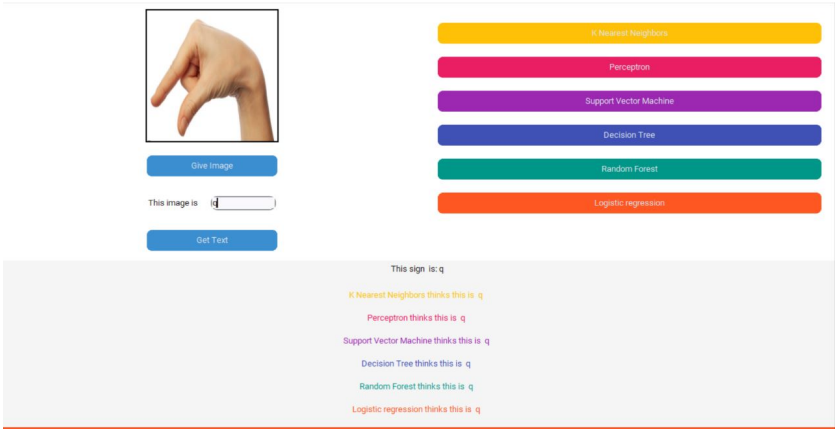

FIGURE 2.13 – interface de l'application

2.3 test de l'application

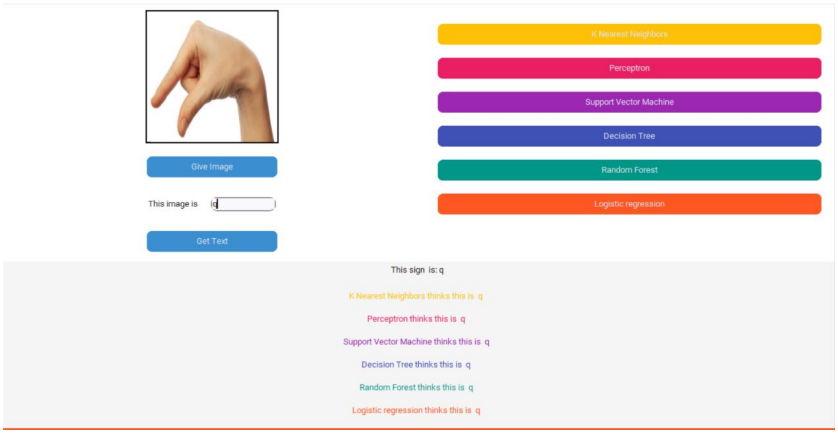

FIGURE 2.14 – test de l'application