

CAHIER DE RECETTES



Interface web - Machine Learning avec TensorFlow.JS



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



SOMMAIRE

I. Introduction	4
II. Description des scenarii	4
III. Scenarii	5
1. Taux d'apprentissage faible	5
2. Taux d'apprentissage élevée	10
3. Modification Epochs/BatchSize	16
4. Montée de charge	18

I. Introduction

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 avons 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.

II. Description des scénarii

Pour chaque scénario :

Identification

Nous avons donné un identifiant unique à chaque scénario.

Description

Nous avons décrit le but et le principe de réalisation du test ainsi que l'environnement de test.

Contraintes

Nous avons décrit les contraintes liées à ce scénario : environnement de test particulier, installation particulière, intervention humaine spécifique, etc ...

Dépendances

Nous avons listé et expliciter les tests à mener préalablement à la réalisation du scénario.

Procédure de test

Nous avons décrit les données en entrée, les résultats attendus, et les critères de validation.

III. Scenarii

1. Taux d'apprentissage faible

Identifiant	Description	Contraintes	Dépendances	Procédure de test
T1	Réaliser un entraînement du modèle avec un taux d'apprentissage faible	Besoin d'un environnement PHP pour faire fonctionner la sauvegarde et la récupération des images (WAMPserver)	Une base qui contient au moins 2 visages différents avec un nombre de photos raisonnable	taux d'apprentissage faible (0.001) 4 classes (visages) Résultats attendus : une <i>accuracy</i> et un <i>loss</i> qui ne progressent pas beaucoup

Ci-dessous, les paramètres pris en compte.



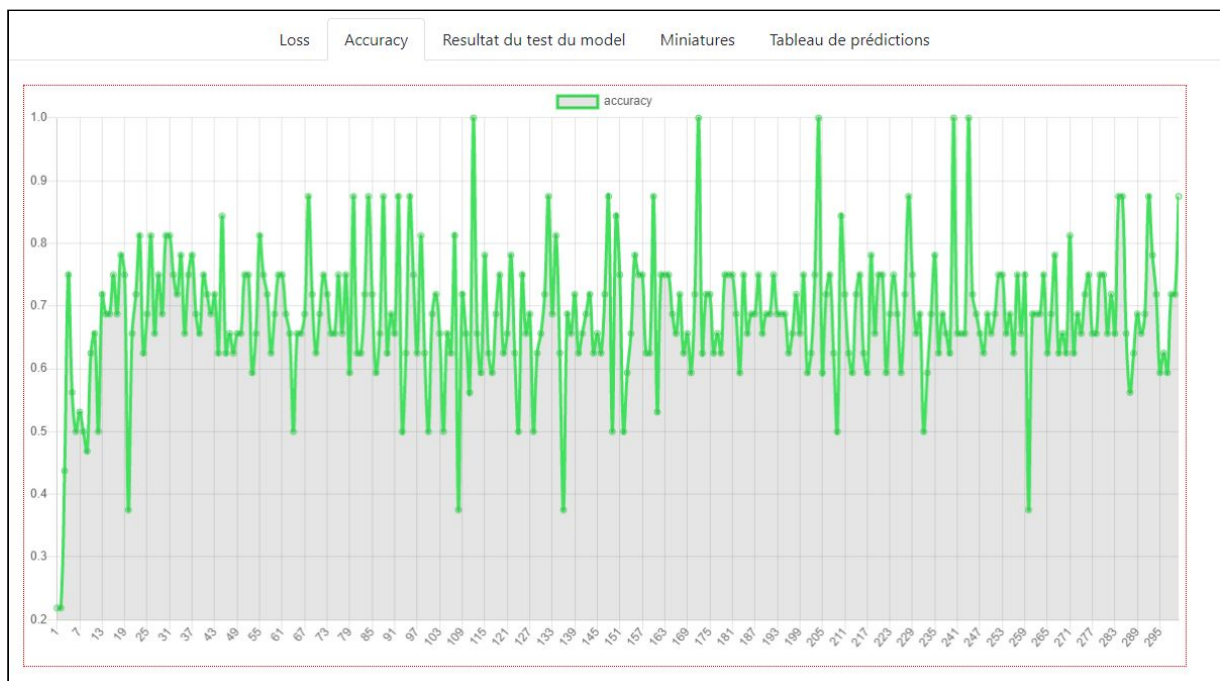
Optimizer
adam ▼

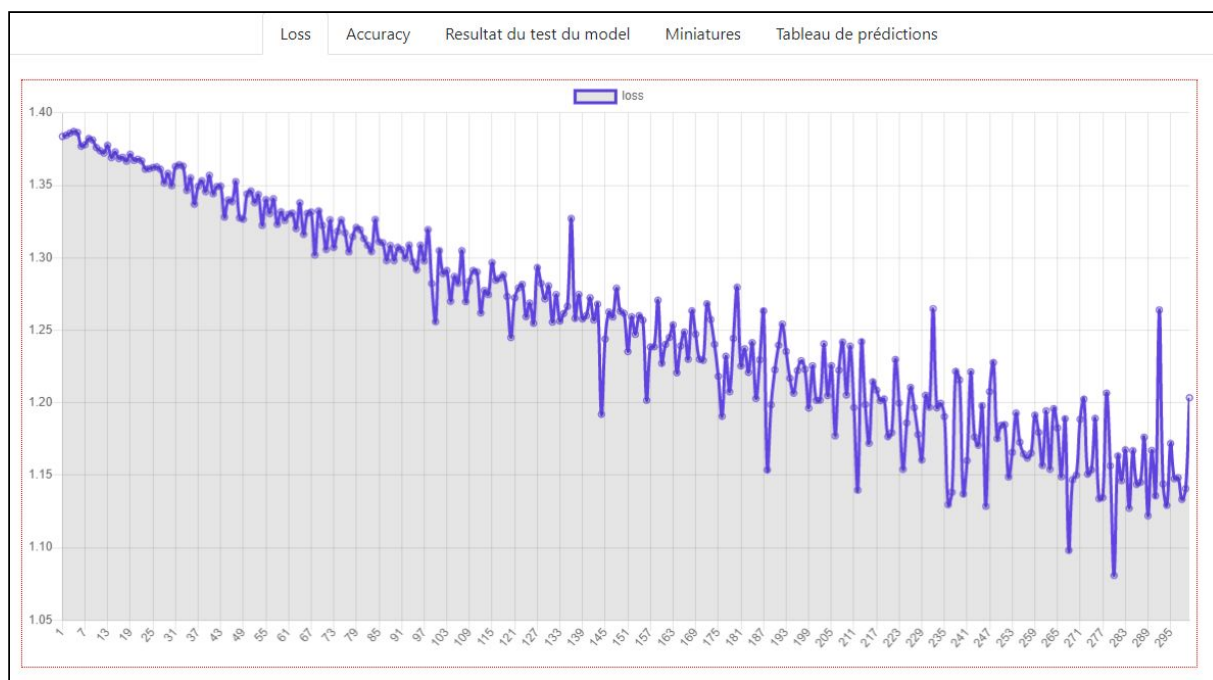
Taux d'apprentissage
0.001

BatchSize
32

Epochs
75

Le taux d'apprentissage est faible, le reste des paramètres sont basiques. En consultant l'*accuracy* et le *loss*, nous pouvons constater un modèle qui apprend peu et qui se trompe souvent. L'*accuracy* varie très fortement et le *loss* diminue mais faiblement.





Les résultats attendus et ceux prédits par le modèle nous montrent un taux d'erreur assez important.

Voici le tableau constaté sur notre application :

[illegible]

[illegible]



Amine	Amine	OK
Amine	Amine	OK
Amine	Amine	OK
Amine	Amine	OK
Amine	Amine	OK
Amine	Amine	OK
Amine	Amine	OK
Amine	Amine	OK
Amine	Amine	OK
Amine	Amine	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Alex	Erreur
Lara	Lara	OK
Lara	Alex	Erreur
Lara	Alex	Erreur
Lara	Alex	Erreur
Lara	Alex	Erreur
Lara	Alex	Erreur
Lara	Alex	Erreur
Lara	Alex	Erreur
Lara	Alex	Erreur
Lara	Alex	Erreur
Lara	Alex	Erreur
Naomie	Alex	Erreur
Naomie	Amine	Erreur
Naomie	Amine	Erreur

Naomie	Amine	Erreur
Naomie	Amine	Erreur
Naomie	Alex	Erreur
Naomie	Alex	Erreur
Naomie	Amine	Erreur
Naomie	Alex	Erreur
Naomie	Amine	Erreur
Naomie	Amine	Erreur
Naomie	Amine	Erreur
Naomie	Alex	Erreur
Naomie	Alex	Erreur
Naomie	Alex	Erreur
Naomie	Alex	Erreur
Naomie	Alex	Erreur
Naomie	Alex	Erreur

Pour conclure, lors des prédictions apportées par le modèle, nous pouvons constater qu'il confond les deux visages masculins du groupe, plus particulièrement lorsque l'on trompe intentionnellement le modèle en portant des lunettes.

Alex : 363

Amine : 196

Lara : 0

Naomie : 0

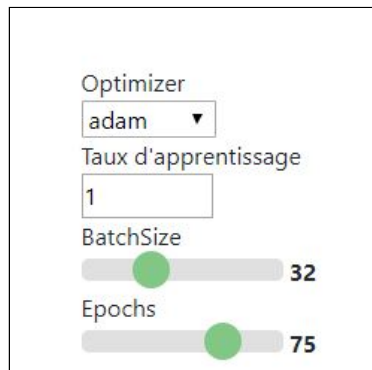
Le taux d'apprentissage est donc trop faible. Il existe un taux d'apprentissage idéal pour chaque problème de régression. Cette valeur dépend de la courbure de la fonction de perte.

Si vous savez que le gradient de la fonction de perte est faible, vous pouvez en toute sécurité essayer un taux d'apprentissage plus élevé, ce qui compense le gradient faible et entraîne un pas d'apprentissage plus grand.

2. Taux d'apprentissage élevée

Identifiant	Description	Contraintes	Dépendances	Procédure de test
T2	Réaliser un entraînement du modèle avec un taux d'apprentissage élevé	Besoin d'un environnement PHP pour faire fonctionner la sauvegarde et la récupération des images (WAMPserver)	Une base qui contient au moins 2 visages différents avec un nombre de photos raisonnable	<p>taux d'apprentissage élevé (1) 4 classes (visages)</p> <p>Résultats attendus : une <i>accuracy</i> et un <i>loss</i> qui progressent rapidement et arrivent à leurs extrêmes</p>

Ci-dessous, les paramètres prises en compte.



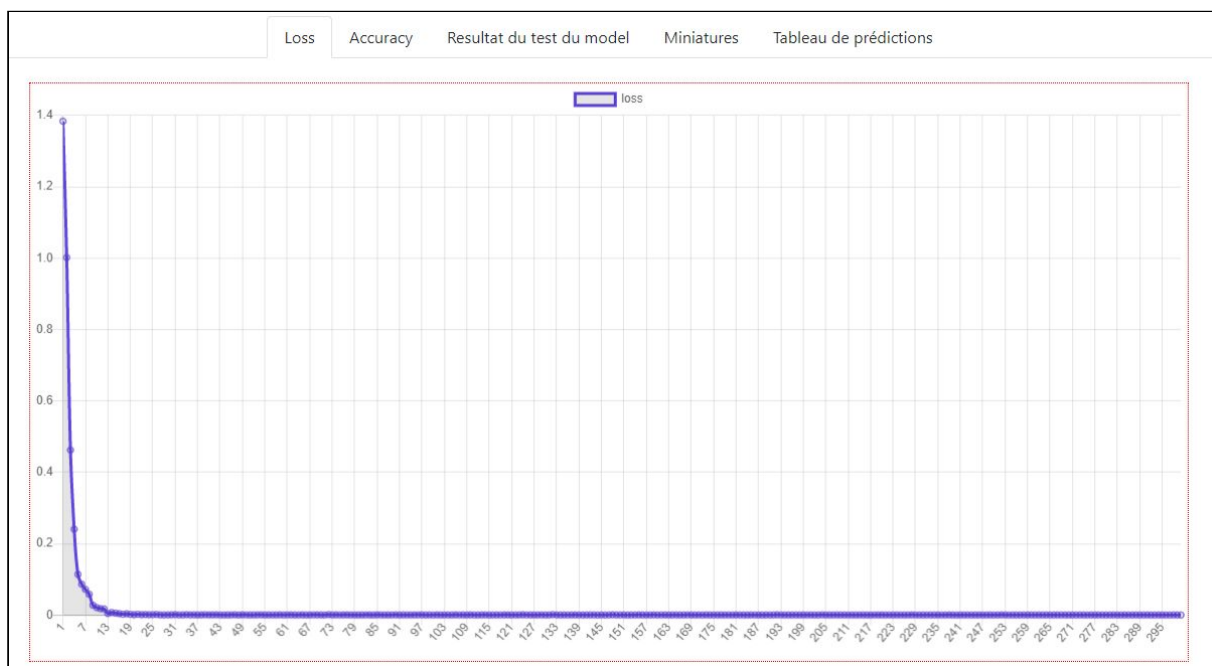
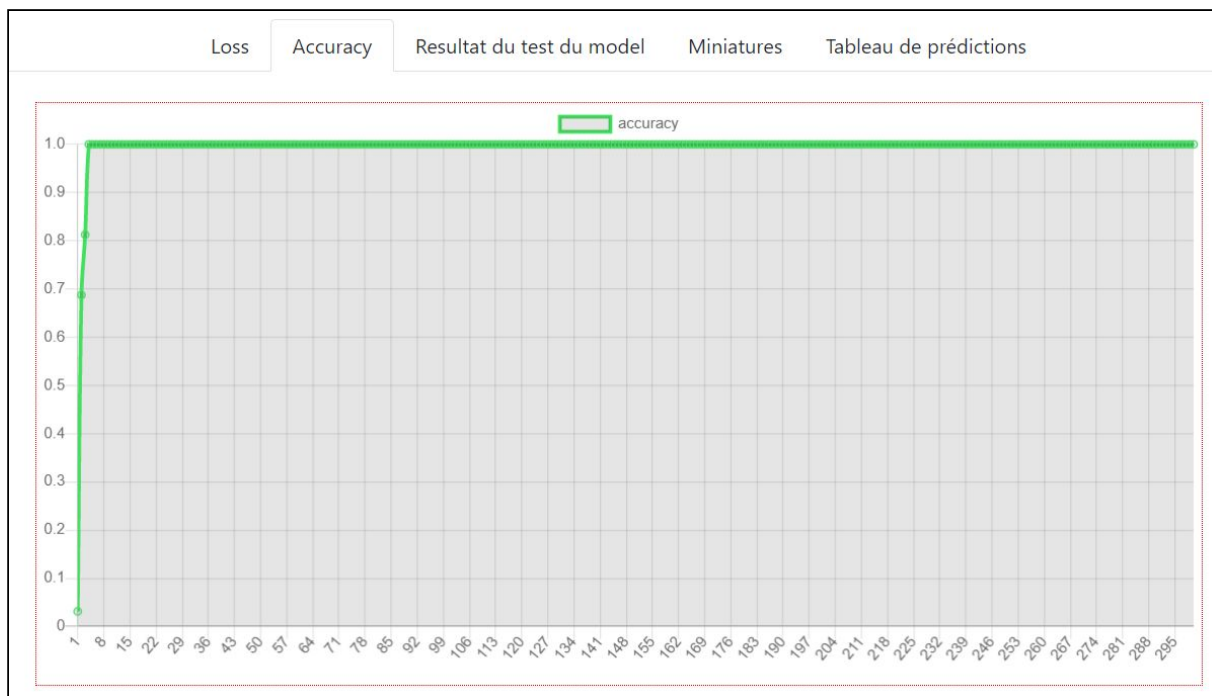
Optimizer
adam ▼

Taux d'apprentissage
1

BatchSize
32

Epochs
75

Le taux d'apprentissage est élevé, le reste des paramètres sont basiques. En consultant l'*accuracy* et le *loss*, nous pouvons constater un modèle qui apprend rapidement et arrive à son paroxysme en quelques secondes. Ce premier reste à la valeur de 1 et le *loss* diminue jusqu'à arrivé à 0.



Les résultats attendus et ceux prédits par le modèle nous montrent un taux d'erreur nul. Voici le tableau constaté sur notre application :

[illegible]

[illegible]

[illegible]



Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Lara	Lara	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK

Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK
Naomie	Naomie	OK

Pour conclure, lors des prédictions apportées par le modèle, nous pouvons constater qu'il confond moins les deux visages masculins du groupe. Il est d'ailleurs difficile de le tromper en portant des lunettes comme dans l'expérience précédente.



Alex : 610

Amine : 79

Lara : 31

Naomie : 73

Le taux d'apprentissage devient donc intéressant à la valeur de 1 mais pourrait être encore plus optimal avec une valeur légèrement plus faible (tout dépend du jeu de données fourni).

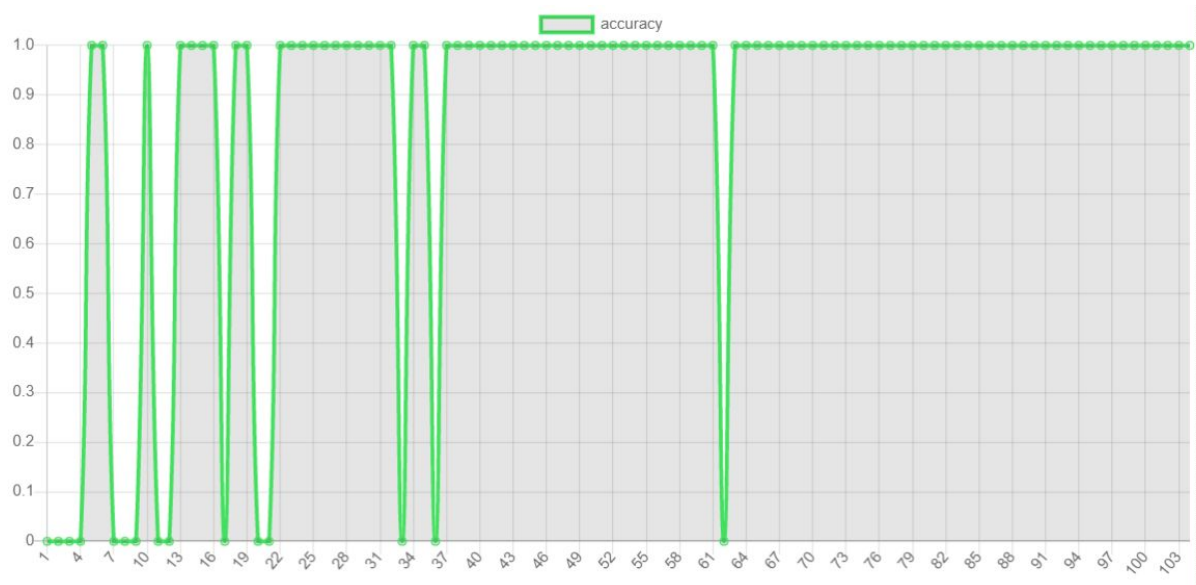
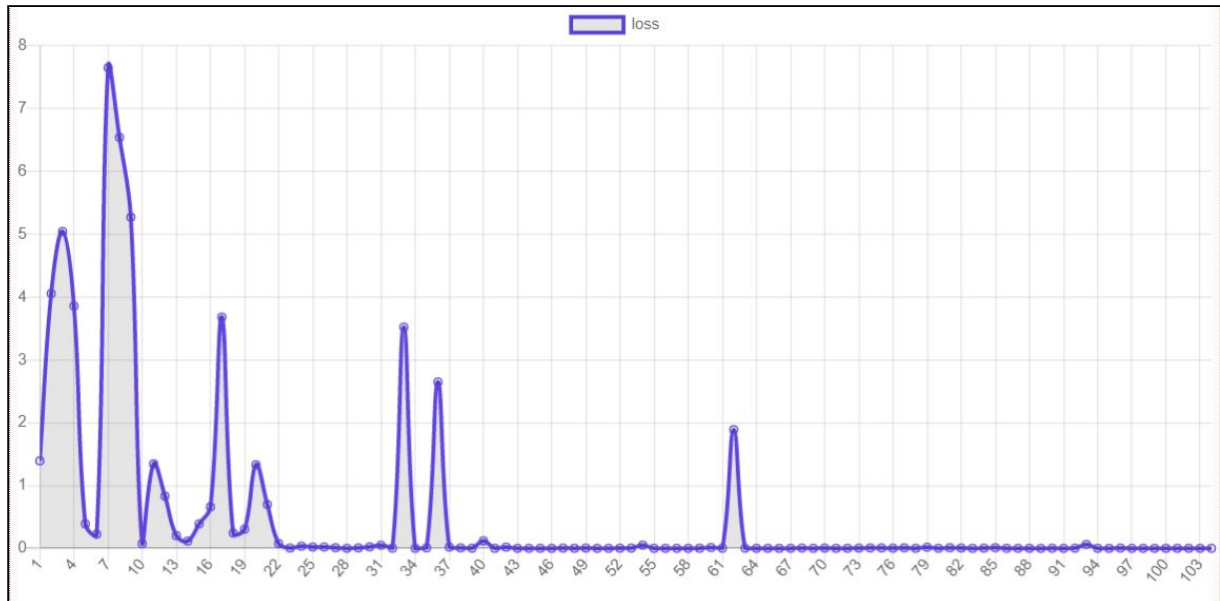
3. Modification Epochs/BatchSize

Identifiant	Description	Contraintes	Dépendances	Procédure de test
T3	Réaliser un entraînement du modèle en modifiant les epochs et la taille du batch (batch size)	Besoin d'un environnement PHP pour faire fonctionner la sauvegarde et la récupération des images (WAMPserver)	Une base qui contient au moins 2 visages différents avec un nombre de photos raisonnable	taux d'apprentissage élevé (1) 4 classes (visages) avec 103 photos en tout Résultats attendus : des graphiques qui montrent un travail plus ou moins important du modèle

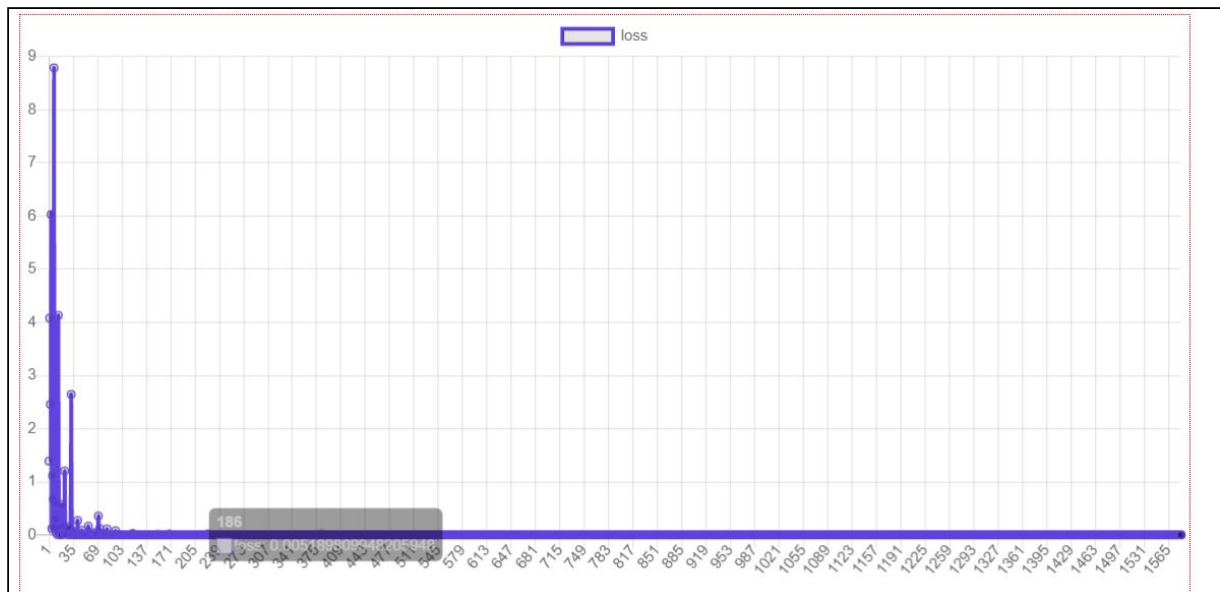
Lorsque l'on entraîne un ensemble de données, le nombre d'époques est le nombre de passages complets dans celui-ci. En comparaison, le *batch size* désigne un nombre d'échantillons traités avant la mise à jour du modèle.

Exemple : Supposons que vous avez un ensemble de données de 200 échantillons (lignes de données) et que vous choisissez une taille de lot de 5 et 1 000 époques. Cela signifie que l'ensemble de données sera divisé en 40 lots de cinq échantillons chacun. Les poids des modèles seront mis à jour après chaque lot de cinq échantillons. Cela signifie également qu'une époque impliquera 40 lots ou 40 mises à jour du modèle. Avec 1 000 époques, le modèle sera exposé ou parcourra l'ensemble des données 1 000 fois. Il s'agit d'un total de 40 000 lots pendant toute la durée de la formation.

Dans cet exemple, l'échantillon possède 103 photos. Si on fixe l'*epoch* à 1 et le *batch size* à 1, il est donc normal d'avoir un graphique contenant uniquement 103 valeurs différentes.



La perte et la précision nous apportent tout de même des valeurs finales intéressantes. Pour ce faible échantillon, 1 *epoch* et 1 *batch size* sont suffisants pour l'entraînement du modèle. En revanche, avec un plus gros échantillon, il est nécessaire de remonter au moins le nombre d'*epoch*. Avec 100 *epoch* et uniquement 1 lot par *epoch*, le nombre de valeurs augmentent drastiquement :



Plus de 1565 valeurs car nous avons interrompu le calcul.

4. Montée de charge

Identifiant	Description	Contraintes	Dépendances	Procédure de test
T4	Réaliser un entraînement du modèle avec 11 visages différents	Besoin d'un environnement PHP pour faire fonctionner la sauvegarde et la récupération des images (WAMPserver)	Une base qui contient au moins 11 visages différents	<p>taux d'apprentissage élevée (1) 11 classes (visages) avec 230 photos en tout</p> <p>Résultats attendus : le modèle est plus long pour s'entraîner. Le modèle continue à prédire les bons visages mais se trompe plus souvent</p>

En plaçant le visage "Alex" devant la webcam, la personne qui obtient le plus de détections est bien la classe "Alex" avec 909 correspondances. Mais le modèle se trompe plus que précédemment. On remarque également que la lumière est un facteur important.

Alex : 909

Amine : 167

Caroline : 7

Etienne : 380

Lara : 5

Mahery : 301

Malik : 358

Naomie : 72

Nazareth : 309

Vincent : 286

Wei : 149

Pour conclure, il faudrait réaliser un test complet avec des centaines de visages et avec 30 images par classes. Chose que nous pourrions pas réaliser actuellement par manque de participation et de temps d'exécution.