

Travail de lecture et de rédaction scientifique sur le  
Federate Learning

Bal Sébastien

3 juin 2021

## Remerciements

## TABLE DES MATIÈRES

<b>1</b>	<b>Machine Learning</b>	<b>1</b>
<b>2</b>	<b>Deep Learning</b>	<b>4</b>
2.1	Les concepts . . . . .	4
2.2	Modèle d'entraînement . . . . .	5
<b>3</b>	<b>Federated Learning</b>	<b>6</b>
3.1	Définition du Federated Learning . . . . .	6
3.2	Implémentation . . . . .	7
3.3	Deep Learning, cas d'utilisation . . . . .	7
3.3.1	Smart Building . . . . .	7
3.3.2	Industrie . . . . .	7
<b>A</b>	<b>Annexe</b>	<b>9</b>

## MACHINE LEARNING

Le Machine Learning appelé en Français apprentissage automatique [x : wikipédia] a pour objectif de traiter l'information afin de lui donner de la valeur ajoutée.

De nos jours, le Machine Learning est présent partout sur la toile, cela va du moteur de recherche comme Google, aux assistants vocaux comme Siri et Alexa, les fil d'actualités des réseaux sociaux comme Facebook et Twitter. Le point commun entre toutes ces plateformes et le stockage massif des données de leur utilisateur appelé Big Data [x : ?]. Le Big Data est une technologie apparue dans les années xxxx, a permis l'essor de l'apprentissage automatique, le Machine Learning. En effet, cet imposant volume de données collectées sur les utilisateurs a permis, dans les exemples cités ci dessus, de mieux cibler le comportement des utilisateurs et donc améliorer les expériences.

Pour fonctionner, le Machine Learning(ML) a besoin de données à ingérer et d'avoir un modèle appris afin de fournir des données à valeur ajoutée comme des algorithmes de prédiction de bourse et la maintenance prédictive. Il est donc nécessaire d'identifier les solutions pour créer ces modèles de ML.

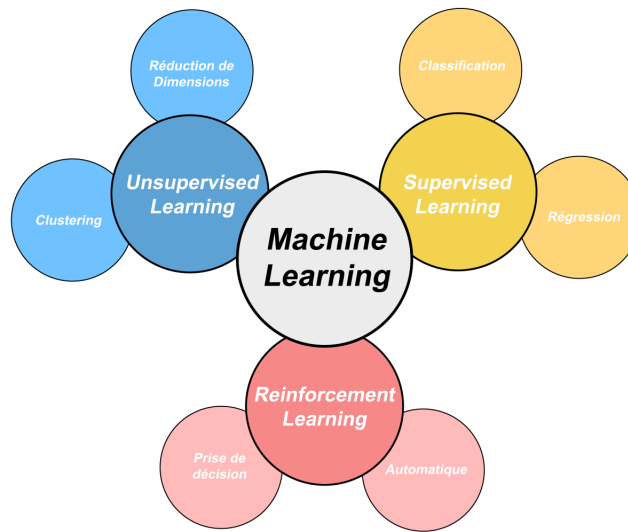


FIGURE 1.1: Familles d'algorithmes les plus utilisés

La première famille, l'apprentissage supervisé (en jaune dans la figure 1.1) consiste à donner des données en entrées, le résultat attendu itéré sur un grand jeu de données afin de trouver la modèle. Pour que le modèle deviennent performant, on fournit un grand volume de données dans le but qu'il se rapproche du modèle attendu.

La deuxième famille, l'apprentissage non-surpervisé (en bleu dans la figure 1.1) consiste à apprendre par reconnaître des ressemblances et des différences entre les données fournies. L'algorithme rassemble les données en groupe suivant ce qui lui semble le plus pertinent. Ainsi quand on passe un modèle bien précis à l'algorithme, il trouve plus facilement car il a été entraîné.

Pour finir, il existe une catégorie qui gère sa propre expérience. En effet, l'apprentissage par renforcement (en rouge dans la figure 1.1) consiste à générer ses propres expériences. On se rapproche de l'automobile autonome, la machine change ses états suivant les actions qu'elle entreprend de faire. Un système de récompense positive et négative est mis en place pour constituer une nouvelle expérience et rendre la machine attentive pour maximiser ses chances de réussite.

Pour conclure, le ML est une technologie qui vise à trouver des modèles, comprendre des comportements afin de prédire les besoins d'une application suivant un utilisateur.

On peut constater que ce type de besoin se focalise sur une application spécifique cependant, le ML connaît des limites au niveau des complexités combinatoire [x : ], or certaines applications nécessitent des applications plus complexes avec plus d'entrées, tels que le traitement des images. Pour palier aux limites du ML, le Deep Learning a vu son essor [x : lien vers essor DL]

## DEEP LEARNING

### 2.1 Les concepts

Le Deep Learning(DL) est représentatif d'un système neuronal comme notre système cérébral, il est conçu de plusieurs neurones qui interagissent entre eux. Pour rendre performant tous ces neurones, il est nécessaire d'avoir une grande quantité de données, un Data Lake. Celui ci fournit au système plusieurs informations afin de lui constituer une "mémoire". Cette mémoire lui permet de reconnaître des éléments bien particulier suivant l'entraînement qu'il aura suivi. Cet entraînement est supervisé par des développeurs qui vérifient que le DL ne sort pas de son modèles définis. Si il s'en éloigne, ils corrigent son algorithme mathématique pour le rendre performant et le remettre sur le droit chemin. Pour que le modèle mathématique deviennent performant, il faudra l'entraîner à reconnaître une donnée en particulier. C'est le sujet de notre prochaine section.

## 2.2 Modèle d'entraînement

Pour entraîner le DL, il lui faut un algorithme mathématique et un grand flux de données afin d'affiner ses recherches et prendre de l'expérience. Dans notre situation, on décide de prendre la reconnaissance d'une image, en particulier celle d'un chat. On fournit à l'algorithme un flux d'images de plusieurs espèces d'animaux, on définit les paramètres qui permettent d'affirmer que l'image que l'on soumet soit bien un chat. Ainsi avec ces critères, l'algorithme devient plus précis car on le guide un peu sur l'objectif qu'il doit atteindre.

Voici un schéma pour l'exemple du fonctionnement du DL, [fig2.1], les boules vertes représentent le bon chemin que le système va prendre pour arriver à vérifier le modèle qui était demandé. Les boules bleues sont celles qui ont des caractéristiques avec le modèle mais ne correspondra pas exactement au modèle qui était demandé. Les boules rouges quand à elles, représentent les erreurs que le système a exclu pour pouvoir apprendre le modèle exacte. Les erreurs sont par la suite renvoyées en amont du système pour que le système ajuste son modèle mathématique

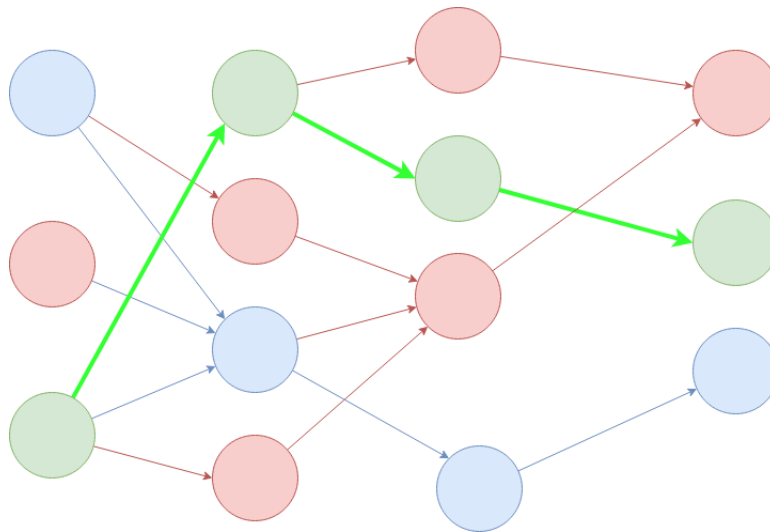


FIGURE 2.1: Autoapprentissage Deep Learning

On peut constater que ce type d'algorithme peut être très vite devenir énergivore. De plus, le DL se focalise principalement sur une application. Or, il peut être intéressant d'étudier plusieurs applications similaires afin d'identifier des modèles plus poussés. C'est dans ce contexte que nous allons introduire le Federated Learning.



## FEDERATED LEARNING

### 3.1 Définition du Federated Learning

Notre époque fait face à deux défis importants en terme d'avancer technologique : les respects des données des utilisateurs et l'augmentation du volume des données.

L'une des plus importantes est celle de la privatisation de ces données sur le web. Avec la protection des données (RGPD) promulgué en 2018, les données privées font entièrement partie de l'utilisateur. Ces données ne peuvent pas être utilisé sans son accord.

Ensuite le silo des données met un frein à l'évolution de l'industrie moderne, c'est en ayant plus de données que l'on peut améliorer la qualité de la formation. Cependant le manque de données valides dans le domaine médical vient principalement du faite que les travailleurs doivent être expérimentés dans certains domaines comme celui de l'industrie médicale.

C'est grâce au Federated Learning (FL) que l'essor de l'industrie a relevé ces défis. Car le FL est un processus d'apprentissage automatique qui vise à exploité les îlots de données tout en gardant la sécurité des données.

En quelques mots, c'est un apprentissage automatique distribué qui permet d'entraîner un modèle mathématique avec un large groupe de données décentralisées qui se trouvent sur des téléphones portables (pour notre cas ici).

## **3.2 Implémentation**

Dans notre modèle, nous allons utiliser le système TensorFlow pour former notre système neuronal. TensorFlow est une bibliothèque open source pour le Machine Learning. C'est un petit couteau Suisse qui contient ici des outils pour permettre de résoudre des problèmes mathématiques.

## **3.3 Deep Learning, cas d'utilisation**

### **3.3.1 Smart Building**

### **3.3.2 Industrie**

## BIBLIOGRAPHIE

- [1] Leslie Lamport, *LaTeX : A Document Preparation System*. Addison Wesley, Massachusetts, 2nd Edition, 1994.
- [2] Pour la partie sur le Machine Learning  
[https ://www.lebigdata.fr/machine-learning-et-big-data](https://www.lebigdata.fr/machine-learning-et-big-data)

---

ANNEXE

**A**

---

ANNEXE

---

# TOWARDS FEDERATED LEARNING AT SCALE: SYSTEM DESIGN

---

Keith Bonawitz<sup>1</sup> Hubert Eichner<sup>1</sup> Wolfgang Grieskamp<sup>1</sup> Dmitry Huba<sup>1</sup> Alex Ingerman<sup>1</sup> Vladimir Ivanov<sup>1</sup>  
 Chloé Kiddon<sup>1</sup> Jakub Konečný<sup>1</sup> Stefano Mazzocchi<sup>1</sup> H. Brendan McMahan<sup>1</sup> Timon Van Overveldt<sup>1</sup>  
 David Petrou<sup>1</sup> Daniel Ramage<sup>1</sup> Jason Roselander<sup>1</sup>

## ABSTRACT

Federated Learning is a distributed machine learning approach which enables model training on a large corpus of decentralized data. We have built a scalable production system for Federated Learning in the domain of mobile devices, based on TensorFlow. In this paper, we describe the resulting high-level design, sketch some of the challenges and their solutions, and touch upon the open problems and future directions.

## 1 INTRODUCTION

Federated Learning (FL) (McMahan et al., 2017) is a distributed machine learning approach which enables training on a large corpus of decentralized data residing on devices like mobile phones. FL is one instance of the more general approach of “bringing the code to the data, instead of the data to the code” and addresses the fundamental problems of privacy, ownership, and locality of data. The general description of FL has been given by McMahan & Ramage (2017), and its theory has been explored in Konečný et al. (2016a); McMahan et al. (2017; 2018).

A basic design decision for a Federated Learning infrastructure is whether to focus on asynchronous or synchronous training algorithms. While much successful work on deep learning has used asynchronous training, e.g., Dean et al. (2012), recently there has been a consistent trend towards synchronous large batch training, even in the data center (Goyal et al., 2017; Smith et al., 2018). The Federated Averaging algorithm of McMahan et al. (2017) takes a similar approach. Further, several approaches to enhancing privacy guarantees for FL, including differential privacy (McMahan et al., 2018) and Secure Aggregation (Bonawitz et al., 2017), essentially require some notion of synchronization on a fixed set of devices, so that the server side of the learning algorithm only consumes a simple aggregate of the updates from many users. For all these reasons, we chose to focus on support for synchronous rounds, while mitigating potential synchronization overhead via several techniques we describe subsequently. Our system is thus amenable to running large-batch SGD-style algorithms as well as Feder-

ated Averaging, the primary algorithm we run in production; pseudo-code is given in Appendix B for completeness.

In this paper, we report on a system design for such algorithms in the domain of mobile phones (Android). This work is still in an early stage, and we do not have all problems solved, nor are we able to give a comprehensive discussion of all required components. Rather, we attempt to sketch the major components of the system, describe the challenges, and identify the open issues, in the hope that this will be useful to spark further systems research.

Our system enables one to train a deep neural network, using TensorFlow (Abadi et al., 2016), on data stored on the phone which will never leave the device. The weights are combined in the cloud with Federated Averaging, constructing a global model which is pushed back to phones for inference. An implementation of Secure Aggregation (Bonawitz et al., 2017) ensures that on a global level individual updates from phones are uninspectable. The system has been applied in large scale applications, for instance in the realm of a phone keyboard.

Our work addresses numerous practical issues: device availability that correlates with the local data distribution in complex ways (e.g., time zone dependency); unreliable device connectivity and interrupted execution; orchestration of lock-step execution across devices with varying availability; and limited device storage and compute resources. These issues are addressed at the communication protocol, device, and server levels. We have reached a state of maturity sufficient to deploy the system in production and solve applied learning problems over tens of millions of real-world devices; we anticipate uses where the number of devices reaches billions.

---

<sup>1</sup>Google Inc., Mountain View, CA, USA. Correspondence to: Wolfgang Grieskamp <wgg@google.com>, Vladimir Ivanov <vlivan@google.com>, Brendan McMahan <mcmahan@google.com>.