UFR de Mathématiques et Informatique

Master 2ème année, 2018-2019, parcours MIAGE Apprentissage – Projet Pluridisciplinaire

SPÉCIFICATIONS TECHNIQUES

UNIVERSITÉ PARIS DESCARTES

Maître d'ouvrage	Séverine AFFELDT & Lazhar LABIOD
Version du document	1.0
Date du document	21/05/2018
Auteur(s)	Naomie FOURNIE Lara LAFORGE Malik CHAIBDOUR Alexandre BARBOSA Amine SANAE
Soumis le	18/06/2019
Type de diffusion	.PDF
Confidentialité	Réservé aux étudiants et encadrants UFR Maths-Info de l'université Paris Descartes

UNIVERSITÉ PARIS DESCARTES

SOMMAIRE

UNIVERSITÉ PARIS DESCARTES

I. Introduction

1) Contexte

Dans le cadre de notre formation par apprentissage en master MIAGE, nous avons l'opportunité de pouvoir gérer un projet dans son intégralité et ce en situation réelle. Plusieurs choix de sujets nous ont donc été soumis et nous avions choisi le sujet portant sur **le Machine Learning avec TensorFlow.js**.

TensorFlow est un outil open source d'apprentissage automatique développé par Google. Ce réseau permet notamment l'implémentation, l'entraînement et l'évaluation de réseaux de neurones profonds. Ce type de réseau est capable d'apprendre à priori n'importe quelle fonction.

Par exemple, imaginons que sur une photo il y a plusieurs choses dont un chat. Pour pouvoir identifier le chat sur la photo, l'algorithme doit être en mesure de distinguer le chat des autres éléments présents sur la photo, et de reconnaître un chat de manière précise quel que soit l'angle sous lequel il est photographié.

Afin d'y parvenir, le réseau de neurones doit être entraîné. Il est donc nécessaire de compiler un ensemble d'images d'entraînement pour pratiquer le Deep Learning. Cet ensemble va regrouper des milliers de photos de chats différents, mélangés avec des images d'objets qui ne sont pas des chats.

Pour ces images, ce sont les pixels qui seront insérés dans un réseau et sont ensuite convertis en données puis transférés sur le réseau. Les neurones artificiels assignent ensuite un poids (un certain nombre de paramètres ajustables) aux différents éléments.

La couche finale de neurones va alors rassembler les différentes informations pour déduire s'il s'agit ou non d'un chat.

Le réseau de neurones va ensuite comparer cette réponse aux bonnes réponses indiquées par les humains.

Dans le cas contraire, le réseau prend note de son erreur et ajuste le poids placé sur les différents neurones pour corriger son erreur. Le processus est répété sur chaque observation et le jeu de données peut être parcouru plusieurs fois (epoch) jusqu'à ce que le réseau soit capable de reconnaître un chat sur une photo dans toutes les circonstances.

TensorFlow est un système d'apprentissage automatique qui se présente comme une bibliothèque dédiée au calcul numérique.

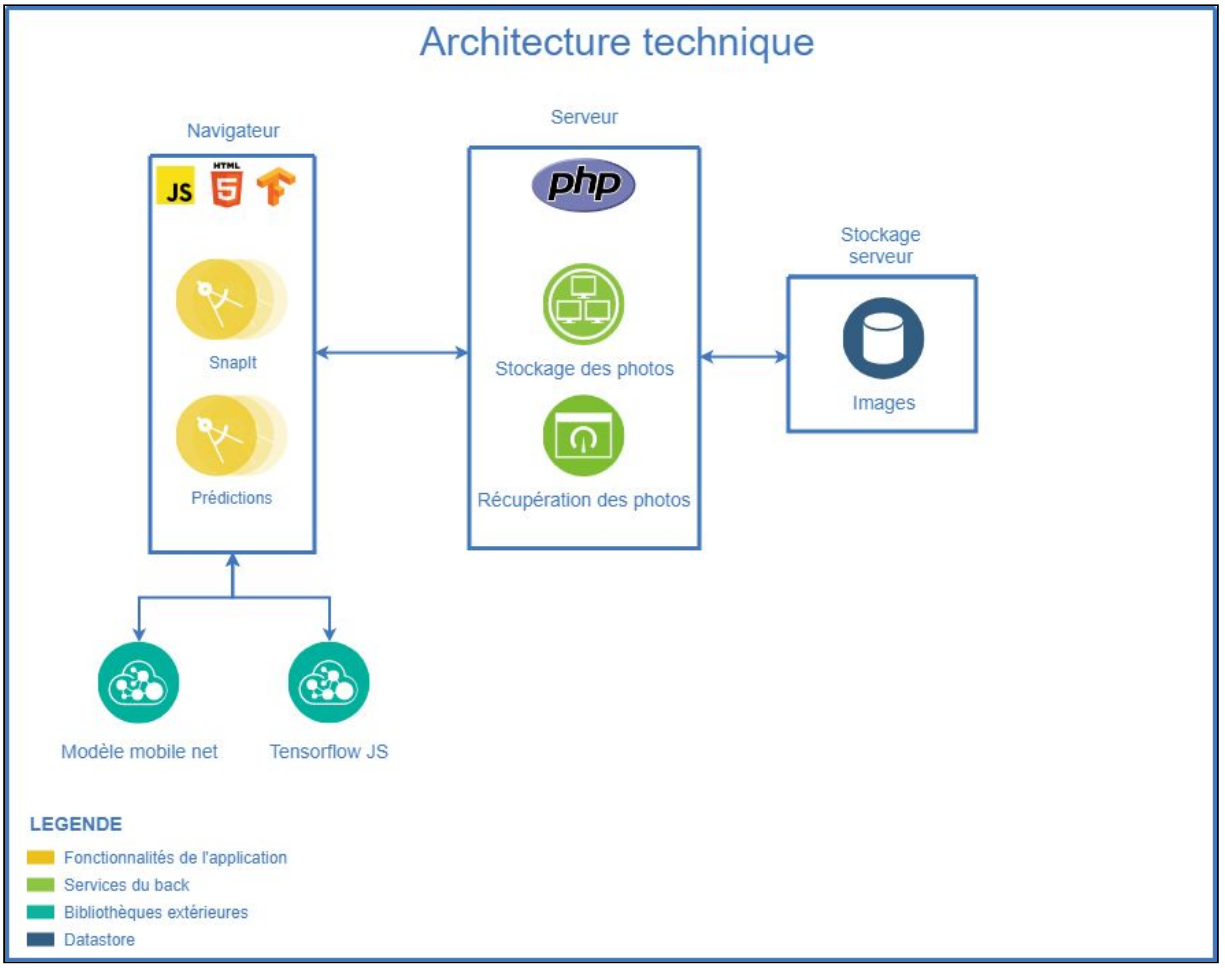
Initialement, le but de TensorFlow était d'optimiser les calculs numériques complexes, mais aujourd'hui il est très connu pour résoudre des problèmes de Deep Learning. Toutefois, ce

framework est assez général pour être utilisé à d'autres fins. Ce document présentera les objectifs que l'on devra atteindre avec le framework TensorFlow.

2) Synthèse de l'existant

Aucun existant. Le sujet a été proposé pour la première fois dans l'Université.

II. Architecture de l'application, technologies et outils utilisés



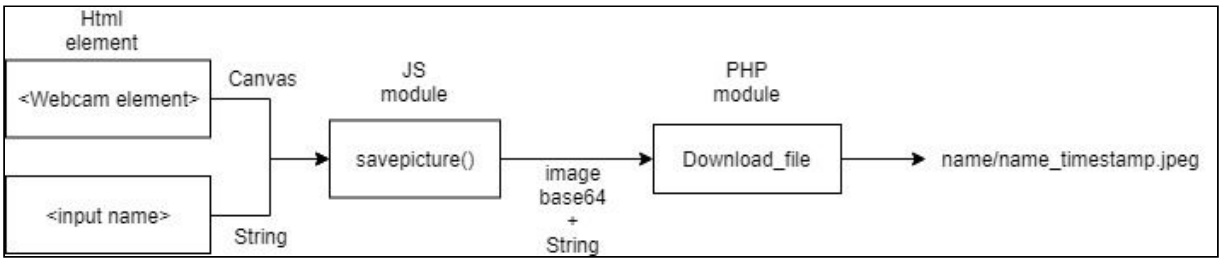
Notre application sera conçue selon une architecture **client serveur**.

- Du côté **serveur** nous utiliserons l'outils wamp pour simuler un serveur en local et du PHP pour coder les fonctionnalité du back-end.
- Du côté **base de données**, les images seront directement stockées dans la mémoire du serveur, sur les disques durs car nous avions aucune information à stocker, mis à part les photos de chaque utilisateur (pas de connexion, ni de données extérieures) et qu'une image est difficilement stockable sur une base.
- Du côté **front**, nous allons concevoir une interface ergonomique permettant d'utiliser les différentes fonctionnalités de façon intuitive à l'aide également d'un guide utilisateurs qui sera implanté dans l'interface. Les langues utilisées sont HTML, CSS, Javascript, Ajax, Tensorflow JS et Mobile Net.

III. Fonctionnalités

1) Fonctionnalités back-end

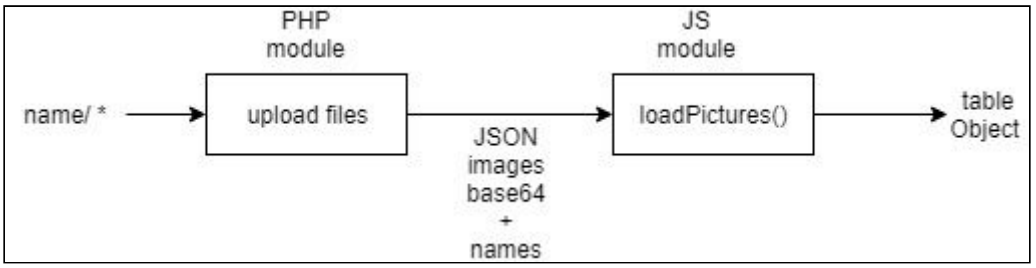
Récupération des images prises par la webcam, conversion en image base64 et téléchargement sur le serveur.



Le module PHP “save picture” permettra de stocker l’image reçu depuis la webcam sur le serveur dans le répertoire approprié.

En fonction du nom entrer dans l’interface pour ajouter une nouvelle personne la fonctionnalité PHP enregistrera la photo dans un nouveau répertoire ayant le nom de la personne ou met la nouvelle photo dans le répertoire existant si la personne existe déjà.

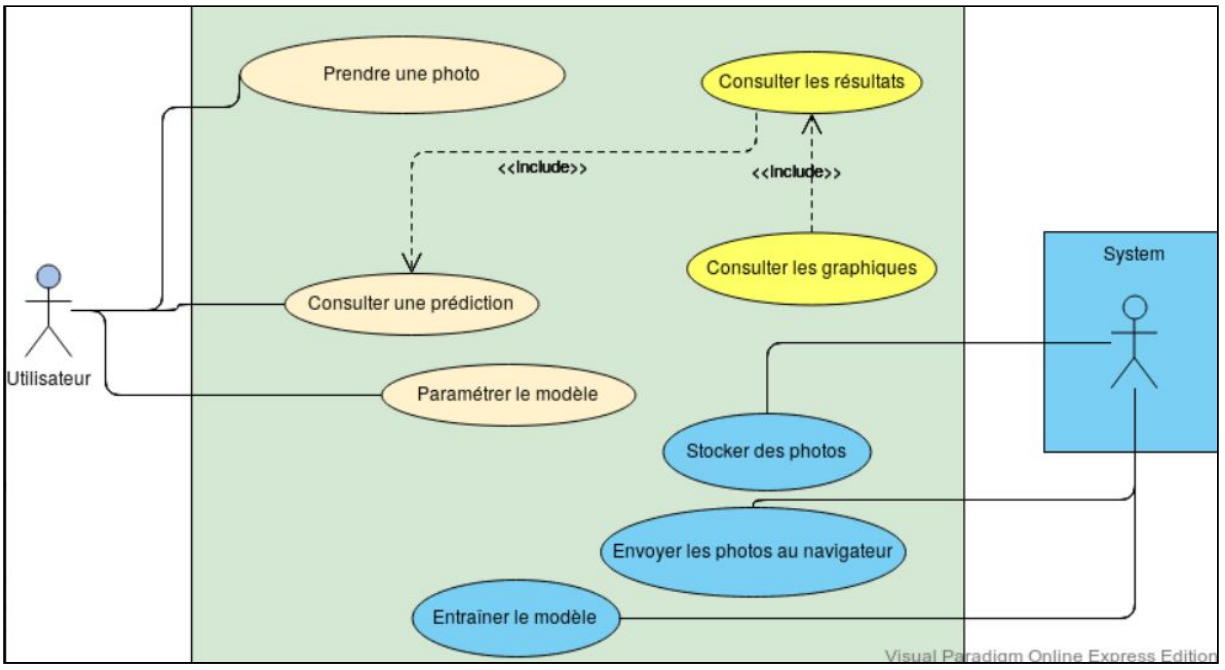
2) Récupérer les images sur le serveur afin de les réinjecter dans le modèle



Le module PHP “upload files” permettra d’uploader toutes les images par dossiers, un dossier représentant une classe, c’est à dire une personne.

Ces images seront ensuite transmises au module JS afin de les transformer en objet canvas exploitable par le modèle Mobile Net.

3) Fonctionnalités Front-End



Voici le processus nominal de l'application :

1. Prédiction de visage par la webcam
2. Entrainement du modèle en variant les hyperparamètres
3. Visualisation des performances du modèle
4. Ajout d'une personne dans le modèle

4) Prédiction de visage par la webcam

on utilisera mobilenet comme modèle pour injecter les différente classe à prédire.

A l'entrée de l'application les hyperparamètre du modèle sont fixé:

- Optimizer : adam

- Taux d'apprentissage : 1

- Batchsize : 32

- Epochs : 75

les hyperparamètres ont été fixé de cette manière car nous avons remarqué que cette configuration nous permettez d'obtenir un modèle assez performant et pas trop lent à la

prédiction. Nous laisserons la possibilité à l'utilisateur de tester la variation des hyperparamètres.

5) Entraînement du modèle en variant les hyperparamètres

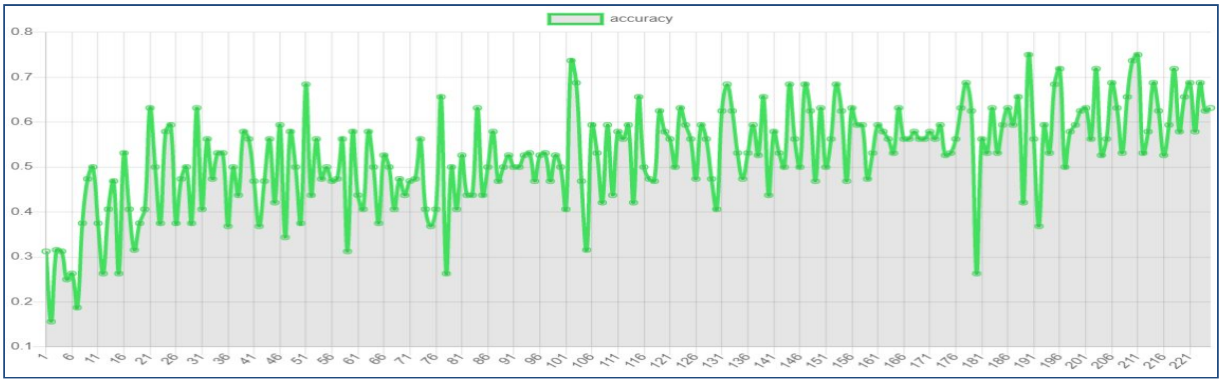
L'utilisateur pourra ré-entraîner le modèle en faisant varier différents paramètres :

- Epochs : Le nombre d'epochs représente le nombre de passages complets de l'ensemble du dataset d'apprentissage.
- Batch size : Le nombre d'échantillons d'entraînement à traiter avant la mise à jour des paramètres internes du modèle.
- Taux d'apprentissage : La vitesse d'apprentissage.
- Optimizer : L'optimizer permet d'initialiser la classe avec des paramètres donnés et sont utilisés pour améliorer la vitesse et la performance afin de former un modèle spécifique.

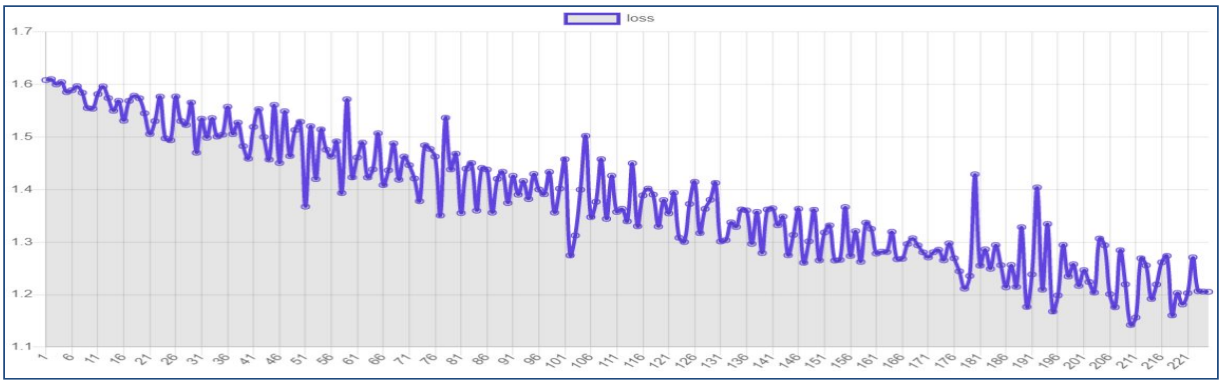
6) Visualisation des performances du modèle

Nous implémenterons dans l'interface la possibilité de visualiser les performances du modèle à l'aide de graphique représentant les fonctions "*loss*" et "*accuracy*" .

Accuracy :



Loss :



Nous aurons sur l'interface un tableau permettant de voir la comparaison entre le résultat réel et résultat prédit par le modèle (% d'erreur).

Il y aura la possibilité également de visualiser le jeu de données sous forme de miniature.

7) Ajout d'un personne

L'utilisateur arrive sur une interface qui lui permet de prendre un photo par sa webcam avec un input pour ajouter son prénom.

Ceci créera une nouvelle catégorie qui portera le prénom de la personne.

La photo est transmise au js par un objet canvas et est transformée en image en base64 et le prénom dans un input est récupéré dans le js en String.