

# M2 : Industry 4.0

Gilles Menez

UniCA : Université de Nice – Côte d'Azur  
Miage  
email : menez@unice.fr  
www : www.i3s.unice.fr/~menez

March 17, 2025: V 1.0

# Table des Matières (1)

Cloud/Fog/Edge .....	3
Le contexte du FL .....	3
Edge computing .....	6
Exemples d'utilisations .....	8
Usines .....	9
Véhicules connectés .....	11
Véhicules autonomes .....	14
CDN : Content Delivery Network .....	15
CDN : Netflix 2016 .....	16
Avantages du Edge Computing .....	19
Défis liés à l'Edge Computing .....	22
 Federated Learning .....	23
Comment ça marche ? .....	24
Utilité ? RGPD ! .....	26
Apprentissage collaboratif .....	28
 Stratégies .....	32
Quelques références .....	32
Ce qui les différencie .....	33
HFL : Horizontal Federated Learning .....	34
VFL : Vertical Federated Learning .....	36
Apprentissage fédéré centralisé .....	38
Apprentissage fédéré décentralisé .....	39
Apprentissage fédéré hétérogène .....	40
 Quelques algorithmes .....	41
Descente stochastique du gradient fédérée (FedSGD) .....	41
FedAvg : Federated Averaging Algorithm .....	42
Federated Optimization .....	43
FedAvg : Central makes its weights from edges .....	45
FedAvg : Each edge computes its contribution .....	46
 Some challenges in federated learning .....	48
MNIST Use Case .....	49
FL : TODO .....	49

## Le contexte du FL

L'intérêt pour le "**Federated Learning**" provient de **la synergie de plusieurs facteurs** dans des domaines scientifiques :

- ▶ Les "réseaux de neurones" : des modèles de calcul qui implémentent un "machine learning" qui donne de bons résultats ... même si on ne sait pas trop pourquoi ;-)

Ces modèles supervisés nécessitent un entraînement et donc souvent "énormément de données" d'apprentissage !

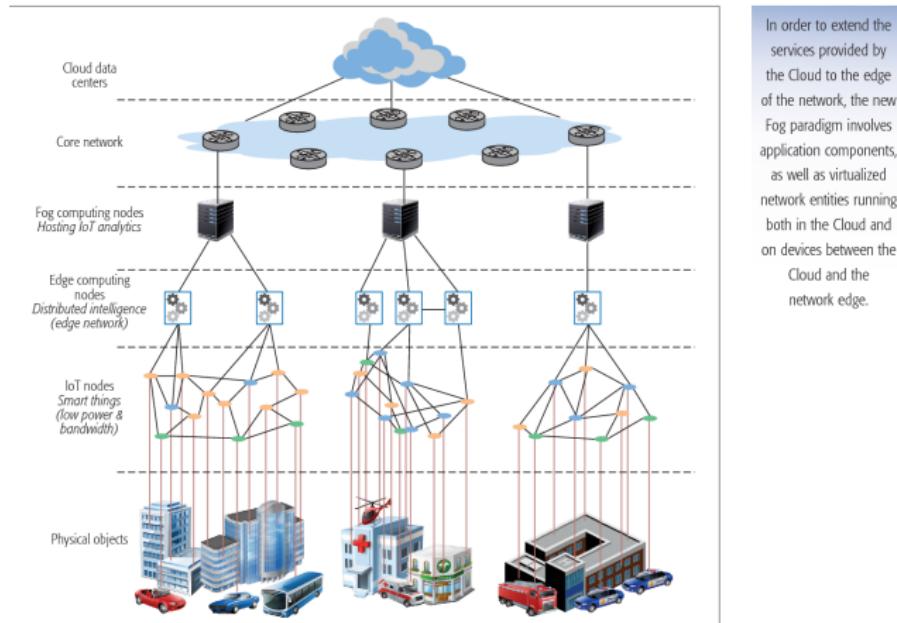
- ▶ L'"Internet des Objets" ... des réseaux d'objets qui vont chercher les données au plus proche du monde réel.

Données qui peuvent être aussi légère qu'un scalaire (une température par exemple) ou bien tout aussi bien lourde qu'une image 4K en vraie couleur.

- ▶ La prise en compte de réalités : économiques, juridiques, environnementales, ...

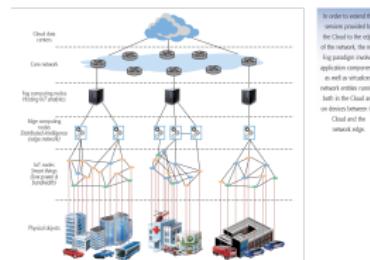
# Cloud/Fog/Edge

La "quête des données du monde réel" a modifié la structure des réseaux leur donnant une "**profondeur**" avec différentes couches : cloud/fog/edge



Les couches des réseaux ubiquitaires de l'IoT : <https://api.semanticscholar.org/CorpusID:39775690>

"Logiquement", la démarche raisonnée de développement d'une application doit essayer d'optimiser la présence des éléments (au moins matériels) qui vont supporter son exécution.



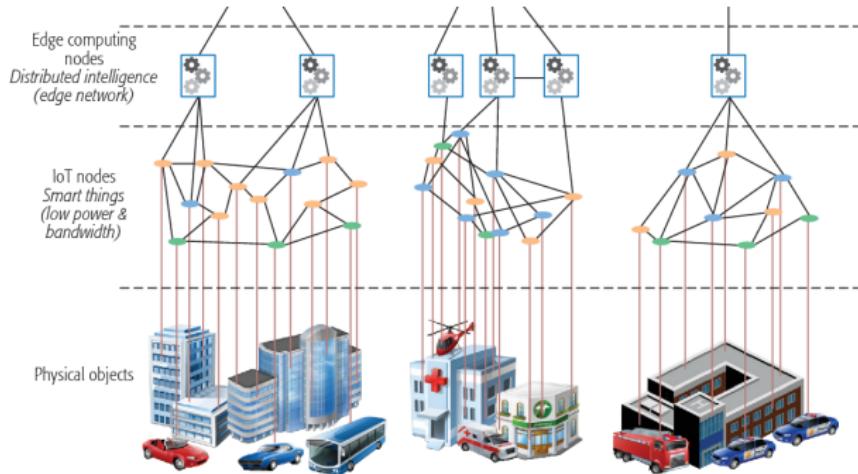
D'où, deux remarques contextualisent la démarche du Federated Learning :

1. Evitez de laisser du silicium inactif !
2. Evitez de communiquer inutilement ! ... surtout compte tenu du volume de données en question.

FL : Une exploitations raisonnées des réseaux et des puissances de calcul distribuées sur ces réseaux ... mais pas que ca !

# Edge computing

L'edge computing désigne le traitement informatique qui s'effectue à **l'emplacement physique, ou à proximité**, de l'utilisateur (consommateur) ou de la source des données (producteur).



En rapprochant les services de calcul de ces emplacements, l'**edge computing** permet aux utilisateurs de profiter de services plus rapides et fiables.

De nombreux cas d'utilisation de l'edge computing sont liés à une **démarche de recherche de performances** :

- ▶ La nécessité de traiter les données localement en temps réel, lorsque la transmission des données vers un datacenter en vue de leur traitement génère des niveaux de latence inacceptables.

Pour les entreprises, on peut considérer qu'une stratégie d'edge computing étend un environnement cloud à de nombreux autres emplacements.

- ▶ Cette démarche offre ainsi la flexibilité d'un "cloud hybride" qui permet d'exécuter les mêmes charges de travail à la fois dans leurs propres datacenters et sur une infrastructure de cloud public (telle que AWS, Azure ou Google Cloud),

\*Le terme "cloud hybride" désigne une combinaison d'au moins 2 environnements de cloud computing qui échangent des informations entre eux et exécutent une série uniforme d'applications pour le compte d'une entreprise

# Exemples d'utilisations

L'edge computing est désormais utilisé dans de nombreux secteurs d'activités.

- ▶ On voit cela ci-après ...

## Usines

Prenons l'exemple d'une usine de fabrication "moderne", donc sans doute robotisée et donc sans doute équipée "IoT" :



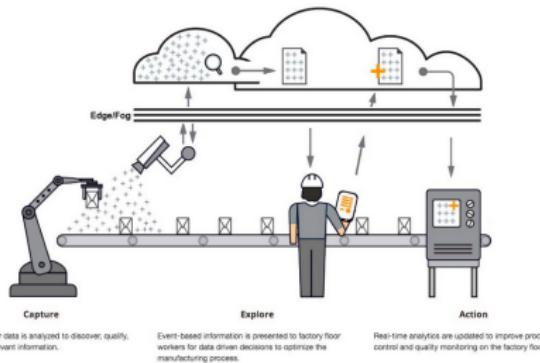
Les capteurs génèrent un flux continu de données qui peuvent être utilisées pour éviter des interruptions et améliorer l'exploitation.

- ▶ D'après les estimations, **une usine moderne qui compte 2 000 équipements peut générer 2 200 Téraoctets de données par mois.**

Essayer de traiter ces données à proximité de l'équipement est plus rapide et moins onéreux que de les transmettre "systématiquement" à un datacenter (distant).

Mais il ne s'agit pas d'effacer complètement le centre du réseau :

- ▶ Il reste judicieux d'utiliser une plateforme de données centralisée pour faire le lien avec l'équipement.



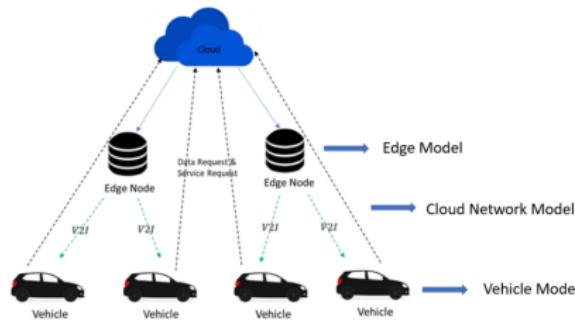
La notion de "Edge" s'entend donc **avec l'existence d'un centre de réseau** :

- ▶ De cette façon, l'équipement peut, par exemple, recevoir des mises à jour logicielles standardisées et **partager des données filtrées qui permettront d'améliorer l'exploitation dans d'autres sites de l'usine.**

# Véhicules connectés

Autre "use vase", le transport est de plus en plus organisé en "flotte de véhicules"(/fleet) :

- ▶ Les bus et les trains sont équipés d'ordinateurs qui permettent de suivre le flux de passagers et la prestation de services.
- ▶ Les livreurs peuvent trouver les trajets les plus rapides grâce à la technologie embarquée dans leur véhicule.



Lorsqu'une stratégie d'edge computing est mise en œuvre, **chaque véhicule exécute la même plateforme standardisée que le reste de la flotte**, ce qui rend les services plus fiables et permet de protéger les données de manière uniforme.

## Une flotte de deux voitures ;-)

McLaren développe une stratégie de véhicules connectés axée sur les capteurs IoT, l'Edge Computing et la simulation afin de **doter plus rapidement ses véhicules de fonctionnalités innovantes.**



*"Non seulement nous utilisons cette connectivité pour changer nos processus et le rythme de développement de nos produits, mais nous l'utilisons aussi pour assurer l'exécution de ces fonctions en optimisant la performance pour se rapprocher le plus possible du temps réel"* par J.Neale, COO McLaren Group.

"Pour vraiment réussir une belle course un week-end de Grand Prix, nous avons besoin d'exécuter l'information en temps réel sur des modèles, à la périphérie du réseau, donc nos besoins en edge computing sont de plus en plus importants et nous avons besoin des données des ingénieurs"

[https:](https://www.lemondeinformatique.fr/actualites/lire-mclaren-veut-accelerer-sur-l-iot-et-l-edge-computing-77026.html)

[//www.lemondeinformatique.fr/actualites/lire-mclaren-veut-accelerer-sur-l-iot-et-l-edge-computing-77026.html](https://www.lemondeinformatique.fr/actualites/lire-mclaren-veut-accelerer-sur-l-iot-et-l-edge-computing-77026.html)

L'objectif est de **concentrer le traitement des données** à la périphérie du réseau et donc **plus près du véhicule**.

- ▶ Cette proximité offrirait ainsi un meilleur accès aux données en temps réel, sans avoir à attendre leur traitement centralisé et leur retransmission aux utilisateurs finaux

## Véhicules autonomes

Viennent ensuite les véhicules autonomes, un autre exemple d'edge computing qui implique **le traitement d'un gros volume de données en temps réel dans une situation où la connexion au réseau peut être instable.**



- ▶ En raison de ce volume, les véhicules autonomes, tels que les voitures sans conducteur, traitent les données des capteurs à bord du véhicule afin de **réduire la latence**.
- ▶ Ils peuvent cependant toujours se connecter à distance à un point central pour recevoir des mises à jour logicielles.

# CDN : Content Delivery Network

Un peu d'histoire sur les réseaux CDN par exemple utilisés par la VOD, ...

- ▶ "Serveurs centralisés"

Traditionnellement, les fournisseurs de services numériques dépendaient de serveurs centralisés pour héberger et distribuer des données à leurs utilisateurs ... par exemple de la vidéo.

Dans ce modèle, quel que soit l'endroit où se trouve l'utilisateur, les requêtes devaient parcourir de longues distances jusqu'au serveur central, ce qui pouvait se traduire par des temps de latence et des temps de chargement lents.

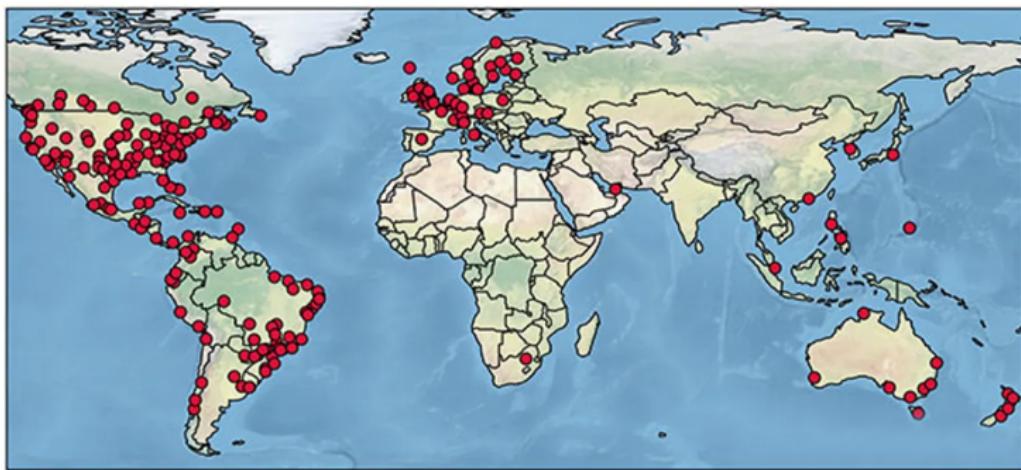
- ▶ CDN or "Content Delivery Networks" :

Les CDN ont été introduits à la fin des années 90 pour relever ces défis.

Leur fonction première était **de mettre en cache** des contenus populaires sur des **serveurs stratégiquement situés dans le monde entier**.

## CDN : Netflix 2016

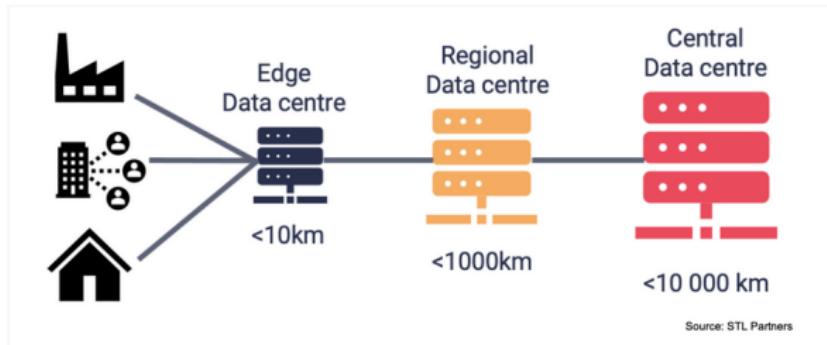
Désormais, quel que soit l'endroit où se trouve un utilisateur, le contenu peut lui être apporté rapidement en interrogeant le réseau de serveurs de mise en cache du CDN au lieu d'interroger le serveur central.



<https://spectrum.ieee.org/researchers-map-locations-of-4669-servers-in-netflixs-content-delivery-network>

" A group from Queen Mary University of London (QMUL) traced server names to identify 4,669 Netflix servers in 243 locations around the world."

Cependant, un CDN n'était rien de plus qu'un stockage à réponse rapide sur le cloud et ne pouvait pas faire de calcul significatif.



<https://stlpartners.com/articles/edge-computing/3-reasons-why-edge-will-change-video-streaming/>

L'introduction du Edge computing dans les CDN contribue à améliorer l'expérience utilisateur (UX) en influant sur plusieurs points :

- ▶ la qualité des flux,
- ▶ la fiabilité des flux,
- ▶ la latence, ...

## Comment ?

Le Edge CDN intervient

- ▶ sur l'encodage vidéo (ajuster la compression selon le lien "final")
- ▶ sur la collecte de statistiques (de consultations/de flux/...),
- ▶ sur l'insertion de flux par exemple publicités :-)
- ▶ sur l'ingestion : le transfert ou la conversion de fichiers vidéo depuis diverses sources (caméras, serveurs, disques durs, etc.) vers une plateforme où ils seront traités, édités ou redistribués.

Ce processus peut par exemple inclure des métadonnées qui facilitent l'organisation et la gestion des fichiers vidéo.

Encore beaucoup d'exemples utilisant le Edge computing : 5G, ...

# Avantages du Edge Computing I

Les raisons d'adoption et les bénéfices attendus sont divers et multiples:

<https://www.redhat.com/fr/resources/automation-at-the-edge-ebook>

1. L'edge computing peut accélérer les services et les rendre plus stables, à moindre coût.

Pour les utilisateurs, l'edge computing se traduit par une expérience **plus rapide**.

De leur côté, les entreprises et les fournisseurs de services profitent d'applications hautement disponibles et à **faible latence** avec une surveillance en temps réel.

2. L'edge computing permet de réduire les coûts de réseau, de lever les contraintes liées à la bande passante, d'écourter les délais de transmission, de limiter les pannes de service et de mieux contrôler la circulation des données sensibles.

Les temps de chargement sont réduits et les services en ligne déployés au plus près des utilisateurs offrent des capacités de mise en cache à la fois dynamiques et statiques.

## Avantages du Edge Computing II

3. L'edge computing permet aussi d'**effectuer localement des analyses et de la compilation du Big Data**, ce qui est indispensable pour les prises de décision en quasi temps réel.
4. L'edge computing **réduit le risque d'exposition des données sensibles**, car les opérations de traitement sont effectuées localement.

Les entreprises peuvent ainsi mieux appliquer les mesures de sécurité et se conformer plus facilement aux réglementations en vigueur.

5. Les sites régionaux qui disposent d'une capacité de calcul locale peuvent **continuer à fonctionner indépendamment d'un site principal**, même en cas de panne sur ce dernier.

Le coût de la bande passante nécessaire à la transmission des données entre le site principal et les sites régionaux est également considérablement réduit lorsque le traitement s'effectue à proximité de la source des données en question.

# Avantages du Edge Computing III

6. Avec une plateforme d'edge computing, les processus d'exploitation et de développement d'applications deviennent plus cohérents.

Ce type de plateforme doit **favoriser l'interopérabilité** afin de tenir compte de davantage d'environnements matériels et logiciels, plutôt qu'un unique datacenter.

Une stratégie d'edge computing efficace permet d'utiliser conjointement des produits de différents fournisseurs dans un écosystème ouvert.

# Défis liés à l'Edge Computing I

L'infrastructure sous-jacente au déploiement de l'EC n'est cependant pas toujours facile à mettre en œuvre et à gérer.

<https://www.redhat.com/fr/topics/edge-computing/what-is-edge-computing>

- ▶ Il peut être plus **difficile de distribuer la capacité de serveurs en périphérie vers plusieurs petits sites** que d'ajouter cette même capacité à un unique datacenter centralisé.

Les petites structures/entreprises peuvent avoir plus de mal à gérer l'augmentation des frais associés aux emplacements physiques.

- ▶ Les sites réservés à l'edge computing sont souvent distants, avec une **expertise technique locale limitée**, voire inexistante.

En cas de panne sur le site, l'infrastructure en place doit être facilement réparable par une main-d'œuvre locale sans expertise technique sous la direction d'un petit nombre d'experts regroupés sur un autre site.

- ▶ Les **opérations de gestion des sites** doivent être facilement **reproductibles** sur tous les sites réservés à l'edge computing afin de simplifier la gestion et le dépannage.

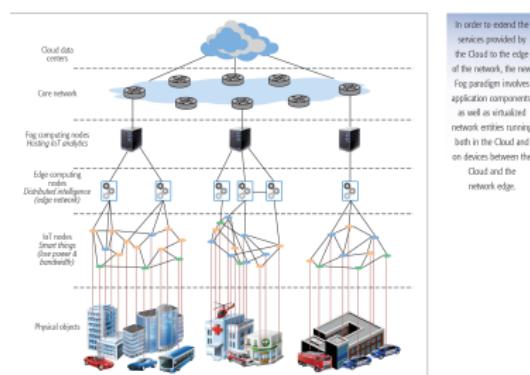
Les difficultés surviennent lorsque la mise en œuvre des logiciels est réalisée de façon légèrement différente d'un site à l'autre.

- ▶ Le **niveau de sécurité physique** des sites réservés à l'edge computing est souvent très inférieur à celui des sites principaux.

Toute stratégie d'edge computing doit prendre en compte un risque accru d'actes malveillants ou d'accidents.

# Federated Learning

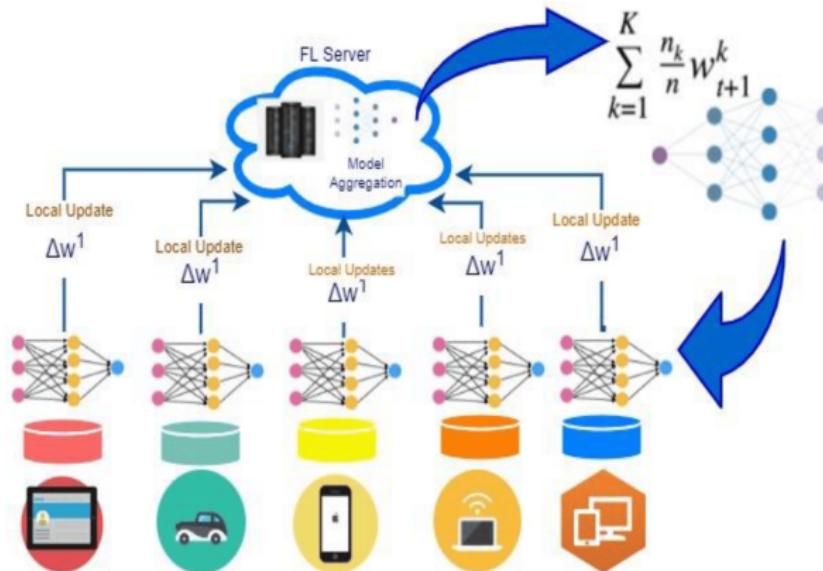
Très bien ! Le Edge computing c'est super ...



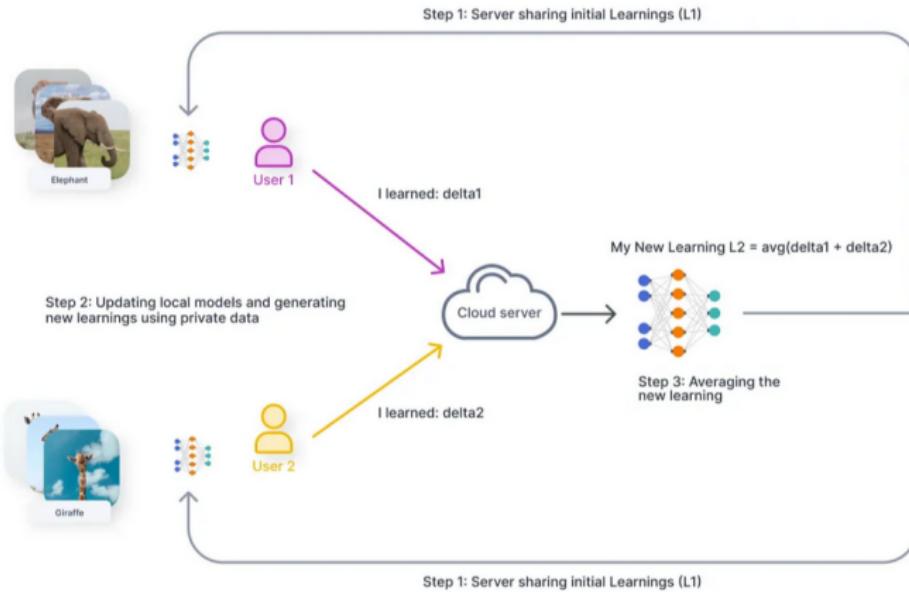
mais **comment on le met en oeuvre ?**

# Comment ca marche ?

L'apprentissage fédéré est une technique d'apprentissage automatique dans laquelle **plusieurs clients entraînent en collaboration un modèle d'apprentissage automatique** :



<https://www.mdpi.com/2079-9292/11/4/670>



- ▶ Après avoir été initialisé par le cloud,
- ▶ **chaque client entraînant un modèle local** sur son propre ensemble de données local,
- ▶ **puis partageant les poids avec d'autres clients (via le cloud) pour permettre la collaboration**, généralement sur plusieurs cycles.

# Utilité ? RGPD !

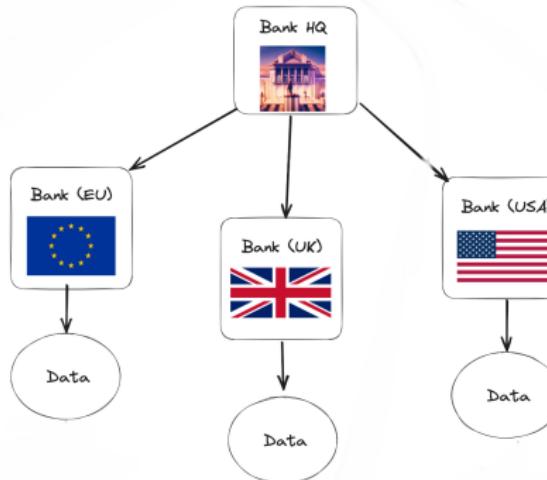
Au delà des performances et des différents avantages mentionnés pour le edge computing , l'apprentissage fédéré est aussi utile dans les scénarios où **les données sont stockées à différents endroits mais sans pouvoir** les centraliser dans une démarche d'apprentissage supervisé traditionnel.

Cela peut être le cas, par exemple

- ▶ si le volume global des données est " BIG",
- ▶ si il ces données sont en constante évolution,
- ▶ ou si **il serait illégal** de leur faire passer une frontière en raison du RGPD !

Prenons par exemple le cas d'une banque qui exerce ses activités dans le monde entier ...

En raison du RGPD, il se peut que la banque ne puisse pas transférer ses données de l'UE vers les États-Unis pour les entraîner ses modèles, et l'inverse peut également être vrai.

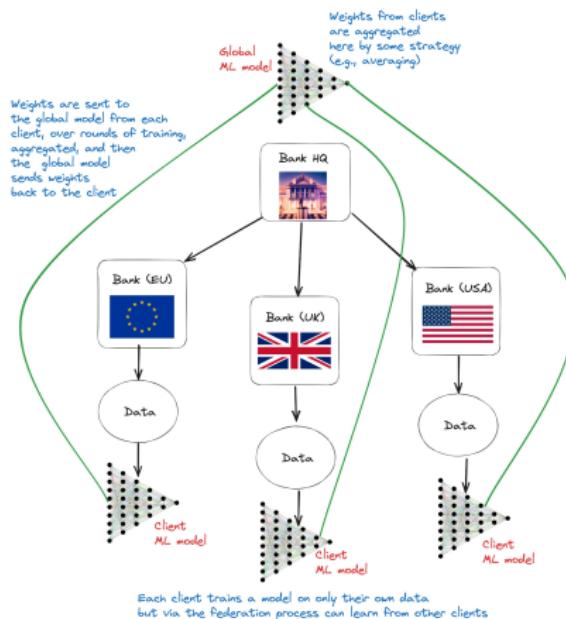


En considérant de plus en plus de pays dans lesquels la banque peut opérer, il devient clair qu'il existe **un grand nombre d'ensembles de données cloisonnés qui ne peuvent pas être centralisés**

- ▶ Chaque ensemble de données ne peut pas quitter le **silo** dans lequel il vit  
:-)

# Apprentissage collaboratif

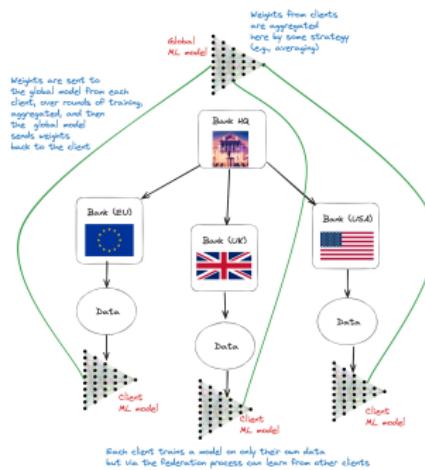
Et pourtant, les modèles d'apprentissage profond ne sont jamais aussi bons que sur des ensembles de données importants MAIS **dans ce cas le modèle central n'y a pas accès !**



[https://wandb.ai/wandb-smle/federated\\_launch/reports/Federated-Learning-with-Weights-Biases--Vm1ldzo0MTE10Dcz](https://wandb.ai/wandb-smle/federated_launch/reports/Federated-Learning-with-Weights-Biases--Vm1ldzo0MTE10Dcz)

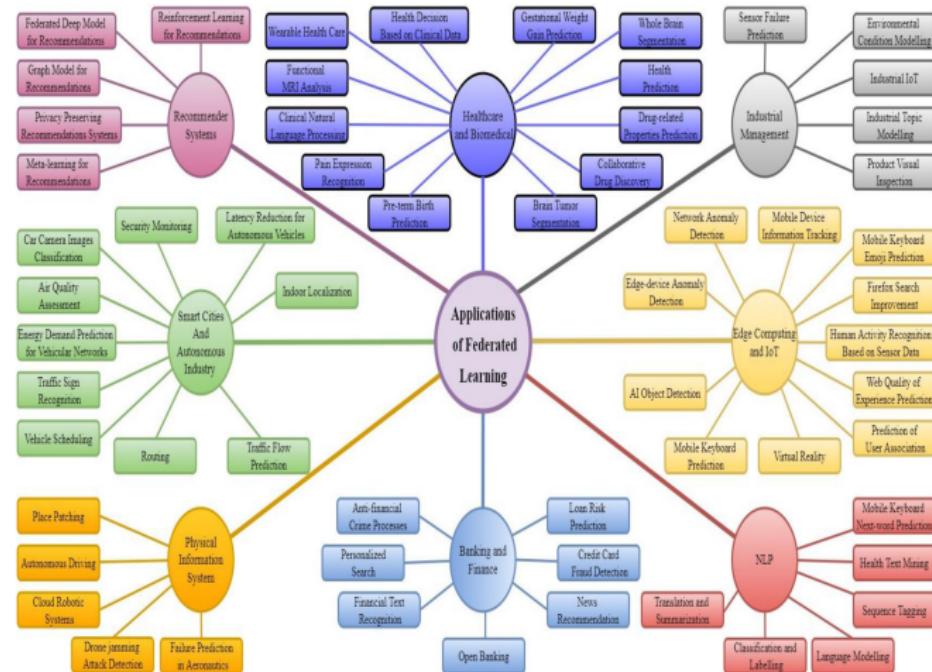
C'est précisément là que l'apprentissage fédéré peut être utile car dans une telle configuration,

1. un modèle serait formé dans chaque silo, sur des données qui ne quittent jamais le silo !
2. et les poids de chaque modèle seraient agrégés ensemble dans le centre du réseau.



- ▶ En agrégeant les poids du modèle local avec d'autres modèles locaux, **une forme d'apprentissage collaboratif se met en place.**

# Domaines d'applicabilité



From : Applications of Federated Learning; Taxonomy, Challenges, and Research Trends

# Use Case "Federated Learning" : MNIST

Dans la suite, nous allons utiliser le Federated Learning dans le contexte de MNIST.

# Quelques références

C'est là que ChatGP se nourrit !

- ▶ [https://groupes.renater.fr/wiki/ml-mtp/\\_media/wiki/intro\\_federated\\_learning\\_bellot.pdf](https://groupes.renater.fr/wiki/ml-mtp/_media/wiki/intro_federated_learning_bellot.pdf)
- ▶ <https://dasci.es/transferencia/open-data/federated-learning-tutorial/>
- ▶ <https://blog.openmined.org/federated-learning-types/>
- ▶ <https://risingwave.com/blog/mastering-vertical-federated-learning-a-comprehensive-guide/>
- ▶ <https://medium.com/aimonks/federated-learning-in-ml-on-the-edge-in-iot-830734e2c59b>
- ▶ <https://flower.ai/docs/framework/tutorial-series-what-is-federated-learning.html>
- ▶ <https://www.ieee-jas.net/article/doi/10.1109/JAS.2024.124215>
- ▶ <https://medium.com/disassembly/architecture-of-federated-learning-a36905c1d225>

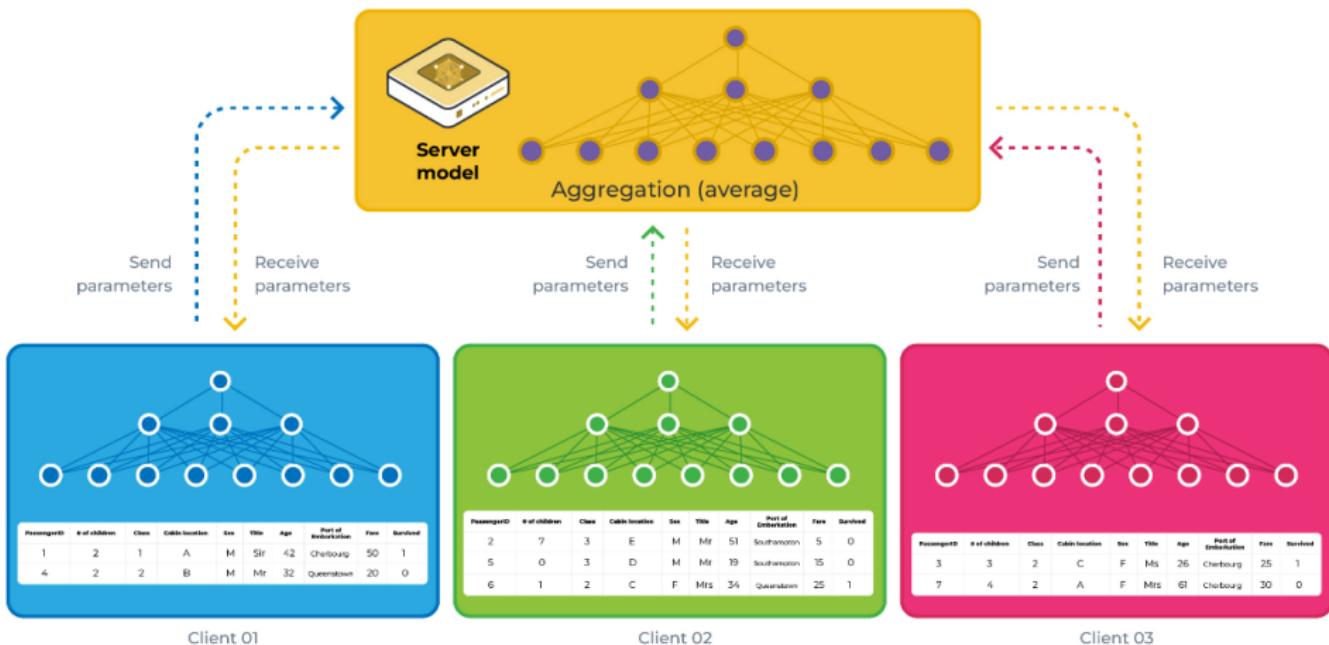
# Ce qui les différencie ...

Dès lors que l'on aborde le problème de la définition d'un modèle sous une forme distribuée, plusieurs possibilités existent.

Voyons en quoi elles diffèrent au niveau de ...

- ▶ la "répartition des données", les clients (/edge hosts)
- ▶ de l'entraînement,
- ▶ de l'agrégation des modèles,
- ▶ de la sécurisation des données,
- ▶ ...

# Horizontal Federated Learning

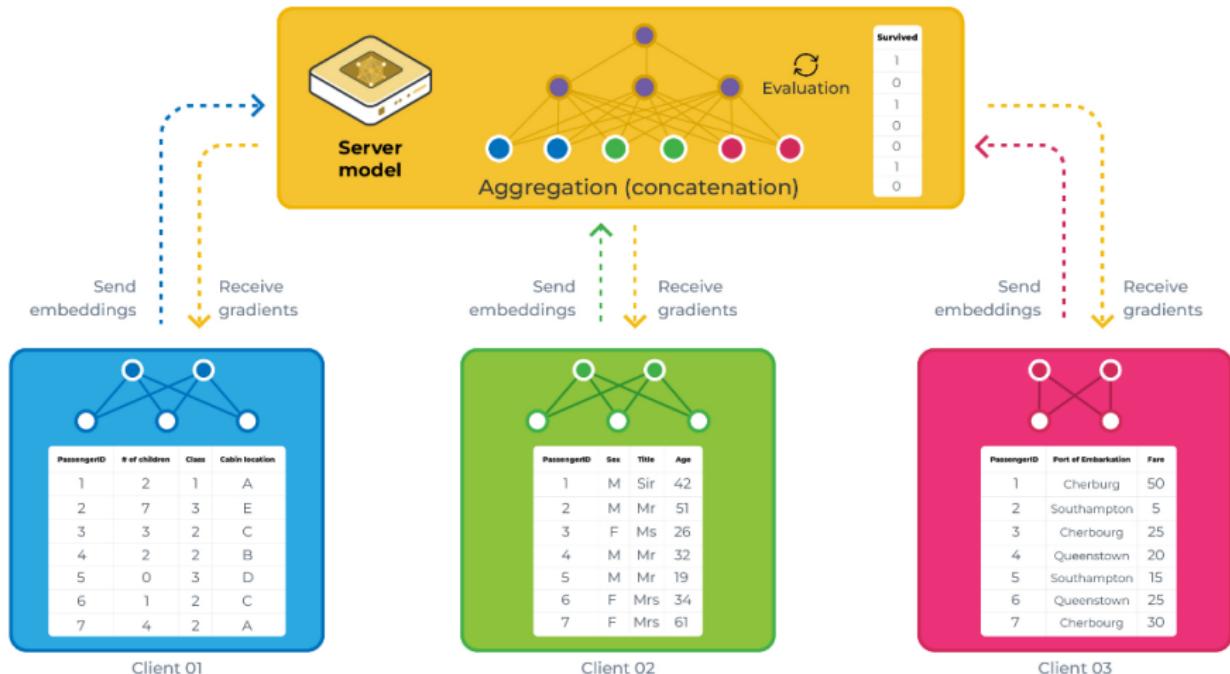


# HFL: Horizontal Federated Learning

HFL, sans doute une des stratégies les plus simples :

1. Au niveau de la "répartition des données", les clients (/edge hosts) :
  - ▶ Mettent en oeuvre le même modèle/réseau
  - ▶ MAIS sur des instances/individus différents
  - ▶ Ces individus sont différents MAIS leurs features sont les mêmes.
2. **Chaque client entraîne un modèle sur ses données locales**, qui contiennent toutes les colonnes de caractéristiques de ses échantillons.
3. **Le serveur agrège ces modèles locaux** en calculant par exemple la moyenne des paramètres ou des gradients pour mettre à jour un modèle global.
4. Les données brutes restent du côté du client, seules les mises à jour du modèle sont partagées, ce qui contribue à préserver la confidentialité.

# Vertical Federated Learning



# VFL : Vertical Federated Learning

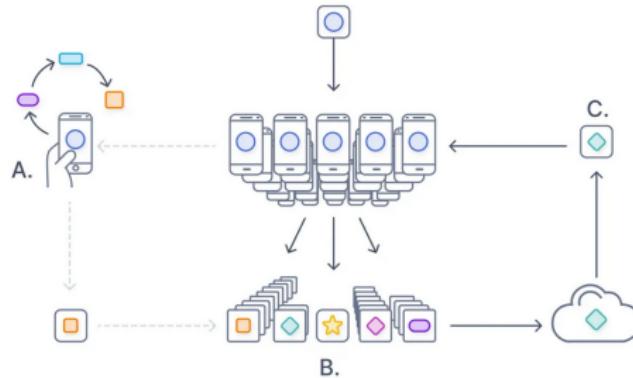
1. Au niveau de la "répartition des données", les clients (/edge hosts) :
  - ▶ Mettent en oeuvre des modèles différents.
  - ▶ **Ils travaillent avec la même population, MAIS ...**
  - ▶ Chaque modèle ne voit qu'une partie des features de la population
2. Les clients forment des modèles sur leurs caractéristiques respectives **sans avoir accès à l'ensemble des caractéristiques**.
  - ▶ **Chaque modèle ne voit qu'une tranche "Verticale" des données.**
3. **Le serveur agrège les mises à jour** telles que les gradients ou les paramètres, qui sont ensuite utilisés pour mettre à jour le modèle global.
  - ▶ Comme chaque client ne voit qu'une partie, **le serveur a généralement un rôle plus complexe** : coordonner des stratégies d'agrégation plus sophistiquées faisant appel à des techniques de calcul multipartites sécurisées.
4. VFL est conçu pour garantir qu'**aucun participant ne puisse accéder à l'ensemble des caractéristiques d'un échantillon**, préservant ainsi "une" confidentialité des données.

# Apprentissage fédéré centralisé

L'apprentissage fédéré centralisé nécessite un serveur central.

- ▶ Il coordonne la sélection des appareils clients au début et rassemble les mises à jour du modèle pendant la formation.

La communication ne se fait qu'entre le serveur central et les appareils périphériques individuels.



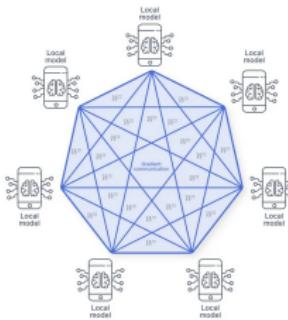
Bien que cette approche semble simple et génère des modèles précis, le serveur central pose un problème de goulet d'étranglement et les défaillances du réseau peuvent interrompre l'ensemble du processus.

# Apprentissage fédéré décentralisé

L'apprentissage fédéré décentralisé ne nécessite pas de serveur central pour coordonner l'apprentissage.

- Au lieu de cela, **les mises à jour du modèle sont partagées uniquement entre les appareils périphériques interconnectés.**

Le modèle final est obtenu sur un dispositif périphérique en agrégeant les mises à jour locales des dispositifs périphériques connectés.



Cette approche permet d'éviter la possibilité d'une défaillance en un seul point. Toutefois, la précision du modèle dépend entièrement de la topologie du réseau des appareils périphériques.

# Apprentissage fédéré hétérogène

L'apprentissage fédéré hétérogène implique d'avoir des clients hétérogènes tels que des téléphones mobiles, des ordinateurs ou des appareils IoT (Internet des objets).

- ▶ Ces appareils peuvent différer en termes de matériel, de logiciels, de capacités de calcul et de types de données.

HeteroFL a été développé en réponse aux stratégies courantes d'apprentissage fédéré qui supposent que les attributs des modèles locaux ressemblent à ceux du modèle principal.

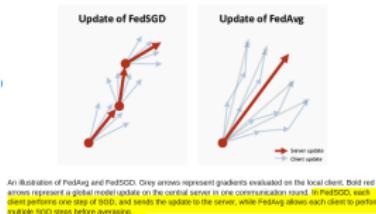
Mais dans le monde réel, cela n'arrive que très rarement.

- ▶ HeteroFL peut générer un modèle global unique pour l'inférence après l'entraînement de plusieurs **modèles locaux variés**.

# Descente stochastique du gradient fédérée (FedSGD)

Dans la descente de gradient stochastique traditionnelle, les gradients sont calculés sur des mini-lots (/mini batch), qui représentent une fraction des échantillons de données obtenus à partir de l'ensemble des échantillons.

- Dans le cadre FedSGD, ces mini-lots peuvent être considérés comme des dispositifs clients différents qui comprennent des données locales.



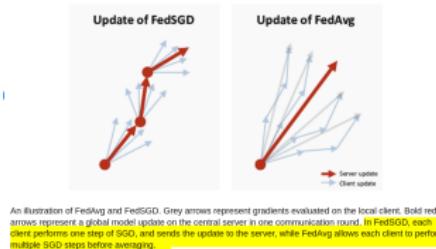
1. Le modèle central est distribué aux clients,
2. et **chaque client calcule les gradients à l'aide des données locales.**
3. Ces gradients sont ensuite transmis au serveur central, qui les **agrège proportionnellement au nombre d'échantillons présents sur chaque client pour calculer l'étape de descente du gradient.**

# FedAvg : Federated Averaging Algorithm

La moyenne fédérée (FedAvg) est une extension de l'algorithme FedSGD.

Les clients peuvent effectuer plus d'une mise à jour locale de la descente de gradient.

1. Au lieu de partager les gradients avec le serveur central, **les poids accordés sur le modèle local sont partagés**.
2. Enfin, le serveur agrège les poids des clients (paramètres du modèle).



La moyenne fédérée est une généralisation de FedSGD :

- ▶ si tous les clients partent de la même initialisation, la moyenne des gradients équivaut à la moyenne des poids.

# Federated Optimization

De façon implicite, l'apprentissage fédéré suppose **la résolution d'un problème d'optimisation fédérée**.

C'est un problème dont les caractéristiques sont particulières (cf [?]) néanmoins l'idée est qu'une solution s'applique à tout objectif de la forme

$$\min_{w \in \mathbb{R}^d} f(w)$$

avec  $f(w)$  décomposable sous forme d'une somme finie :

$$f(w) = \frac{1}{N} \sum_{i=1}^N f_i(w) \tag{1}$$

Dans le contexte du machine learning, la fonction  $f_i$  sera la fonction de perte (loss function) :

$$f_i(w) = loss(x_i, y_i; w)$$

qui représente la perte sur la prédiction en  $x_i$  de  $y_i$  faîte par un modèle  $w$  (déterminé par ses poids).

Supposons qu'il y ait  $K$  clients/edges sur lesquels les données ont été partitionnées.

Soit  $P_k$  l'ensemble des indices des points/individus attribués au client  $k$  et  $n_k = |P_k|$  le cardinal de  $P_k$ .

On peut re écrire, la fonction objectif (1) comme

$$f(w) = \sum_{k=1}^K \frac{n_k}{K} F_k(w)$$

avec

$$F_k(w) = \frac{1}{n_k} \sum_{i \in P_k} f_i(w)$$

Ce qui (sous certaines conditions sur le partitionnement : (IID) indépendantes et identiquement distribuées) signifie que la moyenne des pertes sur les edges  $F_k(w)$  vaut la perte sur le modèle global ( $f(w)$ ).

# FedAvg : Central makes its weights from edges

- ▶  $C$  is a fraction of clients
- ▶ The  $K$  clients are indexed by  $k$
- ▶  $\beta$  is the local minibatch size
- ▶  $E$  is the number of local epochs

**Result:** Server executes this code and Learn with edges !

```

initialise  $w_0$                                      /* Poids initiaux du central */
for each round  $t=1, 2, \dots$  do /* A chaque round, le central met à jour ses poids */
     $m \leftarrow \max(C.K, 1)$            /* On prend une fraction des K clients/edges, */
     $S_t \leftarrow$  random set of  $m$  clients/edges          /* avec au minimum 1 edge */
    for each client  $k \in S_t$  in parallel do           /* Tous font évoluer leurs poids */
        |    $w_{t+1}^k \leftarrow ClientUpdate(k, w_t)$       /* Le client  $k$  calcule ses poids */
    end
     $m_t \leftarrow \sum_{k \in S_t} n_k$                   /* Le modèle central évolue en calculant */
     $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$  /* la moyenne pondérée des poids des edges */
end

```

# FedAvg : Each edge computes its contribution

**Result:** Each client updates its weights

```
ClientUpdate(k,w) :           /* Run on client k, updates its "w" weights */;  
  
for each local epoch  $i$  from 1 to  $E$  do  
    /* Pour toutes les époques prévues localement */;  
    for batch  $b \in \beta$  do  
        /* chaque batch fait évoluer les poids locaux */;  
         $w \leftarrow w - \eta \nabla l(w; b)$  ;  
    end  
end  
return (/send)  $w$  to server;
```

## Autres algorithmes

cf <https://www.researchgate.net/publication/369417303>

Since then, numerous variants of FedAvg algorithms have been created to handle many of the federated learning challenges, including

- ▶ “FedProx”,
- ▶ “FedOpt”,
- ▶ “FedMa”,
- ▶ ...

# Some challenges in federated learning

cf [https://groupes.renater.fr/wiki/ml-mtp/\\_media/wiki/intro\\_federated\\_learning\\_belle.pdf](https://groupes.renater.fr/wiki/ml-mtp/_media/wiki/intro_federated_learning_belle.pdf)

et <https://www.v7labs.com/blog/federated-learning-guide>

# TODO

C'est un sujet "libre" autour du FL.

- ▶ Rien de plus dangereux ;-) car c'est vous qui posez le sujet ... et cela en dit long !

Je veux mesurer votre réflexion et pour faire cela il faut vous poser des (les bonnes) questions !

1. Finir de faire marcher correctement,
2. Faire marcher avec un approvisionnement des données "par les edges" ?
3. Etre critique : qu'est ce qui se passe SI ... ??? Vous avez tout un tas de challenges dans le document de Bellet référencé au début !
4. Comparer avec d'autres approches que FedAVG,
5. ... etc

**A la fin, un "rapport écrit" et le code qui va avec.**

Je vous propose un "début" de Federated Learning appliqué à MNIST => TSVP

# Ce qui est donné ... |

Il y a trois fichiers :

1. `f1_dataquest.py`

Il s'agit de récupérer la base des chiffres et d'en faire des Datasets.

- ▶ En fait, il faut se placer dans le cas de données d'entraînements dont la taille ne "permet PAS" le load classique !

La fonction `tf.data.Dataset.from_tensor_slices` dans TensorFlow est utilisée pour créer un pipeline de données (un objet Dataset) à partir de tenseurs (ou tableaux NumPy).

- ▶ Elle découpe ("slice") les tenseurs passés en entrée le long de la première dimension pour générer des éléments individuels du dataset.

Pour quoi faire ?

- ▶ Convertir des données en un format itérable pour l'entraînement de modèles.
- ▶ Associer automatiquement des features et des labels (si les données sont fournies sous forme de tuple ou de dictionnaire).
- ▶ Faciliter le prétraitement (normalisation, augmentation) et la gestion de gros datasets.

## Ce qui est donné ... II

### 2. `fl_model.py`

Dans la mesure où serveur central et edge ont **le même modèle (RNN)**, je me disais que commencer à avoir une approche objet pouvait être plus propre !

- ▶ Vous trouverez une classe "MyModel" qui définit le RNN utilisé dans cette implémentation du FL.
- ▶ C'est construit sur TensorFlow/Keras ... donc c'est fait pour être simple.

Dans ce module, il y a un test qui permet de donner les bases de l'utilisation d'un MLP.

Il y a aussi quelques lignes pour montrer comment on peut "jouer" avec les coefficients du modèle.

### 3. `edge_sim.py`

Ce fichier est en "cours de construction". Je voulais vous montrer UNE itération du FL :

- ▶ Après avoir récupéré les données,

## Ce qui est donné ... III

- ▶ Le serveur central utilise les données pour calculer un entraînement de son modèle.  
Dans cette version il a accès à toutes les données ... cela se discute ... mais disons que c'est la version optimal ?  
Et cela peut être utile pour comparer/évaluer le FL.
- ▶ Je crée des "edges" qui sont autant de machines qui vont participer au FL
- ▶ Ces edges reçoivent chacun un morceau de données **ET** ils reçoivent le modèle initialisé par le serveur central.
- ▶ Ensuite ils entraînent leurs modèles,
- ▶ Et font remonter les coefficients au serveur central.
- ▶ Enfin on compare, les performances entre le modèle central calculé classiquement et celui calculé en intégrant les coefficients des edges.
- ▶ ...

Ce code est encore incomplet fonctionnellement et structurellement

## Petit rappel sur la mise à jour des poids

La mise à jour des poids dans Federated Averaging (FedAvg) suit la formule suivante :

$$w^{(t+1)} = \sum_{k=1}^K \frac{n_k}{n} w_k^{(t)}$$

Explication des termes :

- ▶  $w^{(t+1)}$  : Nouveaux poids globaux après agrégation.
- ▶  $w_k^{(t)}$  : Poids du modèle du client  $k$  après entraînement local.
- ▶  $n_k$  : Nombre d'exemples de données détenus par le client  $k$ .
- ▶  $n = \sum_{k=1}^K$  : Nombre total d'exemples de données dans l'ensemble fédéré.
- ▶  $K$  : Nombre total de clients.