الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université Akli Mohand Oulhadj - Bouira -
Tasdawit Akli Muḥend Ulḥağ - Tubirett -

وزارة التعليم العالي والبحث العلمي
جامعة أكلي محند أولحاج
- البويرة -

Faculté : Sciences et Sciences Appliquées
Département : Informatique

Thèse

Présentée par
Mr. NAOUI Mohammed Anouar

Pour l'obtention du diplôme de
Doctorat en science
En Informatique

Thème

Une Architecture Multi Couche pour le Data Mining

Dans les systèmes Big Data

Soutenu le: **15/06/2021**

Devant le jury composé de :

Dr. AMAD Mourad, MCA,	Président	Université de Bouira
Dr. LEJDEL Brahim, MCA,	Rapporteur	Université d'El-Oued
Dr. AYAD Mouloud, MCA,	Co-Rapporteur	Université de Bouira
Pr. KAZAR Okba, Professeur,	Examinateur	Université de Biskra
Pr. KHOLLADI Med Kheireddine, Professeur,	Examinateur	Université d'El-Oued
Dr. SAOUD Bilel, MCA,	Examinateur	Université de Bouira
Pr. ALARABI Louai, Professeur,	Invité	Univ. d'UMM-ALQURA, Arabie Saoudite

Laboratoire d'informatique, Mathématique et Physique pour l'agriculture et les Foret (LIMPAF)

Remerciements

Je voudrais dans un premier temps remercier, mon directeur de thèse Dr. LEJDEL Brahim, maître de conférences à l'université El-Oued, pour la haute responsabilité, ses conseils, ses encouragements, et pour la confiance qu'il m'a portée.

J'adresse mes sincères remerciements au Dr. AYAD Mouloud, maître de conférences à l'université de BOUIRA, pour ses conseils, ses critiques et pour son soutien. Je remercie énormément le Prof. MELLOULI Sehl, professeur à l'université de LAVAL, Canada pour ses conseils et son soutien malgré les conditions critique durant la pandémie de COVID19.

Je remercie en premier lieu le président du jury Dr AMAD Mourad, maître de conférences à l'université de Bouira, ainsi que les membres du jury, Pr. KAZAR Okba, Professeur, à l'université de Biskra, Prof. KHOLLADI Med Kheireddine, Professeur, à l'université d'El-Oued et Dr. SAOUD Bilel, maître de conférences à l'université de Bouira, pour avoir participé à l'évaluation de ce travail.

Je remercie chaleureusement tous les membres de ma famille tous mes amis pour leur support et leurs encouragements. Enfin, on remercie tous ceux qui nous ont aidés de près ou de loin dans l'élaboration de ce travail.

Travaux scientifiques réalisés

La liste suivante représente les articles publiés dans le cadre de cette thèse

Revues internationales

1. Naoui Mohammed Anouar, Lejdel Brahim, Ayad Mouloud, Belkeiri Riad, et al.2020c. Integrating deep learning, social networks, and big data for healthcare system. Bio-algorithms and med-systems, 16(1). 2, 57
2. Naoui Mohammed Anouar, Lejdel Brahim, Ayad Mouloud, Amamra Abdelfattah, et al.2020d. Using a distributed deep learning algorithm for analyzing big data in smart cities. Smart and sustainable built environment.
3. Naoui Med Anouar, Lejdel Brahim, Ayad Mouloud, 2020a. Integrating iot devices and deep learning for renewable energy in big data system. Scientific bulletin.
4. Naoui Med Anouar, Lejdel Brahim, Ayad Mouloud, 2020b. Using k-means algorithm for regression curve in big data system for business environment. Revista cubana de ciencias informáticas, 14(2), 34–48.

Conférences internationales

1. Naoui Mohammed Anouar, Lejdel Brahim, Ayad mouloud, Amamra Abdelfattah et Okaba Kazar. Distributed machine learning approach based on big data system for energy consumption prediction. The first international Conference on artificial intelligence and its Applications (AIAP'18), 4-5 Decembre 2018, ELOUED.
2. Naoui Mohamed Anouar, Lejdel Brahim ,Ayad Mouloud , Amamra Abdelfatah, Kazar Okba. Using deep learning system for IoT device based on renewable energy, 2nd International Conference on Artificial Intelligence in Renewable Energetic Systems. ESC-KOLEA, Tipasa - Algeria, November 24 - 26, 2018.

Les résumés

Résumé

Au cours de cette décennie, la croissance des données est rapide à cause des informations qui circulent dans les entreprises, les données des utilisateurs des réseaux sociaux, la technologie des internet des objets, et l'utilisation des appareils mobiles. Cette croissance soulevée de nombreuses questions par les chercheurs dans le but d'extraction de connaissance de grandes masses de données. Cette discipline de recherche est le Data Mining pour le Big Data. Plusieurs chercheurs définis Big Data par les données caractérisées par leur volume, variété , vélocité, véracité, volatilité, et valeur. Notre vision est de traiter le lien entre le Data Minng et le Big Data. Dans ce travail, nous proposons une architecture multi couches pour le Data mining dans les systèmes Big Data. Cette architecture permettant le traitement efficace de processus de fouille de données. Notre architecture a deux contributions. La première est l'utilisation des algorithmes profonds et distribue pour améliorer deux axes très importants la durée d'exécution et le taux d'apprentissage de processus de Data Mining. Dans la deuxième contribution, nous focalisons sur l'intégration de diverses sources de données.

Mot Clé : Big Data, Data Mining, Architecture Big Data, Apprentissage profond distribue.

Abstract

During this decade, the growth of data is rapid due to the information circulating in companies, the data of social media users, the technology of the Internet of Things, and the use of mobile devices. This growth raises many questions for researchers to extract knowledge from large masses of data. This research discipline is Data Mining for Big Data. Several researchers define Big Data by data characterized by its volume, variety, velocity, veracity, volatility, and value. Our vision is to address the link between Data Mining and Big Data. In this work, we propose a multi-layered architecture for data mining in Big Data systems. This architecture allows the efficient processing of the data mining process. Our architecture has two contributions. The first is the use of deep and distributed algorithms to improve two critical areas of data mining process execution time and accuracy. In the second contribution, we focus on the integration of various data sources.

Keywords : Big Data, Data Mining, Architecture Big Data, Distributed Deep learning.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

Depuis le début de la croissance des données, à cause de l'évolution de la technologie, et l'augmentation de leur nombre d'utilisateurs, les chercheurs ont proposé le terme Big Data pour bien différencier les nouveaux problématiques liés. Ce concept est défini dans la revue nature par les données à grande échelle qui ne peuvent être présentées, traitées et analysées à l'aide des technologies, méthodes et théories existantes (Goldston, 2008).

Le Big data sont les données qui se caractérisent par :

— Volume : la quantité massive de données générées par les utilisateurs (Assunção *et al.*, 2015; El Kassabi *et al.*, 2018; Uddin *et al.*, 2014).

— Variété : les données varient en fonction de leur type qui peut être structuré, semi-structurées ou non structurées (Assunção *et al.*, 2015; El Kassabi *et al.*, 2018; Uddin *et al.*, 2014).

— Vélocité : la vitesse de génération des données (Assunção *et al.*, 2015; El Kassabi *et al.*, 2018; Uddin *et al.*, 2014).

— Véracité : l'exactitude et la précision des données (Assunção *et al.*, 2015; El Kassabi *et al.*, 2018; Uddin *et al.*, 2014).

— Volatilité : la durée pendant laquelle les données resteront valables (Uddin *et al.*, 2014).

— Valeur : la capacité de l'exploitation de données (Assunção *et al.*, 2015; Uddin *et al.*, 2014).

Le Big Data a attiré les chercheurs dans plusieurs axes de recherche a savoir, la proposition des frameworks conceptuels pour Big Data, le problème de stockage et de traitement de Big Data, le problème de l'analyse de Big Data. ect.

Ces architectures doivent répondre aux exigences de l'énorme volume de données, la variété des types de données, la vitesse d'acquisition et de traitement de données (Krishnan, 2013).

Le Data Mining est un axe de recherche actif. Ils ont défini comme le processus de l'extraction de connaissance à partir de données. Les chercheurs vus que le processus de Data Mining nécessite des méthodes et d'outils pour puissent être répondre aux exigences de Big Data. Le défi de Data Mining pour les applications Big Data est d'explorer les grands volumes de données et d'extraire des informations ou des connaissances utiles pour une action future (Rajaraman & Ullman, 2011). Le processus de Data Mining pour les systèmes Big Data doit pouvoir extraire les données qui caractérisent par volume, variété, véracité et volatilité (Qiu *et al.*, 2016).

Plusieurs travaux de recherche proposée des méthodes et des outils qui permettent l'extraction de connaissance de Big Data. Certains chercheurs suggérés des architectures pour le but de traiter tous les aspects liés aux Big Data, tel que leurs stockages, traitement, visualisation, et analyse. Autres chercheurs proposés une architecture Big Data sans poser la question de processus de Data Mining. Plusieurs architectures des systèmes Big Data sont proposées selon le classement suivant :

— Architecture générale : des architectures décrivent les composantes globales des systèmes Big Data (Anthony & Arjuna, 2011).

— Architecture référentielle : des architectures standard assurent l'interopérabilité des systèmes par la normalisation. Elle facilite l'instanciation de nouvelles architectures concrètes (Demchenko *et al.*, 2014; Levin, 2013; Supriya *et al.*, 2016; Sang *et al.*, 2016; Pääkkönen & Pakkala, 2015).

— Architecture orientée application : l'objective de cette architecture est la proposition d'une architecture pour une application quelconque (Osman, 2019; Bilal *et al.*, 2016; Rathore *et al.*, 2015; Mezghani *et al.*, 2015; Thusoo *et al.*, 2010a; Sumbaly *et al.*, 2013).

— Architecutre de Cloud Computing : architecture Big Data dans le Cloud Computing (Schmidt & Möhring, 2013).

Objectifs de la thèse

Malgré l'existence de plusieurs travaux de recherches, ces travaux restent insuffisantes et ne prend pas en compte l'impact d'architecture avec le processus de Data Mining dans deux axes très importants : la durée d'exécution et le taux d'apprentissage. De plus, les travaux précédents n'ont pas traité le processus de Data Mining comme un processus d'intégration de divers sources de données et technologies pour atteindre un objectif bien défini.

Les contributions de notre thèse sont exprimées par les questions suivantes :

— Existe-t-il une relation entre le choix de l'architecture et la performance du système Big Data dans le processus de Data Mining en termes de temps de traitement et de la prédiction de modèle ?

— Quels sont les avantages de l'apprentissage approfondi dans les systèmes Big Data.

— Quelle est l'architecture qui permette l'intégration de différentes sources de Big Data et technologies et quels sont leurs avantages ?

Pour répondre aux questions précédentes, nous proposons deux contributions (Naoui *et al.*, 2020a), (Naoui *et al.*, 2020b), (Naoui *et al.*, 2020c), (Naoui *et al.*, 2020d).

La première contribution est une proposition d'une architecture pour l'apprentissage approfondi distribué dans les villes intelligentes (Naoui *et al.*, 2020d). Cette architecture est évaluée par des métriques qui nous permettent de tester le temps d'exécution et le taux d'apprentissage. Les avantages de cette contribution sont : la souplesse de stockage et de traitement de données. Optimisation de temps d'exécution et le taux d'apprentissage des algorithmes.

Nous avons aussi proposé l'approche agent au Big Data pour améliorer la performance du système. L'architecture proposée est utilisée pour la modélisation de courbe régression . Les agents utilisés sont des agents pour la linéarisation des courbes et un agent pour déterminer le meilleur modèle par l'utilisation d'algorithme k-means (Naoui *et al.*, 2020b).

La deuxième contribution est une architecture d'intégration des réseaux sociaux et le système Big Data des hôpitaux pour la reconnaissance des maladies par des images radiographiques (Naoui *et al.*, 2020c). Les avantages de cette intégration sont : l'intégration des multiples sources de données, la réutilisation et le partage des modèles et connaissances entre les utilisateurs des réseaux sociaux.

L'intégration des déférentes technologies est utilisée dans le contexte des énergies renouvelables. Cette architecture nous permet de bien exprimer la relation entre la technologie des objets, Big Data et apprentissage profond. Les objets intelligents sont les senseurs de climat.

Ces objets acquièrent les données et le sauvegardent dans le système Big Data. L'apprentissage profond améliore le taux de prédiction (Naoui *et al.*, 2020a) .

Organisation de la thèse

Cette thèse est organisée en trois chapitres. Dans le premier chapitre nous abordons le Data Mining, leurs concepts et les différentes vues de ce concept par plusieurs chercheurs. En outre la relation entre Data minig, Machine Learning et le processus d'exploration de connaissance. D'autre part, nous décrivons les formalismes de base de Data Mining et les algorithmes les plus connus.

Dans le deuxième chapitre, nous présentons le Big data, leurs caractéristiques, architectures, les classes des architectures, et nous terminons par les contributions récentes des systèmes Big Data.

Le troisième chapitre est consacré à la présentation de nos contributions. Dans la contribution N°1, nous présentons l'architecture d'apprentissage approfondi distribué pour les maisons intelligentes, les métriques d'évaluation de cette architecture, comparaison, résultats et discussion. Dans la contribution N°2, nous présentons l'architecture pour l'intégration des réseaux sociaux, apprentissage approfondi et Big Data. L'expérimentation de cette architecture est réalisée dans le domaine de la santé. Les images radiographiques de multiples sources sont utilisées afin de prédire les maladies thoraciques.

Enfin, nous terminons par une conclusion générale qui récapitule notre contribution.

Chapitre 1

Data Minig

Sommaire

1.1 Introduction

L'évolution de la technologie d'information et de communication dans plusieurs secteurs a causée de nouvelles problématiques. L'évolution rapide des données mit de nouvelles questions dont l'objectif est le changement de grandes quantités des données, vers un ensemble des savoirs et des connaissances utiles dans tous les domaines. Dans nos jours, les activités des échanges des informations entre utilisateurs ou entreprises, prendre beaucoup de façon. Il s'agit d'une relation utilisateur-utilisateur ou groupe d'utilisateurs, par exemple le cas des réseaux sociaux, ou utilisateurs-entreprises via un site ou une application. Aussi, avec la révolution de technologie d'internet des objets, la relation prenait un facteur principal en terme de processus de traitement et de la gamme de données stockées. Les donnés collectés prendre l'intérêt des chercheurs afin de transformer ces données en ensemble de connaissances utilisables et exploitables. A cet effet, dans ce premier chapitre, nous présentons des définitions générales et les processus de Data Mining. Ensuite, nous décrivons la modélisation mathématique et les algorithmes d'apprentissage. De plus, nous arborons l'approche et l'optimisation d'apprentissage profond.

1.2 Définition de Data Mining

Data Minig est l'ensemble des savoirs fondé depuis les années 90 dont l'objectif est l'extraction de connaissance à partir des données (Piateski & Frawley, 1991; Han *et al.*, 2011). Certains chercheurs considèrent le processus de fouille donnée comme une étape de processus d'extraction de données (Knowlege Discovery in Data Base KDD) (Fayyad *et al.*, 1996). Autres, affirment que le processus de fouille de données et le processus d'extraction de données sont le même processus (Han *et al.*, 2000; Aggarwal, 2015). Un autre terme qui prend une large place dans les publications récentes Machine Learning. Le but de Machine Learning et Data Minig est l'extraction des connaissances utiles à partir des données (Kononenko & Kukar, 2007). Cependant, Machine Learning se concentre sur le développement de techniques de modélisation de données (Kononenko & Kukar, 2007). Les définitions ci-dessus montrent les différentes vues des chercheurs à la fouille de données :

1.2.1 Définition 1

Une nouvelle génération de théories et d'outils informatiques pour aider les individus à extraire des informations utiles (connaissances) des volumes croissants de données numériques (Fayyad *et al.*, 1996).

1.2.2 Définition 2

Le Data Mining est le processus de découverte de connaissances non triviales et utiles à partir de grande quantité de données (Abboud, 2018).

1.2.3 Définition 3

La fouille de données est l'art de l'exploitation de données pour la génération de connaissances afin de prédire ou décrire le comportement de données par l'utilisation d'un modèle (Stéphane, 2012).

1.2.4 Data Mining et processus d'extraction de connaissance

Comme nous avons déjà mentionné, certains chercheurs considèrent le processus de l'extraction de connaissance à partir des bases de données, comme le processus de Data Mining, dont le rôle est l'extraction automatique des modèles représente des connaissances implicitement stockées ou capturées dans des grandes bases de données, des entrepôts de données, le Web, d'autres référentiels d'informations massives ou des flux (Ceri *et al.*, 2003).

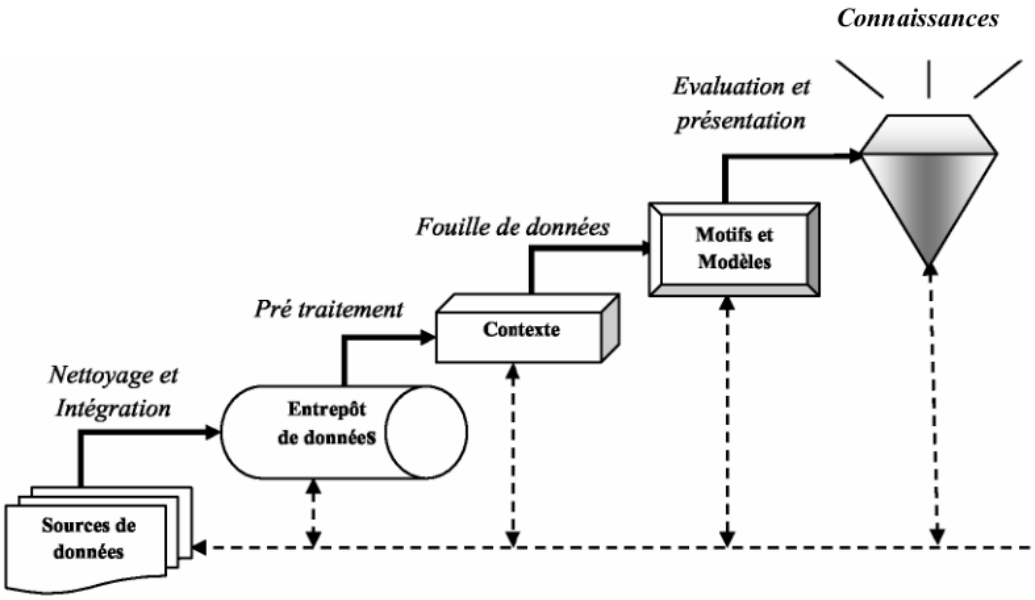

FIGURE 1.1 – Processus d'extraction de connaissance à partir des données (Fayyad *et al.*, 1996)

1.2.5 Data Mining et Machine Learning

Witten & Frank (2002) et Han *et al.* (2011) ont défini la fouille de données par l'extraction d'informations implicites, auparavant inconnues et potentiellement utiles à partir de données, dont la Machine Learning fournit la base technique pour l'exploration des données. Il est utilisé pour extraire les connaissances dans les bases de données.

1.2.6 Notre vision

Nous avons vu que la mise en œuvre d'un processus de Data Minig est une machine Learning. Dans le Data Mining, on concentre sur les aspects les algorithmes et le fondement théorique, mais dans la machine Learning on s'intéresse de plus par le développement. Pour cette raison, le terme Machine Learning est le plus utilisé dans la littérature scientifique et par les ingénieurs de développement. Par contre le terme Data Ming est rarement utilisé dans le domaine du développement.

1.3 Processus de Data Mining

Le processus de data Mining est l'ensemble des étapes à suivre du début de processus jusqu'à la fin, où on construit le modèle qui décrit, ou prédire les données. On peut dire que

le processus de fouille de données peut comporter trois étapes : préparation des données, apprentissage et validation.

— Préparation : consiste à sélectionner les données cohérentes et pertinentes au problème posé. Par exemple en évitant les données nulles, mal saisie, etc.

— Apprentissage : cette étape est le cœur de processus de data Minig. On applique les algorithmes et les méthodes qui nous permettent de décrire ou prédire le comportement de données.

— Validation : le modèle élaboré dans la phase précédente peut être testé par exemple, par test de prédiction, matrice de confusion, erreur quadratique, etc.

1.4 Modélisation mathématique d'apprentissage

Dans cette section, nous présentons le fondement mathématique d'apprentissage. On désigne par le terme apprentissage, l'ensemble des algorithmes qu'on peut apprendre à partir des données. Les deux principaux éléments dans n'importe quel type de Data Minig sont les données et l'algorithme ou modèle. Nous cherchons à répondre comment représenter les données et comment construire un modèle.

1.4.1 Données comme vecteurs, matrices et tenseurs

Les données sont formalisées sous la forme de vecteur, matrice ou tenseur.

— Vecteur de données : les données représentées par un vecteur. Supposons les données suivantes $(x_1, x_2, x_3, ...x_n)$. On peut traiter les données comme un vecteur (V) de longueur (n) dont les éléments de vecteurs respectivement sont : $x_1, x_2, x_3, ...x_n$.

$$V = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

— Matrice de données : les données peuvent être formalisées sous la forme matricielle, si les données décomposées en deux indicent par exemple une image décomposée en matrice de pixel de deux dimensions.

$$x = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{pmatrix}$$

— Tenseurs de données : les tenseurs sont des vecteurs de nombres disposés sur une grille régulière avec un nombre de variables d'axes supérieur ou égal 0. Par exemple un élément de tenseur A identifié par les coordonnés (i, j, k) par $A_{i,j,k}$. Les vecteurs sont des tenseurs d'ordre 1, et les matrices d'ordre 2.

1.4.1.1 Normes des données

Dans la phase d'apprentissage, on peut calculer les normes de données. Les normes ont plusieurs significations par exemple une distance, similarité, etc. La norme (L^p) mesuré la

taille d'un vecteur. Si $(p \in \mathbb{R})$ et $(p \geq 1)$), on a :

$$\| x \|_p = \left(\sum_i |x_i|^p \right)^{\frac{1}{p}} \tag{1.1}$$

Dans l'équation 1.1 si $(p = 2)$, l'équation retourne la norme euclidienne, ou c'est la distance euclidienne, et si $(p = 1)$ c'est la distance de Manhattan.

1.4.2 Modèles

Le modèle c'est l'algorithme d'apprentissage des données. Mitchell *et al.* (1997) considèrent qu'un programme informatique apprendrait de l'expérience (E), concerne une certaine classe de tâches (T) et une mesure de performance (P), si la performance aux tâches (T), mesurée par (P), s'améliore avec l'expérience E. L'expérience est la capacité d'apprendre les données. Les tâches sont le processus de la création du modèle. La performance (P) c'est le facteur qui mesure le modèle.

Il existe deux types de fonctions qui peut-être décrivent un modèle, le modèle comme fonction et le modèle comme probabilité de distributions (Deisenroth *et al.*, 2020).

1.4.3 Modèle comme fonction

L'apprentissage automatique formalisé par la fonction (f), l'espace de départ (X), et (Y) l'espace d'arrivée : $f : \quad X \longrightarrow Y$

Supposant (D) est l'ensemble de données à entraîner $D = \{(x_1, y_1), (x_2, y_2), ..(x_n, y_n)\}$ ou $\forall\, i \in [1..n]\ y_i = f(x_i)$, (g) est une fonction d'approximation de (f). Le choix de la fonction (g) parmi l'ensemble des fonctions candidates, se fait par l'utilisation d'hypothèse (H) . Le meilleur modèle sera sélectionné par les choix de fonction (g) qui représente bien la fonction (f) et les données.

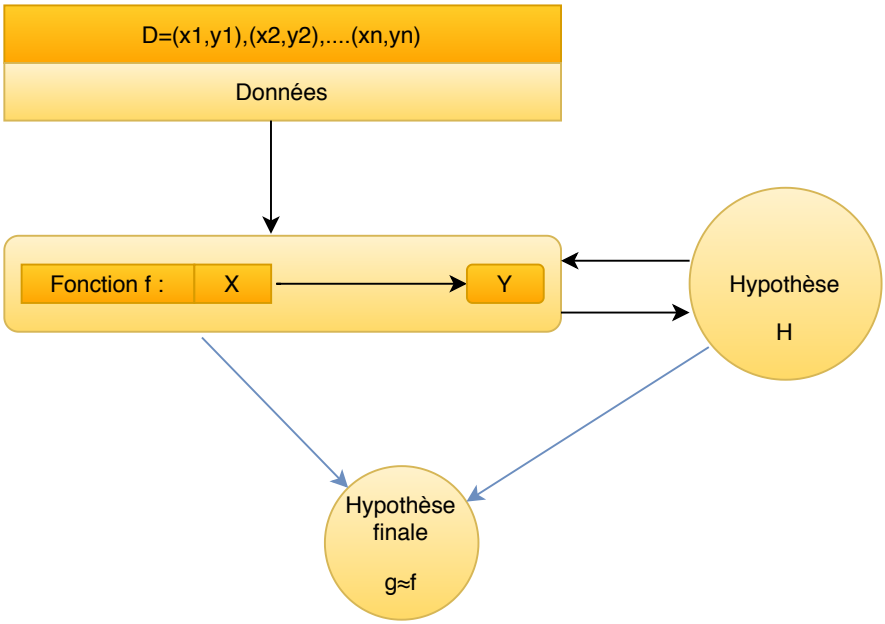

FIGURE 1.2 – Fonction d'approximation (Abu-Mostafa *et al.*, 2012)

Par exemple, si nous avons les données $D = \{(x_1, y_1), (x_2, y_2), ..(x_n, y_n)\}$. Dans un problème de régression linéaire, la fonction (f) est donnée par :

$$\begin{array}{rccl} & X & \longrightarrow & Y \\ f: & Y & = & f(X)+\epsilon \\ & Y & = & w_1x_1+w_2x_2+\ldots+w_nx_n+\epsilon \end{array}$$

L'ensemble des fonctions candidatures (H) définies par les valeurs de vecteur W= $(w_1, w_2, \ldots, w_n)$. Le meilleur fonction g calculée par l'optimisation d'erreur ϵ.

1.4.4 Modèles comme probabilité de distribution

L'incertitude est un concept clé dans le domaine d'apprentissage. Elle résulte du bruit sur les mesures des ensembles de données. Pour la quantification et la manipulation cohérente de l'incertitude, les chercheurs utilisent la théorie des probabilités qui constitue l'un des fondements de théorie de l'apprentissage.

Pour l'ensemble des données $D = \{(x_1, y_1), (x_2, y_2), ..(x_n, y_n)\}$ sont des variables aléatoires, exprimées par la distribution de probabilité P. Cette probabilité, prendre plusieurs formes notamment, probabilité marginale, probabilité conditionnelle, la règle de chaîne des probabilités conditionnelles, les lois de Bernoulli et Gaussian.

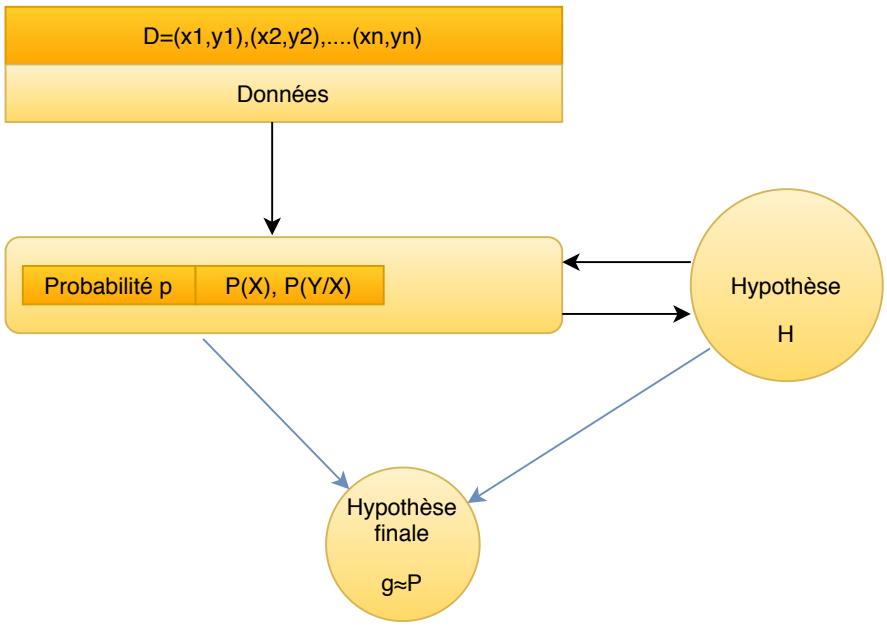

FIGURE 1.3 – Fonction probabiliste

— Probabilité marginale : c'est la distribution de probabilité sur un seul sous-ensemble. Supposons que nous ayons des variables aléatoires discrètes X et Y, et nous avons $P(X,Y)$. On peut trouver $P(X)$ avec la règle :

$$\forall x \in X, P(X=x) = \sum_y P(X=x, Y=y) \tag{1.2}$$

Pour les variables continues, nous devons utiliser l'intégration au lieu de la somme :

$$\forall x \in X, P(X=x) = \int P(X,Y)dy \tag{1.3}$$

— Probabilité conditionnelle : dans de nombreux cas, nous nous intéressons à la probabilité d'un événement, qu'un événement (X = x) se produit étant donné que l'événement (Y

= y) a eu lieu. Nous désignons la probabilité conditionnelle que (Y = y) étant donné (X = x) comme P(Y = y X = x). La probabilité conditionnelle peut être calculée par la formule :

$$P(Y = y|X = x) = \frac{P(Y = y, X = x)}{P(X = x)} \tag{1.4}$$

La règle de la chaîne de probabilité 1.5 (chain rule of probability) est dérivée par l'application successive de la règle du produit dérivé par l'équation 1.4

$$P(X_1, X_2, ..X_n) = P(X_1) \prod_{i=2}^{n} P(X_i|X_1, ..X_{i-1}) \tag{1.5}$$

— Distribution de Bernoulli : c'est une distribution discrète de probabilité dont la variable aléatoire peut prendre deux valeurs binaires, 1 pour la probabilité p et 0 pour la probabilité q=p-1.

Les propriétés de loi de Bernolli sont : $P(x) = \begin{cases} p & \text{si } x = 1 \\ 1-p & \text{si } x = 0 \\ 0 & \text{si non} \end{cases}$ La loi de Bernoullli calcule par l'équation :

$$P(X = x) = p^x(1-p)^{1-x}.x \in 1, n \tag{1.6}$$

— Distribution de Gauss : la distribution normale est la distribution de probabilité la plus importante en statistique, il décrit comment les valeurs d'une variable sont distribuées. Il s'agit d'une distribution symétrique où la plupart des observations se regroupent autour du pic central , et autres valeurs extrêmes dans les deux queues de la distribution. Supposant la moyenne des données μ et l'écart type σ. La loi normale de Gauss N :

$$N(x; \mu, \sigma^2) = \sqrt{\frac{1}{2\pi\sigma^2}} \exp -\frac{1}{2\sigma^2}(x-\mu)^2 \tag{1.7}$$

1.4.5 Types de modèle

Les relations entre les données (variables) déterminent le type d'algorithmes de Data Mining, dans les principaux types sont : classification, régression, transcription, traduction, détection d'anomalies (Goodfellow *et al.*, 2016).

— Classification : est défini par la fonction $f : \mathrm{R}^n \longrightarrow \{C_1, C_2...C_n\}$. Dont $y = f(x)$ est décrit par l'entrée (x) et la sortie (y) tel que $y \in \{C_1, C_2...C_n\}$.

— Régression : la fonction est définie $\mathrm{R}^n \longrightarrow \mathrm{R}^n$ la valeur de sortie est une valeur numérique. $y = f(x)$ retourne pour chaque valeur de x une valeur numérique de y.

— La machine de traduction : le rôle de ce type de modèle est la traduction de symbole depuis une langue vers une autre. Supposant qu'un ensemble de symboles de la langue $L_1 = \{x_1^1, x_2^1...x_n^1\}$ et $L_2 = \{y_1^2, y_2^2...y_m^2\}$. La langue L_1 et L_2 de langueur n, m. La fonction $f : L^1 \longrightarrow L^2$. Dont $y^2 = f(x^1)$. La machine traducteur est largement utilisé dans la traduction des langues.

— Transcription : la transcription est définie lorsque nous avons des données $D_1 = \{x_1^1, x_2^1...x_n^1\}$ et $D_2 = \{y_1^2, y_2^2...y_m^2\}$. Tel que le type est D_1 est t_1 et le type D_2 est t_2. La fonction de transcription est $f : D^1 \longrightarrow D^2$. Dont $y^2 = f(x^1)$. Par exemple si nous prenons une image de format pixel comme entrée et il retourne la sortie texte. La traduction et la transcription sont similaires sauf que dans la traduction les données en entrée et sortie sont du même type.

— Détection d'anomalie : La détection d'anomalie implique le cas ou une donnée n'est pas cohérente en relation avec une hypothèse (H). Supposant que (H) est estimé par un seuil s, Donc : $f(x) = \begin{cases} y = f(x,h) & \text{si y > s, Anomalie} \\ \text{Si non} & \text{Non anomalie} \end{cases}$

— Sortie structurée : dont la sortie est un vecteur (ou une autre structure de données contenant plusieurs valeurs) avec des relations importantes entre les différents éléments. Il s'agit d'une large catégorie qui englobe les tâches de transcription et de traduction décrites ci-dessus, mais également de nombreuses autres tâches. Un exemple est l'analyse syntaxique, ou le mappage d'une phrase en langage naturel dans un arbre qui décrit sa structure grammaticale et le marquage des nœuds des arbres comme étant des verbes, des noms ou des adverbes (Collobert, 2011) .

Les types de modèles sont classés en deux classes d'apprentissage : supervisé, et non supervisé. Dans l'apprentissage supervisé, les classes des données à prédire sont préalablement connues. Tandis que dans l'apprentissage non supervisé, les classes sont automatiquement générées . Le défi central de l'apprentissage automatique est que nous devons bien performer sur de nouveaux entrants inédits, pas seulement ceux sur lesquels notre modèle a été entraîné. La capacité de bien fonctionner sur des entrées précédemment non observées est appelée test.

1.4.6 Analyse des erreurs écarts-biais

Erreur due au biais : comme nous avons discuté dans la section précédente, le modèle est une fonction d'approximation. L'erreur de modèle, calculée par une mesure d'erreur sur l'ensemble d'entraînement, appelée erreur d'entraînement. L'erreur calculée par les données non entraînées c'est l'erreur de test. Le modèle est optimisé pour avoir le meilleur résultat. L'objectif de cette optimisation consiste à minimiser les deux erreurs, l'erreur de l'entraînement et l'erreur de test. Cette optimisation est appelée en anglais tarde off biais variance.

— Erreur de biais : considérée comme la différence entre la prédiction attendue de modèle et le réel.

— Erreur de variance : est définie comme la valeur attendue de l'erreur sur une nouvelle entrée.

— Sous-apprentissage (Underfitting) : le sous-apprentissage se produit lorsque le modèle n'est pas en mesure d'obtenir une valeur d'erreur suffisamment faible sur l'ensemble d'entraînement.

— Sur-apprentissage (Overfitting) : un sur-apprentissage se produit lorsque l'écart entre l'erreur d'apprentissage et l'erreur de test est grand.

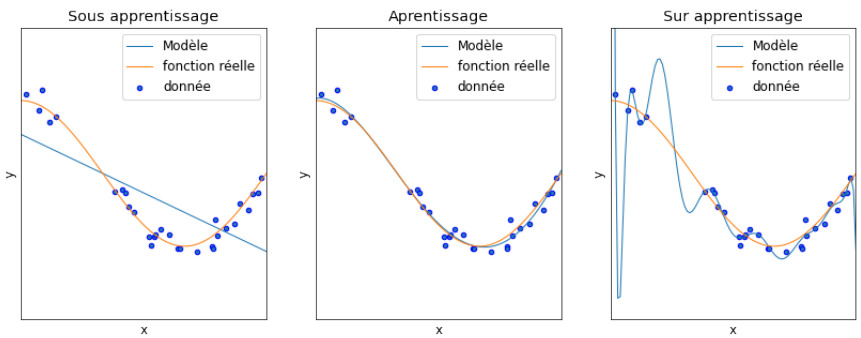

FIGURE 1.4 – Erreurs écarts-biais

1.5 Algorithmes d'apprentissage

1.5.1 Arbre de décision

L'arbre de décision, proposée en 1980 Quinlan (1980), est une structure hiérarchique et arborescente, où chaque nœud interne représente une condition sur un attribut et chaque branche représente un résultat du test. L'étiquette de classe est représentée par chaque nœud feuille (ou nœud terminal). Étant donné les données (D), les valeurs d'attribut des données sont testées par rapport à l'arbre de décision, par l'utilisation des tests condition de la racine à un nœud feuille qui contient la prédiction de classe pour les données. L'arbre de décision est facile à construire par rapport aux autres méthodes de classifications. Parmi les algorithmes d'arbre de décision, on trouve C4.5 (Quinlan, 1993), ID3 (Deisenroth *et al.*, 2020), CART (Breiman *et al.*, 1984).

Algorihtme 1 Principe d'arbre de décision (Han *et al.*, 2011)

Input : Partition de données, D, qui est un ensemble de tuples d'apprentissage et leurs étiquettes de classe associées ;
Liste d'attributs, l'ensemble des attributs candidats ;
Méthode de sélection d'attributs, une procédure pour déterminer le critère de fractionnement qui «mieux» partitionne les tuples de données en classes individuelles. Ce critère se compose d'un attribut de fractionnement et, éventuellement, d'un point de fractionnement ou d'un sous-ensemble de fractionnement.
Output : Un arbre de décision.
1.créer un nœud N
2. **si** *les tuples en D sont tous de la même classe)* **alors**
  retourner N comme un nœud feuille étiqueté avec la classe C
3.**si** *la liste d'attributs est vide* **alors**
  retourner N comme un nœud feuille étiqueté avec la classe majoritaire en D
4. appliquer la méthode de sélection des attributs (D, liste d'attributs) pour trouver le «meilleur» critère de fractionnement ;
5.étiqueter le nœud N avec un critère de division ;
6. **si** *l'attribut de fractionnement a une valeur discrète et fractionnements multi autorisés* **alors**
  liste d'attributs ← liste d'attributs - fractionnement d'attribut ; // supprimer l'attribut de fractionnement
7.**pour** *chaque résultat j du critère de fractionnement* **faire**
  D j l'ensemble des tuples de données dans D satisfaisant le résultat j ; // une partition **si** *D j est vide alors* **alors**
    attacher une feuille étiquetée avec la classe majoritaire en D au nœud N ;
  sinon
    attacher le nœud renvoyé par Générer l'arbre de décision (D j, liste d'attributs) au nœud N ;
8.retourner N ;

1.5.2 Apprentissage par ensemble

Contrairement aux approches d'apprentissage qui tentent de construire un seul modèle à partir des données, les méthodes d'apprentissage par ensemble essaient de construire un ensemble de modèles et de les combiner (Zhou, 2012). L'apprentissage par ensemble est la

base de plusieurs techniques, par exemple Bagging, Booting et Foret aléatoire. Le principe de base de ce paradigme d'apprentissage est l'entraînement de plusieurs modèles pour résoudre un même problème.

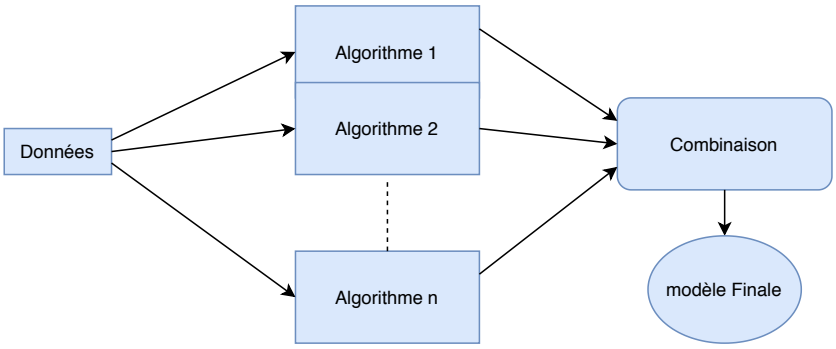

FIGURE 1.5 – Apprentissage par ensemble

1.5.2.1 Bootstrap

C'est un échantillon aléatoire des données prises avec remplacement (Efron & Tibshirani, 1986). Cela signifie qu'après un point de données sélectionné pour le sous-ensemble, il est toujours disponible pour une sélection ultérieure. L'échantillon de Bootstrap a la même taille que l'ensemble de données d'origine. Par conséquent, certains échantillons seront représentés plusieurs fois dans l'échantillon tandis que d'autres ne seront pas sélectionnés. Les échantillons non sélectionnés sont généralement appelés échantillons hors du sac (out-of-bag). Pour une itération donnée de rééchantillonnage Bootstrap, un modèle est construit sur les échantillons sélectionnés, est utilisé pour prédire les échantillons hors du sac (out-of-bag). La figure 1.6 montre la procédure à suivre dans le Boostrap. (D) est l'ensemble des données{X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12}

— Prendre des échantillons des données d1, d2, dn.

— Pour chacun des échantillons applique l'algorithme Algorithme 1, Algorithme 2 et Algorithme N.

— Chaque algorithme retourne un modèle Y1,Y2,Yn

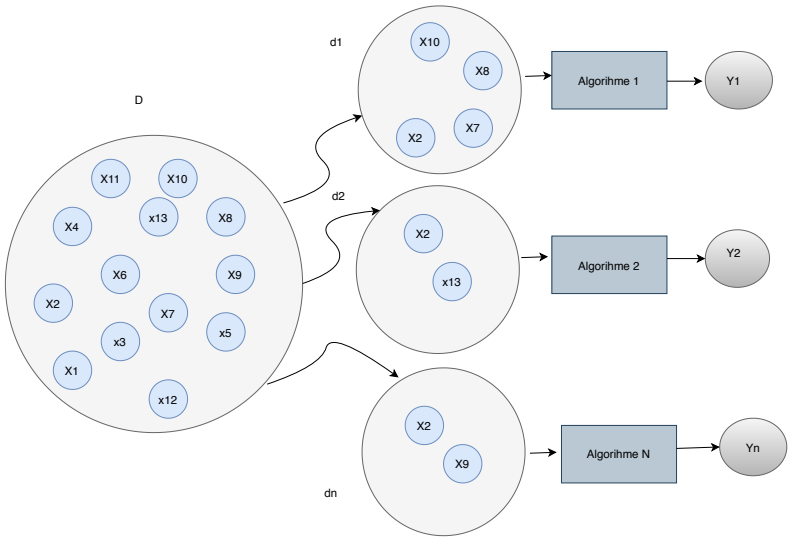

FIGURE 1.6 – Les étapes de Boostrap

1.5.2.2 Bagging

Le modèle à base d'arbre de décision, discuté en 1.5.1, à une variance élevée. Pour améliorer la qualité de modèle, il faut appliquer le Bootstrap afin d'améliorer l'estimation ou la prédiction de l'arbre. Supposant que $D = \{(x_1, y_1), (x_2, y_2), ..(x_n, y_n)\}$ pour prédire la fonction $f(x)$, Bagging fait la moyenne de cette prédiction et une collection d'échantillons Bootstrap qui réduise la variance. Pour chaque échantillon (bag) D, tel que $D = \{D^1, D^2, ...D^m\}$ le Baggin estime :

$$f(x)_{bag} = \frac{1}{m} \sum_{i=1}^{i=m} f^i(x) \tag{1.8}$$

1.5.2.3 Boosting

Le Bagging algorithme implique la création de plusieurs copies de l'ensemble de données d'apprentissage à l'aide du Bootstrap, et l'ajustement d'un arbre de décision distincte à chaque copie. Puis la combinaison de tous les arbres afin de créer un modèle prédictif unique. Notamment. Chaque arbre est construit sur un ensemble de données , indépendant des autres arbres. Le boosting fonction de manière similaire, sauf que les arbres sont construits d'une manière séquentielle : chaque arbre est construit en utilisant des informations provenant d'arbres déjà construits. Boosting n'implique pas d'échantillonnage Bootstrap ; à la place, chaque arbre est adapté à une version modifiée de l'ensemble de données d'origine.

1.5.2.4 Forêts aléatoires

Basé sur des idées d'agrégation de modèles, la forêt aléatoire (Random forests) est un algorithme populaire très efficace pour les problèmes de classification et de régression introduit par Breiman Breiman (2001). Elle appartient à la famille des méthodes d'ensemble, apparaissant dans la machine learning à la fin des années 90 (Dietterich, 1998, 2000). L'algorithme de foret aléatoire construit des ensembles des arbres de décision à partir des données. La forêt aléatoire est la modification substantielle de Boostrap qui constitue une grande collection d'arbres décorrélés, puis les calcule. Sur de nombreux problèmes, les performances de forêt aléatoire sont très similaires à celles de Boosting et plus simple à former et à régler. En conséquence, la forêt aléatoire devient populaire.

Algorihtme 2 Forêt aléatoire pour régression ou classification

Input : D
Output : T : Foret aléatoire
for *For b = 1 to B :* **do**
échantillon bootstrap D^i de taille N donnée d'entraînement Créerer un arbre de forêt aléatoire T_i i donnée Boostrap, en répétant récursivement les étapes suivantes pour chaque nœud terminal de l'arborescence, jusqu'à ce que la taille minimale du nœud n min soit atteinte.
Sélectionnez m variables au hasard parmi les p variables
Choisissez la meilleure variable / point de partage parmi les m
Fractionner le nœud en deux nœuds
Sortie de l'ensemble des arbres T_i
Pour faire une prédiction à un nouveau point $f(x)_{bag} = \frac{1}{m} \sum_{i=1}^{i=m} f^i(x)$:
Classification : Supposant $c(x)_{bag}$ être la prédiction de classe de la b foret aléatoire arbre. puis $f(x)_{bag}$ sélectionner le vote majoritaire $(f(x)_{bag})$

1.5.3 LightGBM

En anglais A Highly Efficient Gradient Boosting Decision Tree, propose un modèle attractif pour l'analyse des données. Ce modèle introduit par les auteurs afin de résoudre le problème d'efficacité et d'évolutivité de gradient Boosting d'arbre de décision (GBDT) dans le cas de grandes dimensions et de grands ensembles de données. Afin de résoudre ces problèmes, Ke *et al.* (2017) a proposé deux techniques, l'échantillonnage unilatéral basé sur un gradient (Gradient-based One-Side Sampling GOSS) et le regroupement exclusif de fonctionnalités (Exclusive Feature Bundling EFB). La première technique utilisée pour exclure certaines instances de données avec de petits gradients. Dans la deuxième technique, ils estiment le gain d'informations, Ke *et al.* (2017) appliqué LightGBM sur plusieurs ensembles de données publics. En conséquence, la vitesse d'apprentissage jusqu'à près de 20 fois par rapport au GBDT conventionnel avec environ de même précision. LightGBM utilisé mieux en pratique et plus rapide pour les données volumineuses (Ma *et al.*, 2018). Il s'agit donc d'un apprentissage automatique pour l'analyse des données massive.

1.5.3.1 Formulation mathématique de LightGBM

Supposons $D = \{(x_1, y_1), (x_2, y_2), ...(x_i, y_i), (x_{i+1}, y_{i+i}), ...(x_m, y_m)\}$ un ensemble de données d'entrainement, où x_i, $i \in [1..m]$, sont les échantillons de données et y_i sont des valeurs d'étiquettes de classe. Dans l'arbre de décision Gradient Boosting, l'objectif d'optimisation est de réduire la fonction de perte. Supposant $f(x)$ une fonction et $\hat{f}$ la fonction représente la fonction estimée entre y et $f(x)$, $L(y, f(x))$ (Chen *et al.*, 2019; Sun *et al.*, 2018) avec :

$$\hat{f}(x) = \underset{E}{\operatorname{argmin}} E_{x,y} L(y, f(x)). \tag{1.9}$$

LightGBM générer (t) arbres de décision, (t) est le nombre d'arbres de décision (Sun *et al.*, 2018) avec :

$$f_T(x) = \sum_{T=1}^{T=t} f_t(x) \tag{1.10}$$

Ke *et al.* (2017) a observé que la précision de l'arbre de décision d'optimisation du gradient ne suffit pas dans le cas d'un grand nombre d'échantillons ou d'une grande dimension de caractéristique. Avant, les chercheurs ont proposé l'échantillonnage unilatéral basé sur un gradient (GOSS) et le regroupement exclusif de fonctionnalités (EFB). GOSS effectue uniquement le petit gradient aléatoire. Il trie d'abord $a \times 100$ %, où (a) est le taux d'échantillonnage des données à gradient élevé. Le au hasard $b \times 100$ % où (b) est le rapport d'échantillonnage des petites données de gradient. L'algorithme EFB de rôle peut optimiser le calcul en évitant des valeurs de caractéristique nulles.

1.5.4 Machines à vecteurs de support

Les machines à vecteurs de support SVM (Support Vector Machine) c'est la construction d'un hyperplan optimal, qui peut être utilisé pour la classification des problèmes linéairement séparables. L'hyperplan optimal maximise la marge. La distance de l'hyperplan au point le plus proche de chaque classe. L'objectif principal de SVM est de maximiser la marge afin qu'il puisse classer correctement les modèles donnés, c'est-à-dire que plus la taille de la marge est plus correcte (Vapnik, 1995). Les Machines à vecteurs de support (SVM), conçus à l'origine pour la classification binaire, sont des classificateurs à grande marge (Zhang, 2001).

La marge est définie comme la distance minimale entre les instances de différentes classes et l'hyperplan de classification. Un hyperplan est défini par la fonction $w^t x_i + b$. La distance euclidienne d'une instance (x_i) à l'hyperplan est donnée par l'équation suivante :

$$\frac{w^t x_i + b}{\| w \|} \tag{1.11}$$

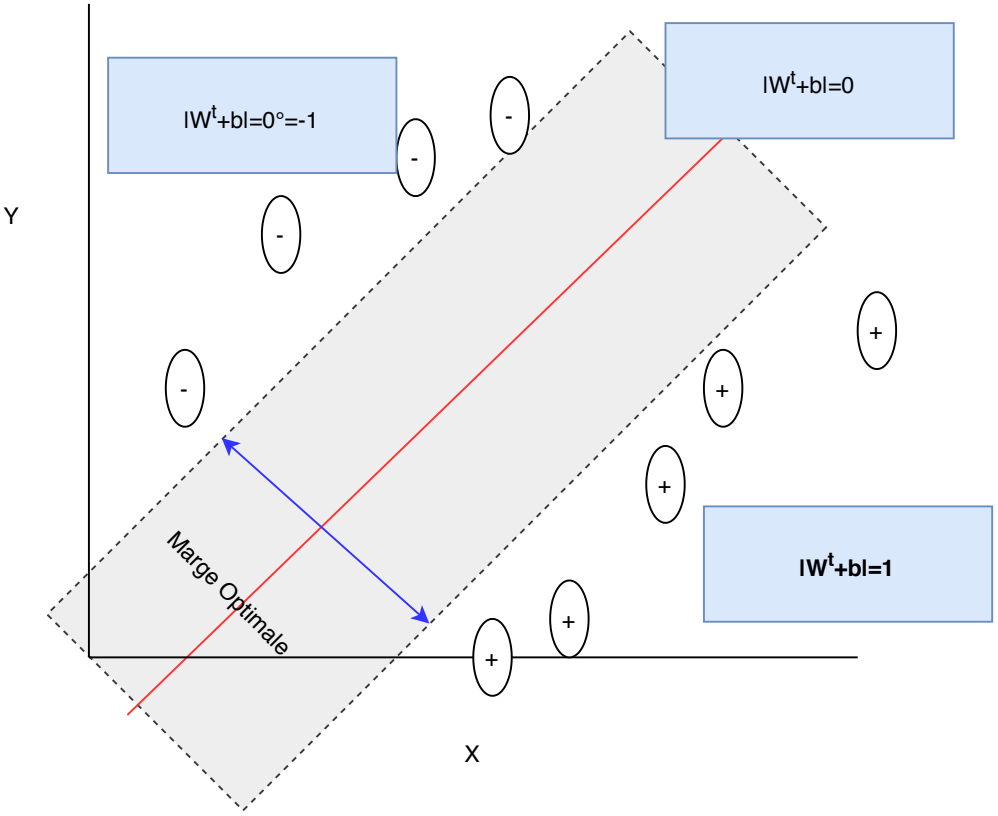

FIGURE 1.7 – Apprentissage par les machines à vecteurs de support

1.5.5 Modèle de bayse

Le modèle Naive Bayes est un modèle d'apprentissage automatique probabiliste. La forme de probabilité est une extension de la théorie de probabilité conditionnelle citée dans l'équation 1.4. On a deux formules de bayes :

$$p(B/A) = \frac{p(A/B)p(B)}{p(A)} \tag{1.12}$$

$$p(B_i/A) = \frac{p(A/B_i) \times p(B_i)}{\sum_i p(B_i/A)p(B_i)} \tag{1.13}$$

1.5.6 K plus proche voisin

K plus proche voisin (KPPV), est un algorithme utilisé pour la classification. Il est très facile à utiliser et il donne des bons résultats. De plus, l'algorithme KPPV ne nécessite aucune connaissance préalable concernant l'ensemble de données pour la classification. Il permet d'effectuer une classification sur la base de la similarité. Comme son nom l'indique, pour la classification d'un nouveau donné, il recherche les (k) donnés les plus proches. La procédure

de classification dans cet algorithme commence par un ensemble de données constitué par leurs attributs. l'algorithme KPPV calcule les (k) points de données les plus proches de lui dans l'espace à (n) dimensions. Pour trouver les (k) points de données les plus proches, diverses mesures de distance sont utilisées, par exemple les mesures citées dans l'équation 1.1

1.5.7 k-moyenne

Proposé par MacQueen *et al.* (1967), k-moyenne (k-mean) est la méthode la plus célèbre d'analyse. Il s'agit d'un processus de regroupement des objets en classe uniforme, appelé cluster, en fonction de similarité entre ces objets. Tout d'abord, choisissiez des points pour représenter le cluster initial. Deuxièmement, rassembler les points d'échantillon restants à leurs points conformément au critère de la distance minimale (par exemple la on suivant les formules 1.1). Puis nous obtiendrons la classification initiale, et si la classification est déraisonnable, itérer de manière répétitive jusqu'à nous obtenons une classification raisonnable.

Algorihtme 3 Principe de l'algorithme k-moyenne

Input : D
Output : K centroïdes finaux
1. Sélectionnez le point K comme centroïdes initiaux
2.**répéter**
    K points en affectant tous les points au centroïde le plus proche
    Recalculer le centroïde de chaque cluster
jusqu'à
    Le centroïde ne change pas

1.6 Approche d'apprentissage profonde

Le Deep Learning a connu un grand succès dans plusieurs applications d'apprentissage automatique, telles que l'analyse d'images, l'exploration de texte ou la reconnaissance vocale. Il est composé de représentations et de fonctionnalités à plusieurs niveaux dans des architectures hiérarchiques. L'algorithme d'apprentissage profond peut être appliqué aux problèmes de classification et de régression. Divers modèles d'apprentissage profond ont été présentés, tels que le réseau de neurones convolutionnels, l'auto-encodeur empilé, le réseau de croyances profondes et le réseau de neurones récurrent.

1.6.1 Réseau de propagation vers l'Avant

Les réseaux de perceptrons multicouches MLP (Multi Layer Perceptron), sont les modèles d'apprentissage profond. Il s'agit d'une fonction d'approximation f. Ce modèle est appelé en anglais feedforward neural network, car les informations transitent par la fonction évaluée à partir de (x), par les calculs intermédiaires utilisés pour définir f, et finalement par la sortie y. Ils sont généralement représentés par de nombreuses fonctions différentes. Le modèle est associé à un graphique acyclique dirigé décrivant comment les fonctions sont composées. Par exemple, nous pourrions avoir trois fonctions f^1, f^2 et f^3 connectées en chaîne, pour former y=f(x)=$f^3(f^2(f^1(x)))$

1.6.2 Réseau de neurones convolutifs

En anglais CNN (Convolutional neural network). Ils sont composés de trois couches, une couche convolutionnelle (convolutional layer), une couche de mise en commun (pooling layer) et une couche entièrement connectée (fully-connected layer) (Simonyan & Zisserman, 2014). Les réseaux de neurones convolutifs ou CNN, sont des types spécialisés de réseau de neurones pour le traitement de données ayant une topologie de type grille connue. Les exemples incluent les données de séries chronologiques, qui peuvent être considérées comme un vecteur prenant des échantillons à intervalles de temps réguliers, et les données d'image qui peut être considérée comme une matrice de pixels. Les réseaux convolutifs ont connu un énorme succès dans les applications pratiques. Le nom de «réseau neuronal convolutif» indique que le réseau utilise une opération mathématique appelée convolution. La convolution est un type d'opération linéaire spécialisé. Les réseaux sont simplement des réseaux de neurones qui utilisent la convolution à la place de la multiplication matricielle générale dans au moins une de leurs couches.

1.6.3 Réseau neuronal récurrent

Convient aux problèmes de données de séries chronologiques. Il apprend les caractéristiques des données de série dans la mémoire dans l'entrée précédente (Graves, 2013). Le fonctionnement de réseau décrit par l'équitation suivante :

$$H^{(e)} = f(h^{(e-1)}, x^{(e-1)}; \theta) \tag{1.14}$$

Où (H) est la couche cachée, la situation dans un état décrit par e, (x) la donnée et (θ) paramètre.

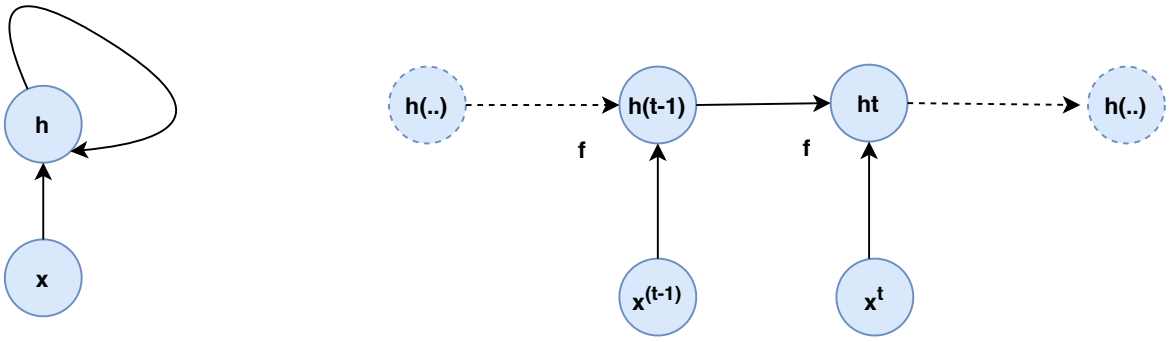

FIGURE 1.8 – Réseau neuronal récurent (Goodfellow *et al.*, 2016)

1.6.4 Encodeur-Décodeur

Auto-encoder, composé d'empilement de plusieurs auto-encodeurs. Il a deux étages, un étage de codage et un étage de décodage (Gehring *et al.*, 2013). La première étape pour apprendre une séquence et le décodage pour générer une séquence, alors le type d'apprentissage est séquence à séquence.

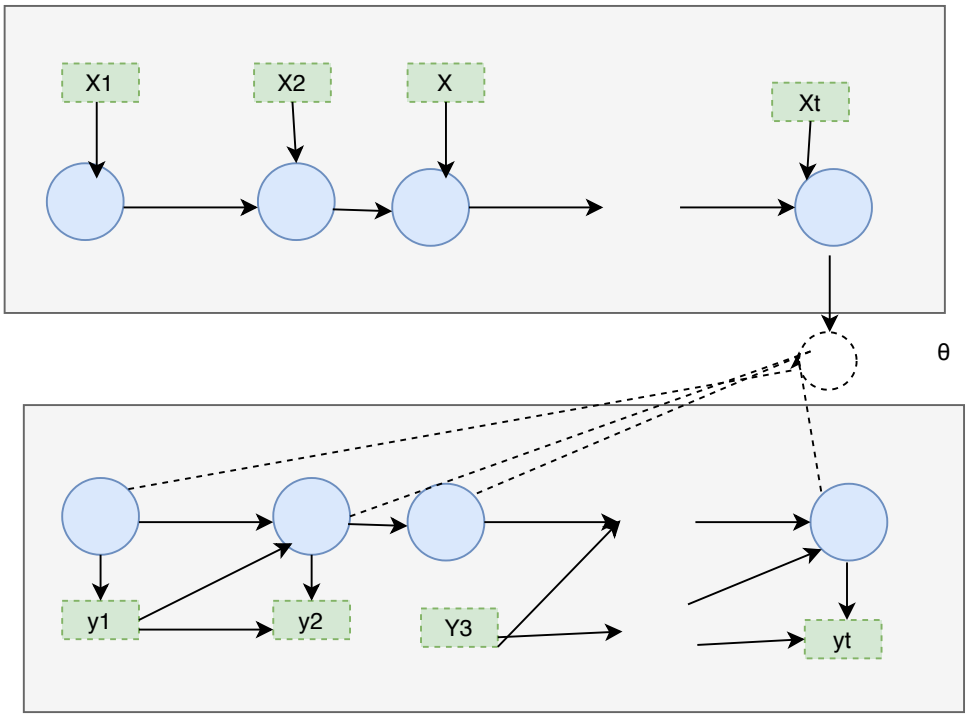

FIGURE 1.9 – Encodeur décodeur réseau de neurone (Cho *et al.*, 2014)

1.6.5 Réseau de croyances profondes

Les réseaux de croyance, en anglais Deep belief network, sont des modèles génératifs construits par plusieurs couches avec des variables latentes. Les machines Boltzmann restreintes sont un exemple de ces réseaux. Ils sont composés de couches visibles et cachées (Larochelle *et al.*, 2012).

1.6.6 Réseau génératif adversaire

Yann LeCun vu que le réseau génératif adversaire est une idée la plus intéressante dans les 10 dernières années en Machine Learning (Campesato, 2020). Le principe de réseau générative adversaire est l'estimation d'un modèle génératif en se basant sur entraînement de deux modèles simultanément. Un modèle génératif qui estime la distribution de données (G), et un modèle discriminant qui calcule de la provenait des données d'entraînement plutôt que de (G). La procédure d'entraînement pour (G) est de maximiser la probabilité que (D) faisant des erreurs. La fonction d'entraînement est une fonction MinMax qui calcule le coût (Goodfellow *et al.*, 2014) :

$$J^{(D)}(\theta^{(D)},\theta^{(G)}) = -\frac{1}{2}\mathbb{E}_{x\sim pdata}\log(D(x)) - \frac{1}{2}\mathbb{E}_{z\sim data}\log(1-D(G(x))) \tag{1.15}$$

Les avantages de GAN (Generative Adversarial Network) sont :

- La capacité de traitement des données à grande échelle.
- Très efficace pour l'apprentissage par renforcement.
- Permettent aux Machines Learning de fonctionner avec des sorties multimodales.
- La génération de bons échantillons.

La figure 1.10 décrit le framework de cette approche :

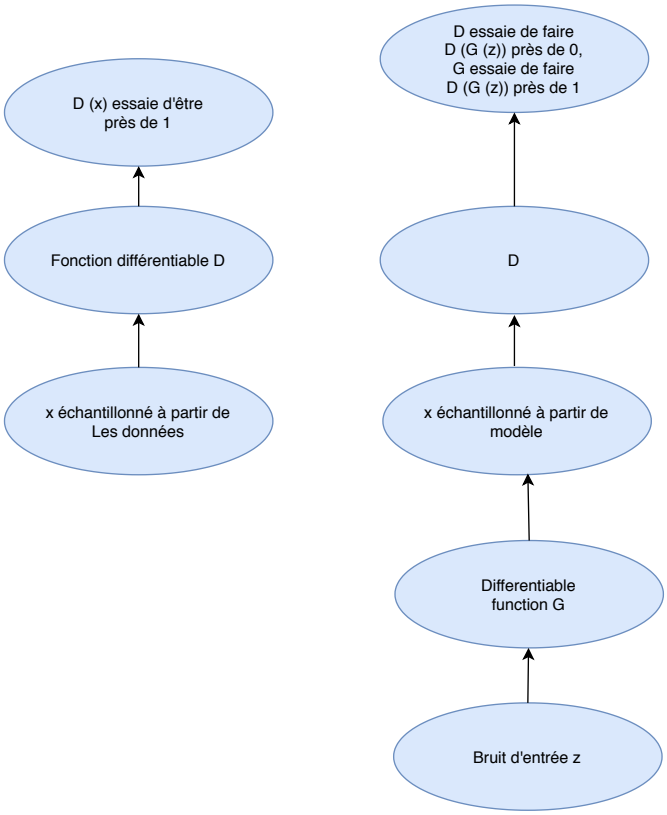

FIGURE 1.10 – Réseau génératif adversaire framework (Goodfellow, 2016)

1.7 Optimisation d'apprentissage profond

L'optimisation d'apprentissage profond est la recherche de meilleure solution de modèle. Formellement, il s'agit de trouver le paramètre (θ) de telle sorte que la fonction de coût $J(\theta)$ est le minimum (Goodfellow, 2016).

$$J(\theta) = \mathbb{E}_{(x,y)\sim\hat{p}data} L((x;\theta), y) \tag{1.16}$$

Où (L) est la fonction de perte ; $f(x;\theta)$ est la sortie prévue lorsque l'entrée est (x) ; et $(\hat{p}data)$ est la distribution empirique.

1.7.1 Descente de gradient stochastique

L'algorithme de descente de gradient stochastique est le plus connu parmi les algorithmes d'optimisation, dont la fonction objective décomposée en somme d'ensemble d'apprentissages.

Algorihtme 4 Algorithme de descente de gradient stochastique (Goodfellow *et al.*, 2016)

Input : ϵ taux d'apprentissage, θ paramètre initiale, r le seuil de l'erreur minimal
Output : θ
while $(r < \theta)$ **do**
1.Échantillon d'un mini-lot d'exemples m de l'ensemble d'apprentissage tell que $\{x^1,...,x^m\}$ et y^i.
2.Calculer l'estimation du gradient. $\hat{g} \leftarrow \frac{1}{m}\nabla_\theta \sum_i Lf(x^i;\theta), y^i)$.
3.Mise à jour : $\theta \leftarrow \theta - \epsilon\hat{g}$

Polyak (1964) a proposé la méthode des moments afin d'accélérer le taux d'apprentissage. Formellement, le moment est une variable de rapidité (v) qui calcule le paramètre (θ) rapidement. L'algorithme de descente de gradient stochastique avec moment est donné comme suit :

Algorihtme 5 Algorithme descente de gradient stochastique avec moment

Input : ϵ taux d'apprentissage, θ paramètre initiale, (r) le seuil de l'erreur minimal, rapiditéα
Output : θ, rapidité v
while $(r < \theta)$ **do**
1.Échantillon d'un mini-lot d'exemples m de l'ensemble d'apprentissage tell que $\{x^1, ..., x^m\}$ et y^i.
2.Calculer l'estimation du gradient. $\hat{g} \leftarrow \frac{1}{m}\nabla_\theta \sum_i Lf(x^i;\theta), y^i)$.
3.Mise à jour : $\theta \leftarrow \theta - \epsilon\hat{g}$
4.Mise à jour : $v \leftarrow \alpha v - \epsilon\hat{g}$
5.Mise à jour : $\theta + v$

1.7.2 Taux d'apprentissage adaptatif

Le principe d'adaptation de taux d'apprentissage est basé sur l'idée de Jacobs (1988) dans son algorithme Delta Bar. Si la dérivée partielle de la perte, par rapport à un paramètre de modèle donné, reste dans le même signe, alors le taux d'apprentissage devrait augmenter. Si la dérivée partielle par rapport à ce paramètre a un changement significatif, alors le taux d'apprentissage devrait diminuer. Cet algorithme est utile pour lot complet (full batch), autres algorithmes AdaGrad et RMSProp sont appliqués sur Mini lot. L'algorithme AdaGrad change le taux d'apprentissage par l'échelle inversement proportionnelle aux racines carrées de la somme de toutes leurs valeurs historiques (Duchi *et al.*, 2011).

Algorihtme 6 Descente de gradient stochastique avec taux d'apprentissage adaptatif

Input : ϵ ,taux d'apprentissage, θ paramètre initiale,r le seuil minimal de l'erreur, σ valeur constante minimum
Output : θ
1.Initialiser la variable d'accumulation de gradient $\omega = 0$
while $(r < \theta)$ **do**
1.Échantillon d'un mini-lot d'exemples m de l'ensemble d'apprentissage tell que $\{x^1, ..., x^m\}$ et y^i.
2.Calculer l'estimation du gradient. $\hat{g} \leftarrow \frac{1}{m}\nabla_\theta \sum_i Lf(x^i;\theta), y^i)$.
3.Accumuler un gradient carré :$\omega \leftarrow \omega + g \odot g$
4.Mise à jour : $\triangle\theta \leftarrow \frac{\epsilon}{\sigma+\sqrt{\omega}} \odot g$
5.Mise a jour .$\theta \leftarrow \theta + \triangle\theta$

1.8 Conclusion

Dans ce chapitre, nous avons présenté l'objectif principal de Data Mining est la modélisation optimale des problèmes par l'utilisation de certains algorithmes en vue de résoudre un problème lié aux divers champs d'applications et domaine scientifique, économique, industrielle gouvernementale, etc.

En premier lieu, nous avons présenté les principaux concepts de Data Mining et leurs relations avec les autres concepts, notamment la découverte de connaissance dans les bases de

données et machine Learning. Nous avons donné la description formelle de Data Mining dans deux axes, données et apprentissage. Dans le premier axe, nous avons présenté les différents types de données possibles tels que : vecteur, matrice et tenseur. Dans l'axe de l'apprentissage, nous avons rappelé les algorithmes et les différentes stratégies pour l'estimation de modèle tel que l'arbre de décision, Bagging, Boosting, etc. Ensuite, nous avons fait un rappel général de principales connaissances d'apprentissage profond, tel que le réseau de convolution et récurrent. Enfin, nous avons clôturé le chapitre par les algorithmes d'optimisation. Le prochain chapitre est consacré pour le Big data.

Chapitre 2

Big Data

Sommaire

2.1 Introduction

Au cours de ces dernières années, la croissance de données est rapide à cause de la nécessité d'échange de l'information entre les utilisateurs. Par exemple, dans les réseaux sociaux, le nombre de textes saisis, les vidéos téléchargés, etc., est très élevé. En outre, les senseurs de technologie d'internet des objets qui acquiert les données en temps réel et les appareils mobiles. Le nombre des objets connecté prévu en 2025 est environ de 32,6 millions d'objets et 50 millions en 2030 (Statista1, 2019). Facebook génère 4 pétaoctets de données par jour (Kinsta, 2020). Le traitement de ces données est devenu complexe et coûteux. De plus, ils nécessitent des nouveaux algorithmes qui traitent des grandes masses de données pour construire un modèle. Les sources de données sont multiples en temps réel et la quantité de données qui nous entoure augmente de façon exponentielle. Pour répondre aux nouvelles exigences des systèmes, le concept de Big Data est apparu dans le but de création d'une cadre théorique permettant de répondre aux nouvelles spécifications des systèmes d'une façon crédible, équitable, efficace et économique. Dans ce chapitre, nous allons présenter les concepts relatifs au Big Data. Nous commençons par les concepts de base de Big Data. En deuxième lieu, nous présentons différentes architectures et ces classifications. Enfin, nous terminerons par quelques contributions sur les systèmes Big Data.

2.2 Concepts de Base de Big Data

2.2.1 Définition

Dans la littérature, diverses définitions de Big Data sont proposées. Kim *et al.* (2014) définirent le Big Data par l'énorme volume données collectées à partir d'une grande variété de sources et qui sont trop importantes, brutes ou non structurées pour être analysées par les techniques de base de données conventionnelles. D'autres chercheurs définit le Big Data en termes d'outils, des processus et des procédures qui permettent à une organisation de créer, de manipuler, gérer, installer et stoker une grande quantité de données (Yin *et al.*, 2013).

2.2.2 XV de Big Data

Afin de décrire les caractéristiques de Big Data, les chercheurs utilisent l'acronyme 3V d'autres utilisent 5V ou 6V. Les caractéristiques XV sont des propriétés des données pour être Big Data : Volume (Assunção *et al.*, 2015; El Kassabi *et al.*, 2018; Uddin *et al.*, 2014), Variété (Assunção *et al.*, 2015; El Kassabi *et al.*, 2018; Uddin *et al.*, 2014), Vitesse (Assunção *et al.*, 2015; El Kassabi *et al.*, 2018; Uddin *et al.*, 2014), Véracité(Assunção *et al.*, 2015; El Kassabi *et al.*, 2018; Uddin *et al.*, 2014), Volatilité (Uddin *et al.*, 2014), Valeur(Assunção *et al.*, 2015; Uddin *et al.*, 2014)

— Volume : la quantité massive de données générées par les utilisateurs, les capteurs, les réseaux, les interactions humaines et les médias sociaux.

— Variété : les données varient en fonction de leur type qui peut être structuré, semi-structurées ou non structurées. Les données structurées sont des données simple à traiter et à manipulée caractérisées par un type prédéfini, par exemple le cas de base de données avec le langage SQL. Les données non structurées, sont les données les plus existés dans les systèmes Big Data, dont les données ne suivirent aucune structure prédéfinie. Ils peuvent être, une publication, commentaire écrit par un facebookeur ou un message instantané sous une application de messagerie. La non-structuration de

données fait une grande difficulté au niveau du traitement de données et par conséquent au niveau d'exploitation de données. Au-delà des données structurées et non structurées, il existe une troisième catégorie, qui est essentiellement le mélange des deux. Le type de données défini comme des données semi-structurées présente certaines caractéristiques déterminantes ou cohérentes, mais ne se conforme pas à une structure aussi stricte que ce que l'on pourrait, attendre d'une base de données relationnelle. Par conséquent, il existe certaines propriétés organisationnelles telles que des balises sémantiques ou des métadonnées pour faciliter l'organisation, mais les données restent fluides.

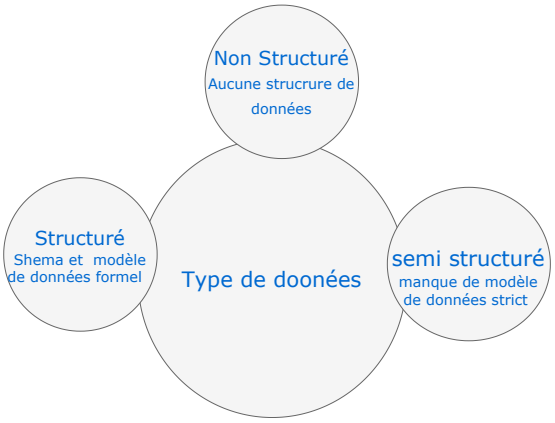

FIGURE 2.1 – Types de données

— Vélocité : c'est la vitesse de génération des données. Par exemple la génération des données en temps réel des senseurs des climats, la génération des données dans les réseaux sociaux et l'utilisation des appareils mobiles par les utilisateurs.

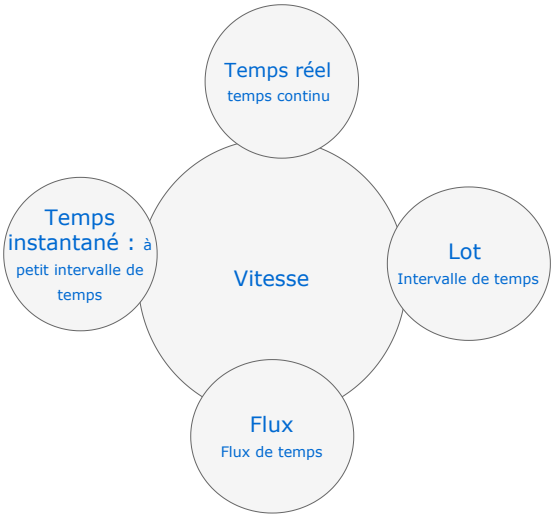

FIGURE 2.2 – Vitesse de données

— Véracité : c'est la capacité d'évaluer l'exactitude et la précision des données, dont le contexte de prise de décision.

— Volatilité : la volatilité indique la durée pendant laquelle les données resteront valables et méritent d'être stockées.

— Valeur : décrit comment une entreprise exploite les données. Par exemple si elles permettent de mieux comprendre les clients, de cibler en conséquence, d'optimiser les

processus et d'améliorer les performances des machines ou de l'entreprise, comprendre le potentiel, ainsi que les caractéristiques les plus difficiles.

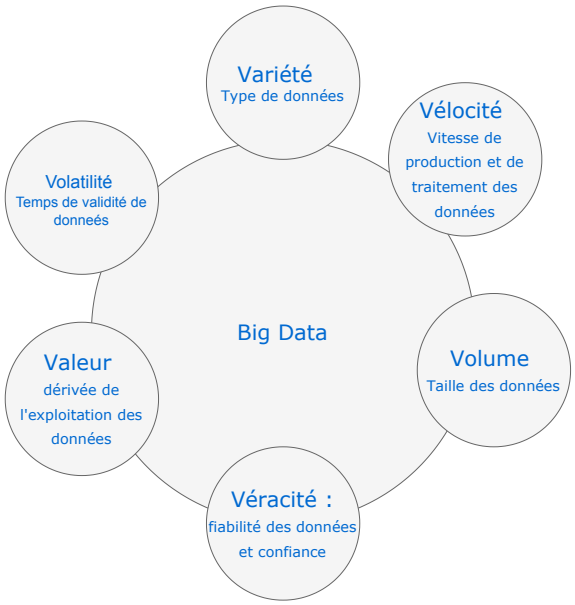

FIGURE 2.3 – 6 V Big data

2.3 Écosystème Big data

Plusieurs travaux dans l'industrie développent des systèmes Big Data. Dans cette section, nous présentons les systèmes Big Data , et les fameux écosystèmes et les servies Hadoop, Apach Spark, Hive, Pig, Flum, Sqoop, Oozi, Mahout, Tez, Flink, Storm, Casandra, Kafka et Zookepeer.

2.3.1 Hadoop

Apache Hadoop (High-availability distributed object-oriented platform), est une plate-forme Java qui permet le stockage, la gestion et le traitement des Big Data. Il s'agit d'un serveur apache fournie un environnement de calcul distribué. Hadoop est devenu une véritable plate-forme de stockage, de traitement et de gestion de centaines de téraoctets et même des pétaoctets de données. Les composants de Hadoop sont : les fichiers HDFS (Hadoop ditributed file system), Job Tracker/Task Tracker, le paradigm Map Reduce, Hadoop YARN et Appach Spark.

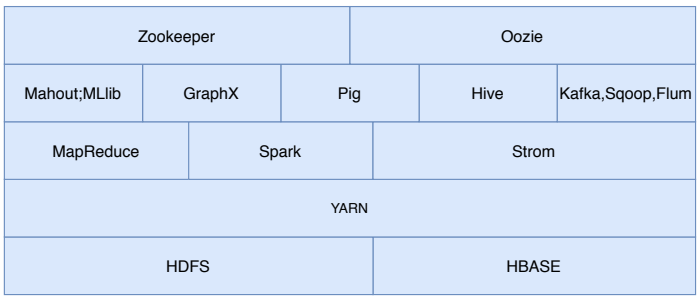

FIGURE 2.4 – Les composantes d'Hadoop (White, 2012; Landset *et al.*, 2015)

2.3.1.1 Les fichiers HDFS

Le HDFS est un système de fichiers distribué et évolutif conçu pour stocker une grande quantité de données dans des serveurs/clusters distribués. Le HDFS partition les données en petits blocs de données (la taille de bloc par défaut est de 64 Mo) et distribue les blocs de données sur son cluster.

— Nœud de nom (Name Node) : il assure la gestion de toutes les métadonnées du système de fichiers et manipule le fonctionnement de l'espace de noms du système de fichiers, il est le responsable des réplications en blocs (Shvachko *et al.*, 2010; Mackey *et al.*, 2009).

— Nœud de données (Data node) : les fichiers sont divisés en blocs de taille fixe et stockés sur des nœuds de données. Les blocs de données sont répliqués pour la tolérance aux fautes et l'accès rapide, les nœuds de données, de temps en temps envoient des messages au nœud du nom (Lam, 2010; Mackey *et al.*, 2009)

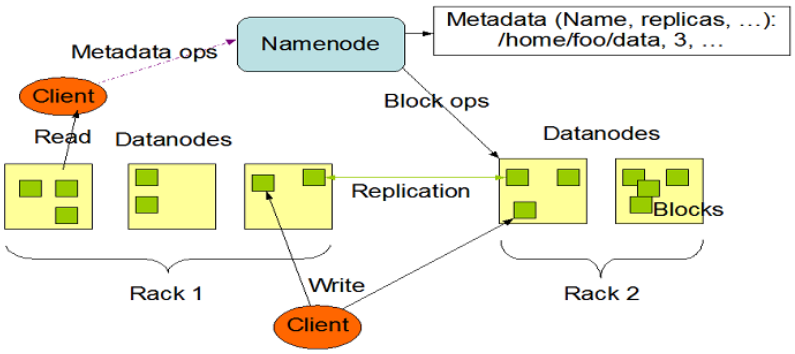

FIGURE 2.5 – Les composantes du fichier HDFS (Borthakur *et al.*, 2008)

2.3.1.2 Job Tracker/Task Tracker

Job Tracker communique le nom du nœud pour déterminer l'emplacement des données. Le JobTracker, responsable du plan d'exécution de chaque tâche, surveille l'exécution des tâches. En cas d'échec ou de succès d'une tâche, le JobTracker utilise d'autres nœuds pour achever toutes les tâches (Lam, 2010; Martha *et al.*, 2013). Le TaskTracker est responsable de l'exécution de chaque tâche, il communique avec le JobTracker par message pour assurer l'exécution Map Reduce (Perera, 2013).

2.3.1.3 Le paradigme Map Reduce

Les primitives Map et Reduce (Dean & Ghemawat, 2010) mettent en œuvre un traitement parallèle. Elles se composent de deux fonctions, Map et Reduce. La fonction Map prend un ensemble de données et le convertit en un autre ensemble de données, elle prend une paire de clés (key, pair) et émet (key, pair) dans la fonction Reduce. La fonction Reduce est constituée d'un maître appelé Jobtracker, et d'un ensemble d'esclaves serveur appelé TaskTracker (Martha *et al.*, 2013; Shafer *et al.*, 2010).

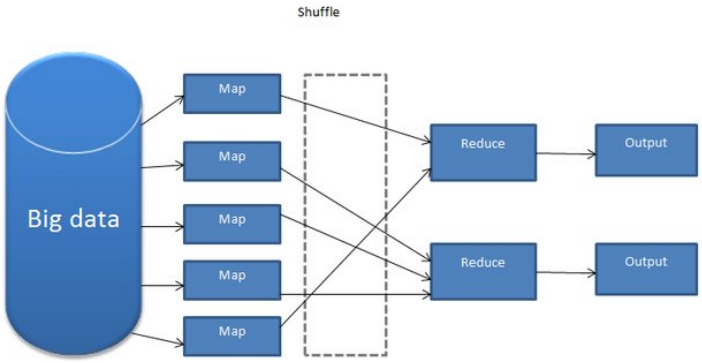

FIGURE 2.6 – Hadoop MapReduce (Borthakur *et al.*, 2008)

2.3.1.4 Hadoop YARN

Hadoop (YARN Yet Another Ressource Negotiator). Il est utilisé essentiellement pour séparer les composants de traitement et le processus de gestion des ressources. YARN fournir une plate-forme avantageuse pour les applications qui ne nécessite pas des tâches. MapReduce(Vavilapalli *et al.*, 2013).

2.3.2 Apach Spark

Apache Spark est un framework de cluster de calcule open source pour le traitement des données en temps réel. La principale caractéristique d'Apache Spark est son calcul en mémoire qui augmente la vitesse de traitement d'une application. Spark fournit une interface pour la programmation distribuée entière avec parallélisme des données implicites et tolérance aux pannes. Il permet de répondre aux exigences des applications par lots, les algorithmes itératifs, les requêtes interactives et en streaming. Les avantages de spark sont : la vitesse, le déploiement et le temps réel. Appach Spark est basé sur deux abstractions : ensemble de données distribuées RDD (Resilient Distributed Datasets) et Graphique acyclique dirigé DAG (Directed Acyclic Graph). RDD est la représentation de données et DAG est un graphe représente un ensemble de sommets et d'arêtes, où les sommets représentent les RDD et les arêtes représentent l'opération à appliquer sur les RDD. Les principaux composants de l'architecture de Spark sont :

— Le contexte de Sparc (Sparc contex) : est l'abstraction qui encapsule le cluster pour le nœud du pilot.

— Les gestionnaires de cluster (Cluster Manager) : allouent des ressources dans toutes les applications.

— Nœud de travail (Worker node) : gèrent les ressources dans une seule machine esclave et communique avec Cluster manager.

— Executeur (Executors) : sont les processus qui peuvent accomplir des tâches.

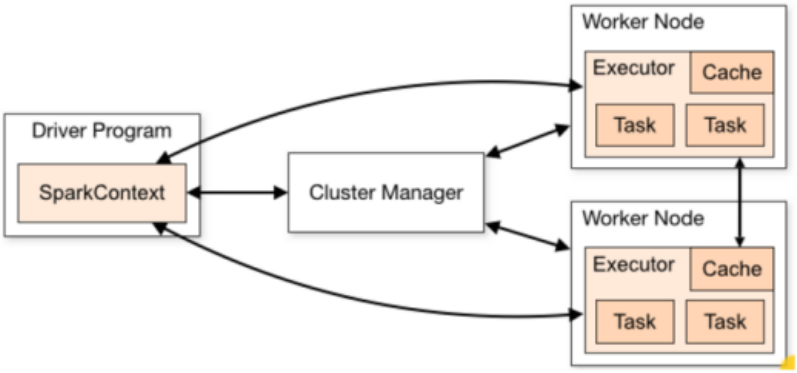

FIGURE 2.7 – Architecture du Spark (Karau *et al.*, 2015)

Spark constitue un environnement pour l'interaction avec les composantes suivantes :

— Spark Streaming : est la composante de Spark qui est utilisée pour traiter les données de streaming en temps réel. Il permet un traitement à haut débit et tolérant aux pannes des flux de données en temps réel.

— Spark SQL : est le module intègre le traitement relationnel avec l'API de programmation fonctionnelle de Spark. Leur rôle est l'interrogation des données via SQL.

— GraphX : l'API Spark pour les graphes et le calcul parallèle de graphes.

— MLlib : Une bibliothèque source libre qui offre aux utilisateurs le développement d'ensemble des machines learning par Scala R ou Python. MLlib est exécute les applications de machine learning plus rapide que Mahout.

— SparkR : Fournit une interface de programmation par le langage R.

2.3.3 Hive

Hive est un SQL like interface pour Hadoop. Hive permet la création, la mise à jour et la suppression de Big Data. En autre il fournit un moyen d'interrogation des grandes masses de données . Hive fonctionne sous le paradigme de Map reduce (Thusoo *et al.*, 2010b). Plusieurs entreprises utilisent Hive comme Facebook, Netflix et amazon.

2.3.4 Pig

Crée par Yahoo, Pig est un langage d'interrogation déclaratif de haut niveau comme le langage SQL. Pig exécuté des programmes Map Reduce et utilise Pig latin langage sous Hadoop système (Swarna & Ansari, 2017).

2.3.5 Flum

Un système à base d'agent fiable, de collection, d'agrégation et capable de transférer efficacement de grandes quantités de données de journaux (logs) provenant de nombreuses sources différentes vers un entrepôt de données centralisé. L'un des avantages de Flum est que ses données sont saisies et stockées en temps réel dans HBase pour être traitées. Flum utilisé pour la collection de données des réseaux sociaux Facebook, Twitter et le site du commerce électronique (Hoffman, 2013)

2.3.6 Sqoop

L'abréviation de deux mots en anglais "Sq" de Sql et "oop" de Hadoop. Un outil d'importation et d'exportation des grandes masses de données entre bases de données relationnelles et les fichiers distribués d'Hadoop. Sqoop utilise les connections java de base de données JDBC et l'interpréteur ligne de commande interface. Sqoop support le parallélisme entre cluster et les tâches Map réduce (Ting & Cecho, 2013)

2.3.7 Oozie

Un système de planification pour gérer l'ensemble d'application des utilisateurs (Job) dans un système distribué. Les applications peuvent être planifiées sous la forme de séquence des tâches. Deux ou plusieurs tâches peuvent également être programmées pour s'exécuter parallèlement les unes aux autres. Il s'agit d'un système évolutif, fiable et extensible (Islam *et al.*, 2012).

2.3.8 Mahout

Mahout est une extension de bibliothèque performant pour Machine Learning dans les systèmes distribués utilise les langages Scala, R et Python. La version actuelle Samsara, fournit un environnement mathématique, tel que l'algèbre linaire, statistique et analyse en composants principale. Mahout utilisé pour les problèmes de classification et les systèmes de recommandation dans les systèmes distribués. Mahout fournit des services complets s'adapte aux MapReduce et Apache Spark (Owen, 2012).

2.3.9 Apache Tez

Apache Tez est un moteur de traitement Hadoop YARN, qui permet des requêtes interactives. Apache Tez simplifie le codage des flux de travail centrés sur les données et améliore les performances du traitement MapReduce de Big Data pour les applications basées sur Apache Hive, Apache Pig ou l'apprentissage machine (Requeno *et al.*, 2018).

2.3.10 Flink

Un api pour les applications streaming, il est conçu pour résoudre le problème de mini lot de Spark streaming. Flink supporte les applications en Scala et Java. Tous les Flink application traitée comme des stream application, le traitement de tâche est effectué par le graphe acyclique dirigé (Akil *et al.*, 2017).

2.3.11 Storm

Storm est une plate forme à source libre évolutive et tolérante aux pannes traitées de manière fiable des flux de données illimités en streaming et en temps réel. Storm fournit des applications distribuées en temps réel. Les composantes de base Storm sont les Spout et les Blots. Les Spots représentent la source des données dans Storm. Le développeur écrire des Spouts pour lire les données à partir de sources de données telles que des bases de données, des systèmes de fichiers distribués, messagerie, etc. Les Spout peuvent être classés en plusieurs catégories. Bolt c'est l'unité logique de traitement dans Storm. On peut utiliser les Bolts pour effectuer tout type de traitement, comme le filtrage, l'agrégation, l'assemblage,

l'interaction avec les données, la communication avec des systèmes externes, etc (v. d. Veen *et al.*, 2015)

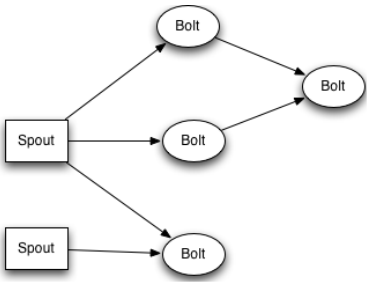

FIGURE 2.8 – Architecture du Strom (Jain, 2017)

2.3.12 Casandra

Casandra est une base de données avec architecture peer-to-peer distribuée NoSql fonctionne dans des clusters avec des nœuds homogènes. Les avantages de Casandra son évolutif élastique, haute disponibilité et tolérance aux pannes haute performance (Hewitt, 2010).

2.3.13 Kafka

Kafka est un framework distribué et open source écrit en Scala et Java. Kafka est développé par LinKedIn en 2010. Le but est de gérer le service de messagerie de la manière la plus souple et efficace. Kafka est couramment utilisé pour la collection des fichiers logs (Garg, 2013).

2.3.14 Zookeeper

Zookeeper est un logiciel de haut niveau, développé par Apache, qui agit comme un service centralisé. Il est utilisé pour maintenir les données de nommage et de configuration et pour fournir une synchronisation flexible et robuste au sein des systèmes distribués. Zookeeper garde une trace de l'état des nœuds du cluster Kafka et garde également une trace des sujets Kafka, des partitions, etc. Zookeeper permet à plusieurs clients d'effectuer des lectures et des écritures simultanées et agit comme un service de configuration partagé au sein du système. Le protocole de diffusion atomique Zookeeper est le cerveau de l'ensemble du système, ce qui permet à Zookeeper d'agir comme un système de diffusion atomique et d'effectuer des mises à jour régulières (Hunt *et al.*, 2010).

2.4 Architecture du Big Data

De nombreux chercheurs sont intéressés par Big Data, dont le but est la proposition des architectures et éclairer l'ensemble des composants et les concepts pour faciliter son implémentions. Les questions fondamentales que nous posons sont :

— Pourquoi l'architecture du Big Data ?

— Quels sont les composants de ces architectures ?

— Quelles sont les différentes classifications de ces architectures ?

2.4.1 Spécification des besoins pour l'architecture du Big Data

La conception d'un Framework conceptuel pour l'architecture Big Data est différente de celles des architectures des données classiques. Krishnan (2013), cite l'ensemble des besoins pour la conception de l'architecture du Big Data par :

1. Volume :
   - La taille des données à traiter est large ; elle doit être divisée en blocs gérables.
   - Les données doivent être traitées en parallèle sur plusieurs systèmes.
   - Les données doivent être traitées simultanément sur plusieurs modules de programme.
   - En raison des volumes, les données doivent être traitées une seule fois et être traitées jusqu'au bout.
   - Les données doivent être traitées à partir de n'importe quel point de défaillance, car il est extrêmement coûteux de redémarrer dès débuts du processus.
2. Vélocité :
   - Les données doivent être traitées à des vitesses en streaming pendant la collecte des données.
   - Les données doivent être traitées pour plusieurs points d'acquisition.
3. Variété :
   - Les données de différents formats doivent être traitées.
   - Les données de différents types doivent être traitées.
   - Les données des différentes structures doivent être traitées.
   - Les données des différentes régions doivent être traitées.
4. Ambiguïté :
   - Les Big Data sont par nature ambiguës en raison du manque de métadonnées et de contexte pertinents dans de nombreux cas.
5. Complexité :
   - La complexité du Big Data nécessite l'utilisation de nombreux algorithmes pour traiter les données rapidement et efficacement.
   - Plusieurs types de données nécessitent un traitement multiple.

2.5 Classification d'architecture Big Data

Avant de définir notre classification, nous devons tout d'abord établir la compréhension de l'architecture. La littérature scientifique offre plusieurs définitions pour décrire ce dernier terme. Certaines définitions les plus répandues sont adoptées aux définitions suivantes : Garlan & Perry (1995) défini comme structure des composantes d'un programme et système, leurs interrelations, et les principes lignes et directrices régissant leur conception et leur évolution dans le temps. Jen & Lee (2000) ont vu l'architecture comme une organisation fondamentale d'un système constitué de ses composantes, de leurs relations entre elles avec l'environnement et des principes qui guident sa conception et son évolution. Ces définitions décrivent l'architecture comme structure. Cette structure est formée des composants ou d'éléments et des relations ou connecteurs entre eux. Dans la littérature scientifique de nombreux travaux de recherche sont proposés. Nous faisons une classification des architectures : architecture générale, architecture référentielle, architecture orientée application et architecture basée sur le Cloud Computing.

— Architecture générale : les architectures générales sont des architectures dont leur but est la proposition d'une architecture sans poser la question des vocabulaires et concept commun pour tous les problèmes qu'on a besoin des architectures du Big Data.

— Architecture référentielle : l'architecture référentielle peut faciliter la création d'architectures concrètes et améliorer la compréhension de la vision globale en incluant des fonctionnalités et des flux de données typiques dans les systèmes Big Data. Une architecture de référence est également utile pour analyser les systèmes Big Data existants, en fournissant la base de classification des processus d'analyse des données et technologies. La catégorisation des processus, des technologies et des services en groupes facilite encore la prise de décision concernant la réalisation des processus et des fonctionnalités du système. Une architecture de référence assure l'interopérabilité des systèmes par la normalisation. Elle facilite ainsi l'instanciation de nouvelles architectures concrètes (Angelov *et al.*, 2012).

— Architecture orientée application : l'objective de ce type d'architecture est la proposition d'une architecture pour une application, par exemple Big Data pour les maisons intelligentes, Big Data pour le transport, Big Data pour la santé, etc. De plus les concepteurs d'architecture Big Data peut résoudre d'autres problèmes comme la sémantique de Big Data ou les Big Data dans les réseaux sociaux.

— Big data pour le Cloud computing : les architectures intègrent le Cloud computing. Il offre les servies de Cloud, softwre as service, plateforme as service et infrastructure as service.

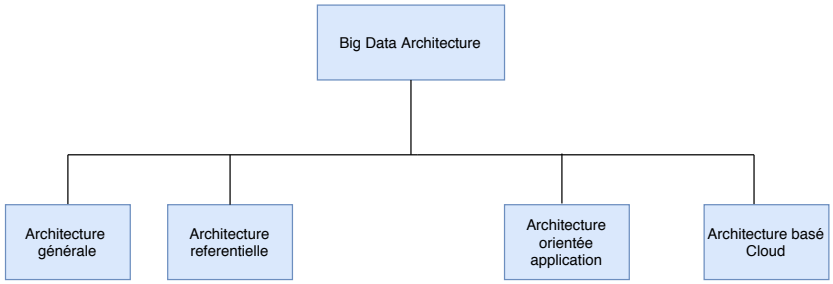

FIGURE 2.9 – Classification des architectures Big Data

Il existe des relations entre ces classes de telle sorte que le concepteur de Big Data suivait un processus séquentiel pour construire une architecture Big Data . Nous proposons le processus décrit dans la figure ci-dessous 2.10 pour la conception des architectures Big Data .

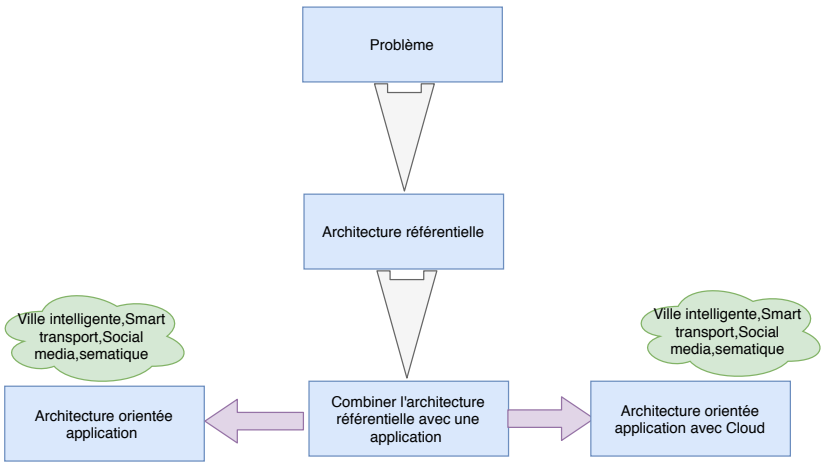

FIGURE 2.10 – Processus méthodologique pour la construction de l'architecture Big Data

2.5.1 Architecture Générale

2.5.1.1 Anthony & Arjuna (2011)

LexisNexis a proposé une architecture traite l'agrégation de données et des services d'information. Cette architecture développé et mis en œuvre de manière indépendante une solution de calcul intensif de données appelée HPCC (High-Performance Computing Cluster), qui était à l'origine appelée DAS (Data Analytics Supercomputer). La vision de LexisNexis pour ce calculateur est basé sur les composants suivantes : contenu structuré et non structuré, super calculateur pour analyser les données, application de l'analyse. L'architecture met principalement en œuvre deux environnements de traitement en cluster distincts : l'un pour l'extraction des données, appelé Thor, l'autre pour la livraison des données, appelé Roxie.

— Thor : le cluster du système Thor est mis en œuvre selon une approche maître/esclave avec un seul processus maître et plusieurs esclaves qui fournissent un environnement d'exécution parallèle pour les programmes codés .Un cluster Thor est similaire dans sa fonction, son environnement d'exécution, son système de fichiers et ses capacités à ceux de Google et Hadoop MapReduce, mais offre des performances nettement supérieures dans les configurations matérielles.

— Roxie : est un acronyme pour Rapid Online XML Inquiry Engine. Roxie utilise un système de fichiers indexés et distribués pour permettre le traitement parallèle des requêtes. Un cluster Roxie est similaire dans sa fonction au Hadoop avec l'ajout des capacités HBase et Hive. Il offre un débit nettement plus élevé puisqu'il utilise un environnement d'exécution, et un système de fichiers plus optimisé pour un traitement en ligne à haute performance en exécutant plusieurs requêtes en parallèle. Les deux clusters Thor et Roxie utilisent le même langage de programmation déclaratif de haut niveau et implicitement parallèle centré sur les données pour le traitement de données parallèle, appelé ECL (Enterprise Control Language).

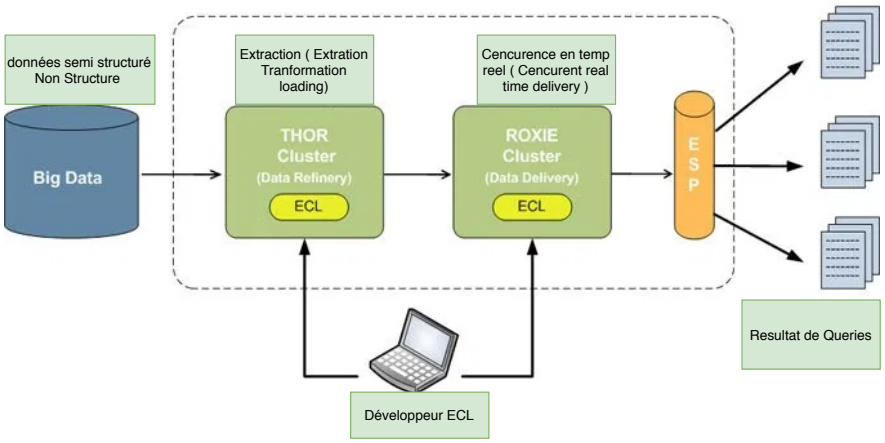

FIGURE 2.11 – Cluster de calcul haute performance (Anthony & Arjuna, 2011)

2.5.2 Architecture référentielle

2.5.2.1 Architecture de Demchenko

Demchenko *et al.* (2014) Définit des composants d'architecture Big Data écosystème. L'auteur a traité le changement de paradigme, qui consiste à passer d'une architecture traditionnelle basée sur les hôtes vers les services à une architecture et des modèles opérationnels

centrés sur les Big Data. Demchenko *et al.* (2014) cité tous les aspects du Big Data écosystème. Les éléments de Big Data comprennent : structures et modèles de données, gestion de cycle de vie Big Data, outils d'analyse de Big data, sécurité de Big Data, infrastructure Big Data.

— Modèles et structures de données : les différentes étapes de la transformation des Big Data nécessiteront des structures, des modèles et des formats de données différents, y compris la possibilité de traiter à la fois des données structurées et non structurées.

— Gestion de cycle de vie Big data : se reflète dans le modèle de gestion du cycle de vie de Big Data : la collection de données, filtrage de données, analyse de données, visualisation de données.

— Outils d'analyse des Big Data : les outils d'analyse pour le Big Data par exemple, Map reduce MLlib.

— Sécurité des données : cadre de sécurité des données avec un accent particulier sur les données centrées qui devraient permettre un stockage, un transfert et un traitement sécurisés des données dans une infrastructure de stockage et de traitement des données distribuées.

— Infrastructure de Big Data : comprend l'infrastructure générale de gestion des données, généralement basée sur le cloud, et la partie analyse de Big Data qui nécessitera des clusters de calcul haute performance, qui à leur tour nécessiteront un réseau haut.

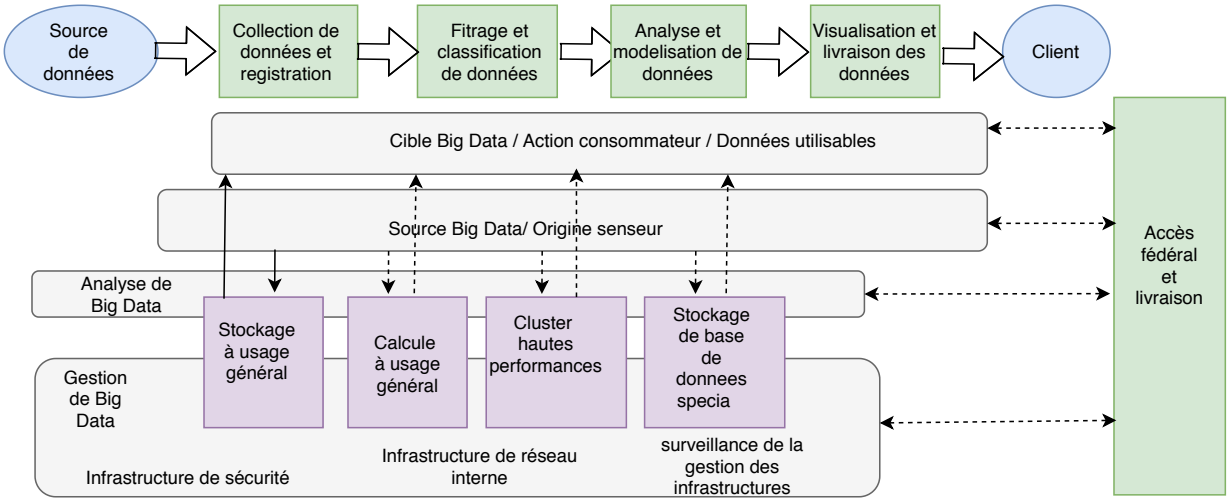

FIGURE 2.12 – Composantes fonctionnelles générales de Big Data (Demchenko *et al.*, 2014)

2.5.2.2 Architecture de Levin

Les composants principaux de Big Data sont : la source de données, transformation des données, infrastructure de données et l'utilisation des données (Levin, 2013).

— Source de données : la source de collection de données.

— Transformation des données : au fur et à mesure que les données se propagent dans l'écosystème, elles sont traitées et transformées de différentes manières afin d'extraire la valeur de l'information.

— Infrastructure de Big Data : est un ensemble de logiciels de stockage ou de base de données, de serveurs, de stockage et de mise en réseau utilisé pour soutenir les fonctions de transformation des données et pour le stockage des données selon les besoins.

— Utilisation de données : fournies sous différents formats, avec une granularité différente et sous différentes considérations de sécurité.

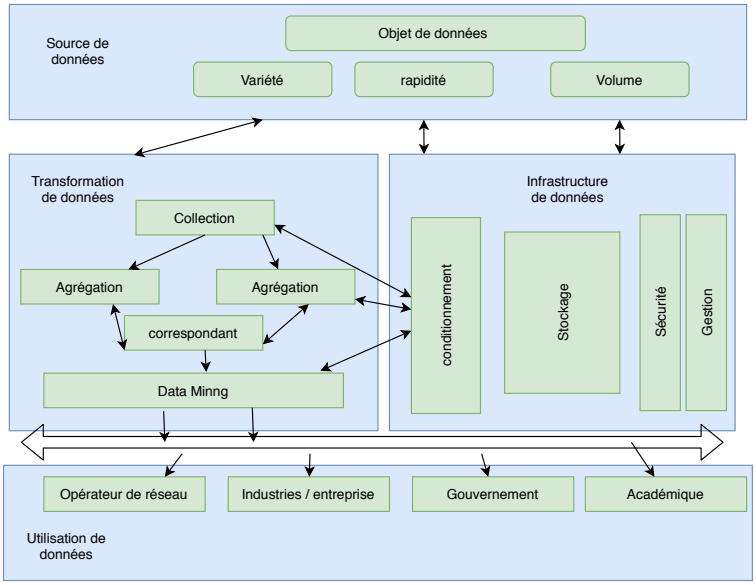

FIGURE 2.13 – Architecture référentielle de Levin (Levin, 2013)

2.5.2.3 Architecture de Supriya

Supriya et ses collaborateurs (Supriya *et al.*, 2016), ont résumé les architectures proposées par les organismes de normalisation disponible dans la littérature (Chang *et al.*, 2015; Alliance, 2012). Les auteurs proposées l'architecture dont les composantes sont les suivants :

— Source des données : les données qui proviennent d'une source hétérogène comme, système de gestion des données, dispositifs intelligents, etc. Les données de types semi-structurée, non structurées. Ces données peuvent varier en termes de format et d'origine. Ces données sont rassemblées dans un système de fichiers appelé Landing zone. En outre, ces fichiers sont séparés dans des sous-répertoires en fonction du type de données. Toutes les modifications apportées aux fichiers, comme le nom et l'extension, peuvent être effectuées dans cette couche.

— Couche de messagerie et stockage de données : dans cette couche, les données collectées sont séparées et chargées sur la base des métadonnées et préparées pour la transformation qui se fait avec différents composants tels que :

  — L'acquisition des données : acquièrent des données provenant de diverses sources.

  — Messagerie de données et stockage : cette composante doit pouvoir déterminer si les données doivent envoyer par message avant de pouvoir être stockées ou les données peuvent être directement envoyées à la couche d'analyse.

  — Digest de données : ce composant est chargé d'envoyer les données dans le format nécessaire pour atteindre l'objectif de l'analyse. Les données chargées sont transformées en plus petits morceaux de fichiers. Un catalogue des dossiers est préparé et les métadonnées sont traitées. Cette étape est basée sur les besoins de l'utilisateur et du traitement. Les données peuvent être partitionnées horizontalement ou verticalement.

  — Stockage de données distribuées : ce composant est responsable de stockage des données à partir de sources de données. Souvent, plusieurs options de stockage

de données sont disponibles dans cette couche telle que le stockage de fichiers distribués (DFS), le cloud, les sources de données structurées, et NoSQL.

— Couche d'analyse : Les données sont transformées sur la base des règles commerciales requises. Cette couche a de multiples composantes à exécuter qui sont :
  — Identification de l'entité : cette composante est chargée d'identifier et de remplir les entités contextuelles. Le composant recueil de données doit compléter ce composant d'identification des entités en envoyant les données dans le format requis.
  — Moteur d'analyse : ce composant peut avoir divers flux de travail, algorithmes et outils qui prennent en charge le traitement parallèle.
  — Gestion des modèles : ce composant est le responsable de la maintenance, de la vérification et la validation des différents modèles statistiques en les formant pour qu'ils soient plus précis.
— Couche de consommation : Dans cette couche, l'ensemble de données qui en résulte peut-être utilisé pour des traitements ultérieurs tels que l'analyse, la production de rapports, l'entreposage, l'intégration et la visualisation. Toutes ces couches peuvent être représentées à partir de l'architecture générale outre ces principales couches de traitement, les données volumineuses peuvent également avoir certains des protocoles définissant des fonctions telles que la sécurité, la confidentialité, la gouvernance des données, etc.

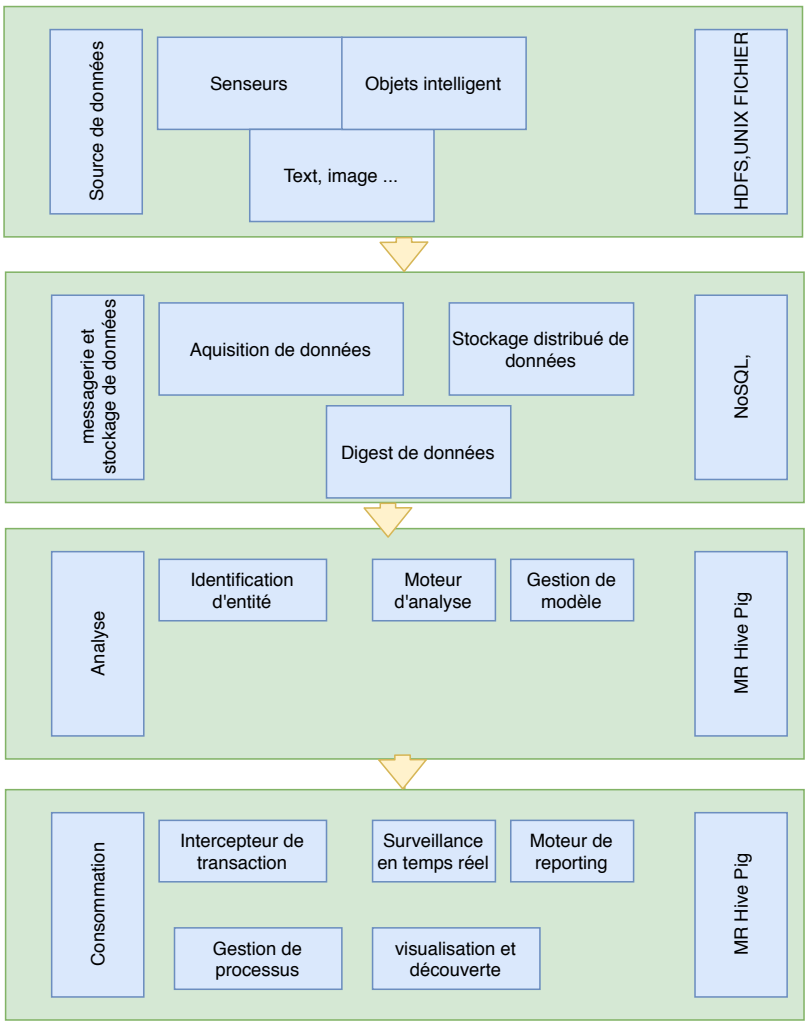

FIGURE 2.14 – Architecture référentielle de Supriya (Supriya *et al.*, 2016)

2.5.2.4 Architecture de Sang

Sang et ses collaborateurs (Sang *et al.*, 2016) ont proposés une architecture référentielle pour Big Data système. Cette architecture couvre tous les aspects et les processus de Big data dont l'objectif est de combler les insuffisantes des architectures Big data et enrichir le niveau d'abstraction de la conception. Sang *et al.* (2016) devisés les architectures Big data en cinq éléments fondamentaux, la source de données, collection de données, analyse et agrégation de données, interface et visualisation, spécification de modèle et Job avec :

— La source de données : se réfère à la source originale des données collectées.

— Collection de données : la collecte de données à partir de multiples sources.

— L'analyse et agrégation des données : se réfèrent aux tâches et aux processus d'analyse de données, tandis que l'agrégation se réfère au stockage de données, y compris les données multidimensionnelles qui stockent les résultats de l'analyse.

— Interface et visualisation : ils représentent les utilisateurs finaux ainsi que des applications telles que les tableaux de bord.

— Spécification du modèle : cela couvre les modèles d'entraînement, les spécifications et la programmation des Jobs avec leurs procédures de stockages.

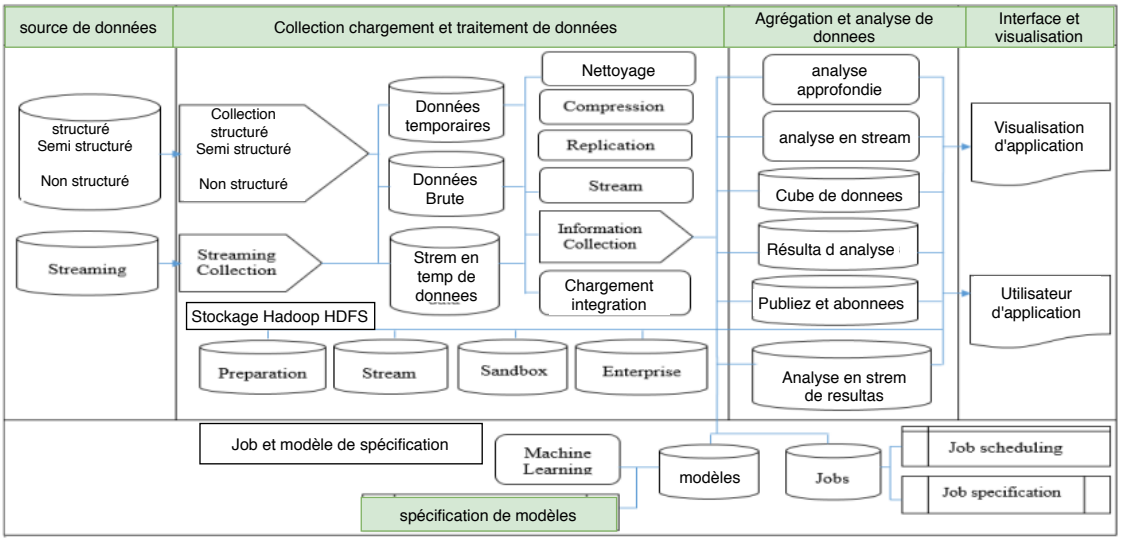

FIGURE 2.15 – Architecture référentielle de Sang (Sang *et al.*, 2016)

2.5.2.5 Architecture de Pääkkönen et Pakkala

Pääkkönen & Pakkala (2015) proposés une architecture référentielle pour le Big Data. Les composons de cette architecture sont, source de données, extraction de données, prétraitement du chargement des données, traitement de données, analyse de données, chargement et transformation de données et interfaçage et visualisation.

— Source de données : ce composant est le responsable des sources des données qui peut être structuré, semi-structuré ou non structuré. Les données en relation avec la vitesse de génération sont en temps réel, lot, flux ou instantané.

— Extraction de données : l'extraction de données est le mécanisme de l'extraction de données depuis une source de données vers le système Big Data par exemple, dans le cas de temps réel d'un senseur internet d'objet le Big Data doit être capable de

l'extraction de données en temps réel. Le système Big Data fournit un mécanisme qui permet de sauvegarder les données temporairement ou de le chargé et sauvegardé dans un trame de données.

— Prétraitement du chargement des données : les données transformées sous la forme interprétable par la machine comme fichier CSV, base de données.

— Traitement de données : les données peuvent combiner, filtrer ou mettre une réplication.

— Analyse de données : ces l'ensemble de composants responsables au processus d'analyse et extraction de connaissance de Big Data.

— Chargement et transformation de données : un ensemble de fonctions et de services fournis par le système par exemple le service OLAP (On line Analytic Application) serve un moyen d'analyse pour les utilisateurs de système.

— Interface et visualisation : c'est le composant visible par les utilisateurs de système, par exemple le tableau de bord, et les applications des utilisateurs.

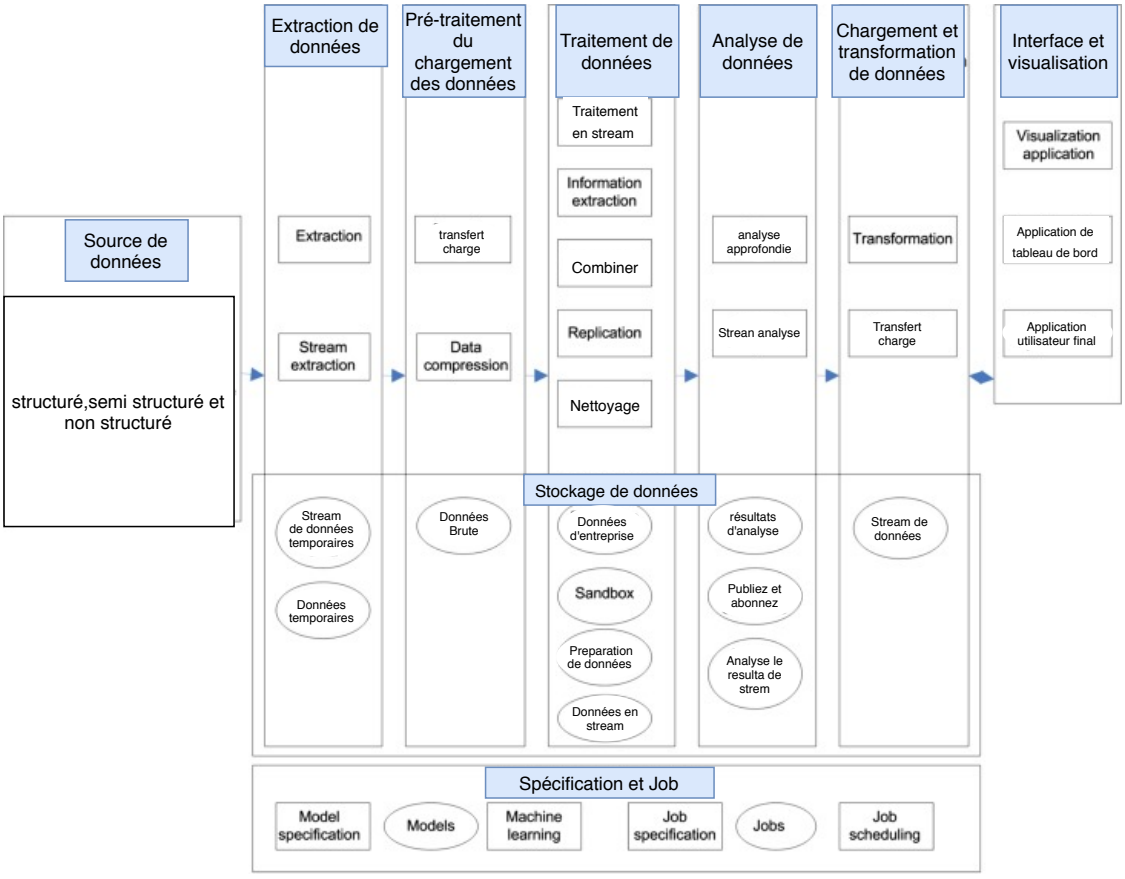

FIGURE 2.16 – Architecture référentielle de Pääkkönen et Pakkala (Pääkkönen & Pakkala, 2015)

2.5.3 Architecture orientée application

2.5.3.1 Architecture SCDAP d'une ville intelligente

Osman (2019) a proposé une architecture Big Data pour les villes intelligentes appelées SCDAP (Smart City Data Analytics Panel). SCDAP est une architecture à 3 couches comprenant : la couche plate-forme, la couche sécurité et la couche de traitement de données.

— Couche plate-forme : est une plate-forme horizontalement extensible comprenant des clusters de matériel, des systèmes d'exploitation et des protocoles de communication.

Dans les plates-formes horizontalement évolutives, des nœuds de calcul supplémentaires peuvent être ajoutés selon les besoins.

— Couche sécurité : les mesures de sécurité suivantes doivent être respectées lors de la conception physique, en particulier pour les analyses critiques :

  — Un accès restreint au cadre devrait être accordé pour les données critiques et sensibles.

  — Authentification de l'utilisateur à plusieurs niveaux.

  — Un journal d'audit complet doit être tenu pour les opérations importantes.

— Couche de traitement de données : il s'agit du mécanisme central de traitement de données qui fournit toutes les fonctionnalités de traitement de données à partir de l'acquisition des données à l'extraction des connaissances. Cette couche prend en charge à la fois le traitement de données en temps réel et des données historiques respectivement. En outre, cette couche fournit deux fonctionnalités importantes qui distinguent le SCDAP, à savoir gestionnaire de modèle et agrégation de modèles lorsqu'ils sont extraits les modèles de données peuvent être gérés et agrégés, respectivement. De manière générale, les composantes de cette couche sont résumées comme suit :

  — Acquisition de données : la principale composante de la saisie de données provenant du monde extérieur. Elle se caractérise par l'évolutivité, l'interopérabilité. L'extensibilité et la capacité à s'adapter dynamiquement au nombre croissant de données, l'interopérabilité est la capacité d'interagir avec des types hétérogènes.

  — Prétraitement des données : fournit des fonctionnalités de nettoyage, de transformation et d'intégration des données d'entrée. Cette composante est responsable de la transformation des données d'entrée en format prêt pour l'analyse.

  — Analyse en ligne : capacité à effectuer le traitement des données en streaming pour des applications.

  — Analyse en temps réel : capacité à traiter des données en temps réel. Les applications en temps réel sont des applications qui fonctionnent dans des délais que l'utilisateur perçoit comme immédiats ou actuels.

  — Dépôt de données par lots : système de gestion du stockage de données, comme Hadoop HDFS, NoSQL database.

  — Analyse de données par lots : analyse de données par lots pour les applications.

  — Référentiel de modèles : le système de gestion de modèles de données extraites, où les modèles d'analyse de données résultantes peuvent être persistants, récupérés ou supprimés avec les métadonnées pertinentes pour des enquêtes futures ou la réutilisation.

  — Agrégation de modèles : fonctionnalité d'ensemble des modèles de données extraites pour des analyses et des interrogations de plus haut niveau et plus complexes.

  — Application intelligente : applications intelligentes construites à partir de modèles d'analyse de données résultants.

  — Interface utilisateur : interface utilisateur pour fournir des outils flexibles permettant l'accès, le reporting et les enquêtes ad hoc pour les modèles persistants et/ou agrégés.

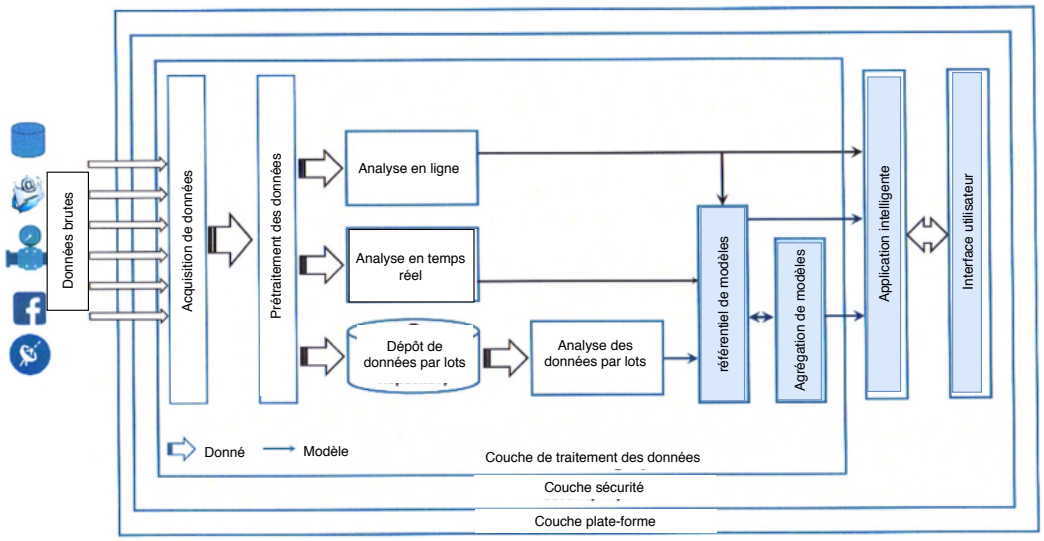

FIGURE 2.17 – Architecture SCDAP d'une ville intelligente (Osman, 2019)

2.5.3.2 Architecture Big Data pour la construction de l'analyse des déchets

L'objectif est de proposer une architecture Big Data pour l'analyse des déchets de construction. À cette fin, au cœur de cette architecture, des composants basés sur des graphes sont utilisés : en particulier, une base de données de graphes (Neo4J) est adoptée pour stocker des ensembles de données très volumineux et diversifiés. En complément, Spark, un système de traitement des graphes très résistant est utilisé. Des extensions par le biais de la modélisation des informations sur les constructions sont également envisagées pour une synergie et une plus grande adoption. Cette intégration symbiotique des technologies permet de créer un environnement dynamique pour l'exploration de la conception et l'optimisation de la gestion des déchets de construction. Les différents éléments constituent l'architecture sont classés en trois couches qui comprennent la couche application, la couche analyse et la couche stockage.

— La couche de stockage : est responsable de la conception de l'outil de simulation des déchets. Il existe deux types de données. Les données historiques qui contiennent les documents de conception, les informations sur les projets et les dossiers d'élimination des déchets. Ces données sont téléchargées une fois et utilisées principalement pour développer le système robuste de gestion des déchets, les modèles d'estimation et les données contenues.

— Couche d'analyse : l'outil de simulation de la valeur réelle des déchets réside dans la capacité à analyser les données. La prévision et la conception des déchets sont des données à forte intensité de calcul. Spark est utilisé pour ces analyses de grande classe, car il offre MapReduce application avec ses capacités inhérentes de stockage en mémoire et de calcul . Les filières d'analyse pour l'estimation et la minimisation des déchets sont principalement mises en œuvre à l'aide de SparkR, MLLib et GraphX. À chaque itération du pipeline analytique, la conception est explorée dans une multitude de dimensions est optimisée en vue de l'efficacité des déchets. L'efficacité de la conception est calculée instantanément et diffusée aux concepteurs, qui peuvent réagir de manière proactive pour atténuer l'impact de la conception et le changement vers l'efficacité des déchets. Ces prévisions intermédiaires ainsi que les prescriptions sont capturées sous forme de graphiques annotés RDF dans le Triple.

— Couche application : l'API Autodesk Revit est utilisée pour le développement du plugin. Revit est largement adopté dans le monde entier et dispose d'un support solide pour l'analyse visuelle prévue. La couche d'application de l'outil de simulation des déchets est construite par l'exploitation des programmes et d'API. Les changements de conception sont les suivants capturés par le biais des fichiers , qui sont chargés dans le HDFS par FLum Apache et finalement chargés au Store Triple. Le Spark Streaming déclenche le pipeline d'analyse pour estimer les déchets et suggérer des idées d'action pour optimiser la conception. Ces idées sont présentées sous la forme des prescriptions, qui sont mises en correspondance avec les attributs visuels du Revit en exploitant la vue de modèle de conception du contrôleur. Les prévisions et les prescriptions sont communiquées au fur et à mesure par le modèle prédicatif PMML (Predictive Model Markup Language). Les concepteurs reçoivent un retour d'information instantané pour optimiser la conception de la construction. Il s'agit d'un processus itératif qui commence à chaque conception et dure après quelques itérations en produisant le dessin. Il est optimal en termes de matériaux, le coût et l'efficacité des déchets.

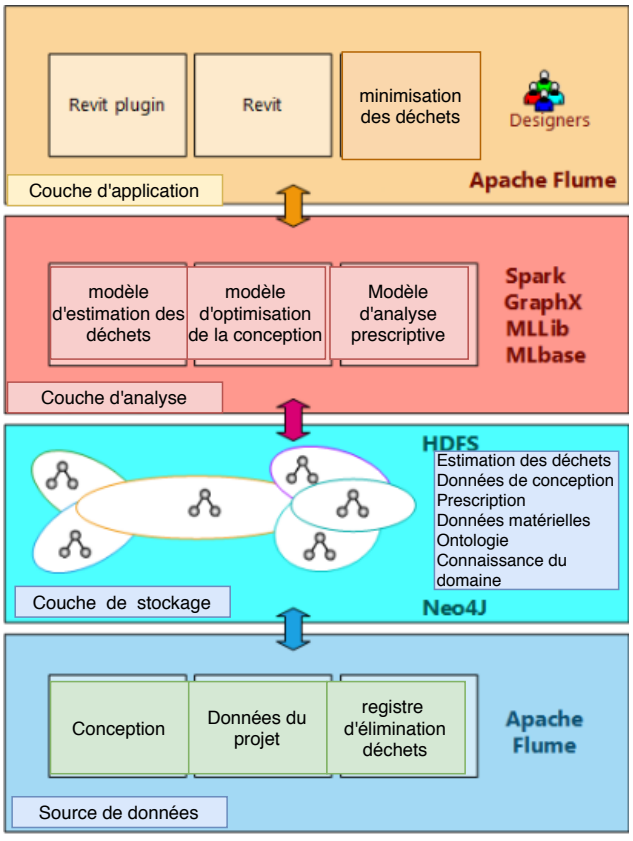

FIGURE 2.18 – Architecture Big Data pour la construction et l'analyse des déchets (Bilal *et al.*, 2016)

2.5.3.3 Transport intelligent

Transport intelligent propose un mécanisme axé sur les graphes pour réaliser le système de transport dans la ville. Le réseau de capteurs déployé pour obtenir les informations générales sur le trafic ainsi qu'un réseau de véhicules pour obtenir la localisation et la vitesse des informations sur le véhicule individuel. C'est un système d'internet des objets, génèrent un volume énorme de données. décrivant les informations sur le trafic de la ville. Rathore *et al.* (2015) ont proposés une architecture efficace qui utilise les outils graphiques avec serveurs de traitement parallèle pour réaliser l'efficacité du temps. Divers algorithmes de graphes

sont utilisés pour réaliser la décision en temps réel de transports intelligents dont le but de faciliter la vie des citoyens ainsi que les autorités. Les données des véhicules et les ressources représentant le trafic réel de la ville sont utilisées pour l'objectif d'analyse et l'évaluation. Le système est mis en œuvre par l'utilisation de l'outil Spark et Hadoop pour générer et traiter des graphes en temps réel . En outre, le système est évalué en termes d'efficacité en considérant le débit du système et le temps de traitement. Le système est divisé en sept couches, la couche des sources de données, la couche de communication, la couche de construction de graphe, la couche de traitement de graphe, la couche de résultats, la couche d'interprétation et la couche application. Chaque couche a sa propre fonctionnalité :

— Couche source de données : est la responsable de la génération des données. Elle comprend tous les aspects de la réalisation d'un réseau de véhicules et du déploiement de capteurs routiers pour générer des informations sur le trafic en temps réel. Les données générées peuvent ainsi être transmises sur l'internet en utilisant des nœuds de relais, des coordinateurs et des passerelles.

— Couche de communication : c'est la responsable de la transmission des données entre les véhicules et le système d'analyse principal, entre les capteurs et le système d'analyse, et entre les différentes unités du système d'analyse. Elle utilise la technologie cellulaire, telle que GPRS, 3G, 4G/LTE, WIMAX pour transmettre des données des véhicules à l'internet. En outre, il utilise la communication Wi-Fi ou Bluetooth pour transférer les données du capteur routier vers l'internet. Toute la communication avec les différentes unités du système d'analyse se fait par Ethernet.

— La couche création de graphes : l'une des principales couches du système, qui génère et met à jour les graphes par l'utilisation des données des véhicules . Elle crée une nouvelle et des mises à jour de graphe. Elle utilise un mécanisme de recherche efficace, qui utilise l'indexation pour rechercher un bord (edge) particulier et mettre à jour si nécessaire. Cette couche augmente également l'efficacité du système en rendant le graphique à traiter sur plusieurs parallèles simultanément tout en divisant le graphique en diverses parties (sous-graphiques) indépendants et mutuellement exclusifs. Ensuite, elle envoie tous les sous-graphes indépendants au serveur de traitement, lorsqu'un traitement est nécessaire. Par conséquent, elle assure également la fonction d'équilibrage de la charge.

— Couche de traitement du graphe : dispose de plusieurs serveurs parallèles pour traiter chaque sous-graphe individuel. Chaque serveur de traitement est équipé de divers algorithmes de graphes, qui fonctionnent en fonction de la demande de l'utilisateur ou des exigences des autorités. À cette couche, chaque serveur a une sortie correspondant à chaque algorithme de graphe pour chaque sous-graphe du graphe principal. Le résultat de chaque serveur est agrégé à la couche suivante.

— La couche des résultats : après l'agrégation, l'analyse est effectuée. La couche de traitement produit des sous résultats et chaque sous résultats correspond à un sous-graphe. Par conséquent, ces sous résultats doivent être agrégés pour l'analyse finale.

— Couche interprétation et application : enfin, aux deux derniers niveaux, les décisions sont prises sur la base des résultats de l'analyse et annoncées à l'audience requise. Ces résultats peuvent être utilisés pour l'identification du chemin efficace de la source à la destination en fonction des conditions de circulation actuelles, ou peuvent être toute annonce aux autorités concernant la circulation, comme le blocage de la route, une intensité de trafic plus élevée, des accidents, etc.

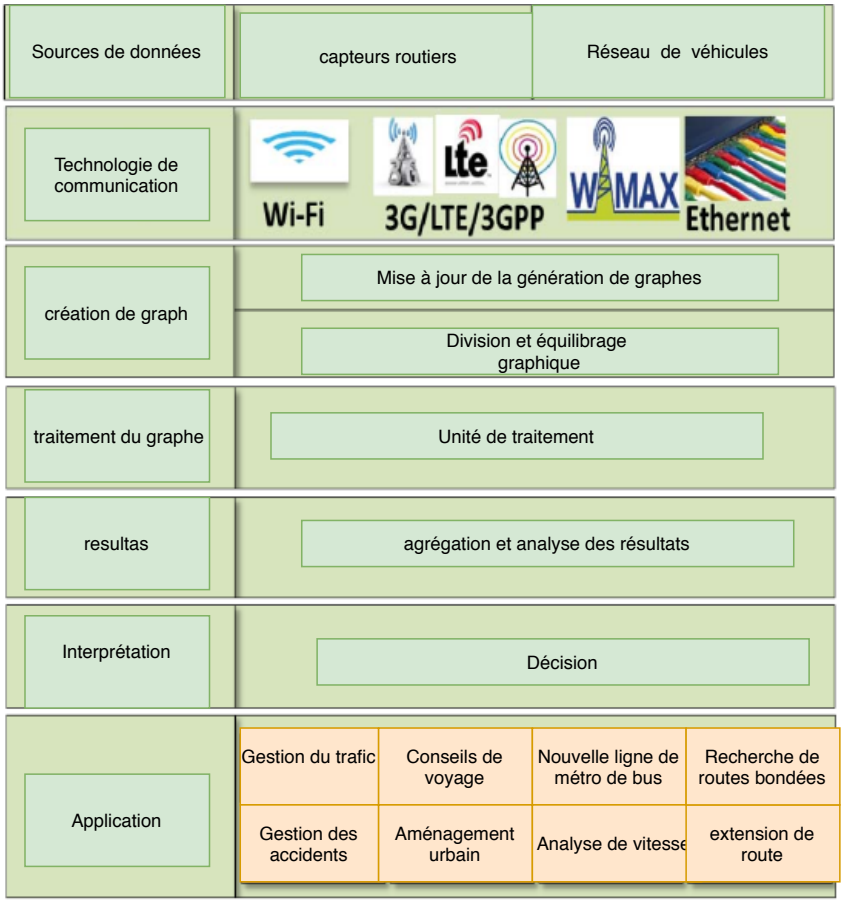

FIGURE 2.19 – Transport intelligent (Rathore *et al.*, 2015)

2.5.3.4 Architecture basée sur la sémantique

Mezghani *et al.* (2015) ont proposé une architecture sémantique générique de Big Data adoptant l'approche connaissance comme service KaaS (Knowledge as a Service) pour le traitement de l'hétérogénéité des données et les besoins des utilisateurs. Le KaaS (Grolinger *et al.*, 2015) a été défini, comme une extension du modèle de référence du NIST (National Institute of Standards and Technology) pour le Cloud Computing (Liu *et al.*, 2011). Il a pour objectif de générer de nouvelles connaissances à partir de données hétérogènes stockées dans le Cloud et les rend disponibles sous forme de service. Les sources multiples génèrent des données hétérogènes qui peuvent être stockée dans des bases de données relationnelles, RDF, NoSQL, etc, ou dans les clusters KaaS grâce à l'application de mise en œuvre. La transformation de ces ensembles de données hétérogènes en connaissances compréhensibles et partageables nécessite des services qui collectent, préparent et traitent ces ensembles de données. L'architecture proposée est composée de trois couches :

— La couche stockage des données : où les données sont stockées dans le Cloud, plus précisément dans la couche PaaS (Platform as a Service).

— La couche Big data : comprend les composants Big Data du NIST associés à des technologies de traitement des données. Elle est principalement composée de quatre services, l'enrichissement sémantique qui donne un sens aux données basées sur l'ontologie. La normalisation des données qui unifie les valeurs des données relatives aux mêmes paramètres observés. Le nettoyage des données qui joue un rôle important dans l'amélioration de la qualité des données en supprimant les enregistrements incorrects ou inexacts telles que celles résultant du mauvais comportement des capteurs. Enfin,

le stockage des données qui permet de stocker les données préparées dans de grands groupes de données pour être exploitées par les services d'analyse et de visualisation.

— La couche de sémantiques : qui offre une compréhension claire aux données. Chaque fournisseur de données doit informer explicitement le KaaS sur ses caractéristiques (par exemple, le type de données qui sont générées sa signification, son lieu de stockage, etc.). Ceci est fait par le biais du service d'annotation sémantique des sources de données qui permet aux fournisseurs utilisant une représentation commune des connaissances basée sur l'ontologie, qui formalise un contexte spécifique à un domaine.

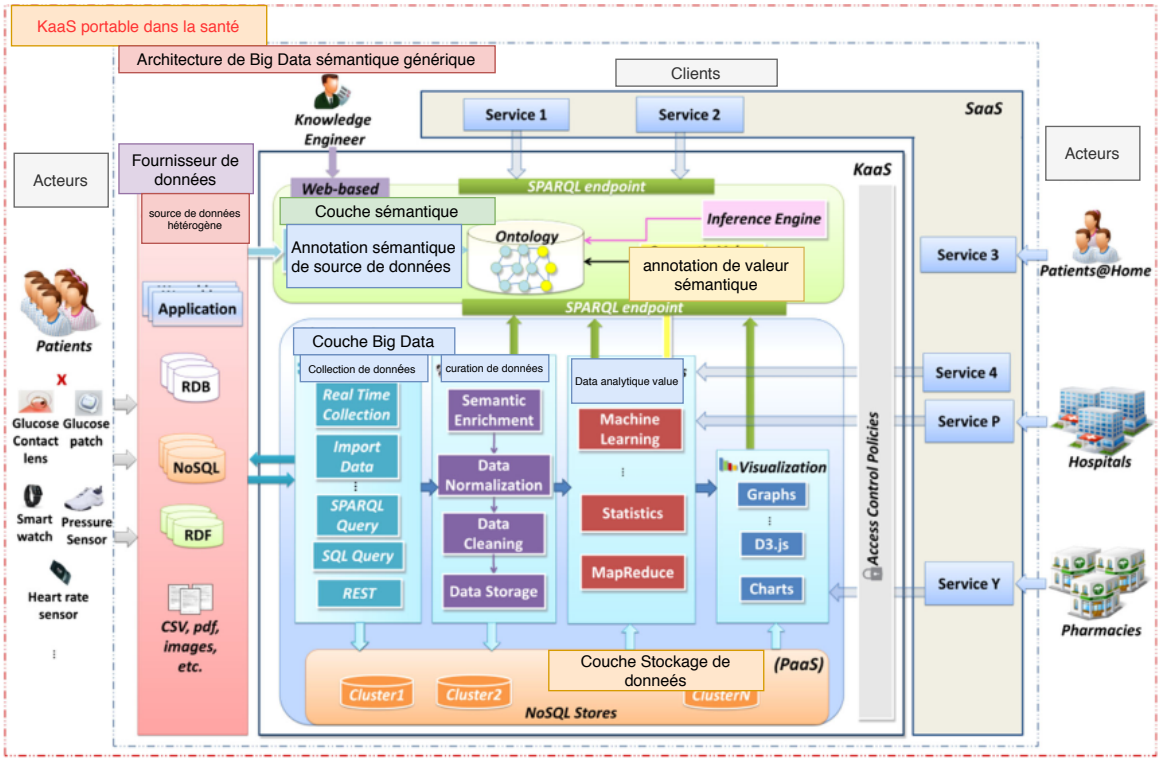

FIGURE 2.20 – Modèle de service pipeline de Big Data (Mezghani *et al.*, 2015)

2.5.3.5 Architecture pour les réseaux sociaux Facebook

Thusoo *et al.* (2010a) Ont présenté une infrastructure d'entreposage et d'analyse des données pour Facebook. Cette architecture réunie et mettre en place un entrepôt de données qui stocke plus de 15 Peta octets de données 2,5 Peta octets après la compression, et charge plus de 60 terra octets de nouvelles données 10 terra octets après compression chaque jour. Le système proposé décrit comment les données circulent dans le système source vers l'entrepôt de données. Le traitement des systèmes de stockage, leurs formats et les optimisations réalisées pour le stockage de Big Data. La source de données : contient my SQL Fédéral et serveur Web, dans mysql Fédéral existe tous les données relatives au Facebook et serveur web qui génère toutes les données du journal. Les données du niveau my SQL Fédéral sont chargées dans les clusters de Hive Hadoop par des processus d'interrogation quotidiens. Ce processus permet d'extraire l'ensembles de données souhaités dans des bases de données my SQL , les données déplacées dans le Cluster Hive Hadoop après la compression des données sources . Il existe deux types de Hive Hadoop où les données deviennent disponibles pour la consommation par les procédures en flux descendant. L'un de ces clusters est Production Hive Hadoop Cluster. Il est utilisé pour exécuter les travaux qui doivent adhérer à des délais de livraison très stricts. Alors que, l'autre cluster ad hoc Hive Hadoop est utilisé pour exécuter des projets

moins prioritaires, les travaux par lots ainsi que toute analyse ad hoc que les utilisateurs souhaitent effectuer sur les ensembles de données historiques. La nature ad hoc des requêtes des utilisateurs faits qu'il dangereux de faire tourner des travaux (Job) de production dans le même cluster. Périodiquement, les données dans les clusters sont compressées par les Jobs de copie et transférées vers le cluster Hive-Hadoop, les copies fonctionnent à des intervalles de 5 à 15 minutes et copier tous les nouveaux fichiers créés dans les clusters. Les données de log sont déplacées vers les Clusters Hive-Hadoop. Les données se présentent principalement sous la forme de fichiers HDFS. Il est publié toutes les heures ou tous les jours sous forme de partitions dans les tables Hive correspondantes via un ensemble de processus de chargement et devient alors disponible à la consommation. Enfin, ces ensembles de données publiés sont transformés par les jobs des utilisateurs ou interrogés par des utilisateurs ad hoc. Les résultats sont soit conservés dans le cluster Hive Hadoop pour une analyse future, soit peuvent même être rechargés au niveau my SQL fédéré afin d'être servi aux utilisateurs de Facebook.

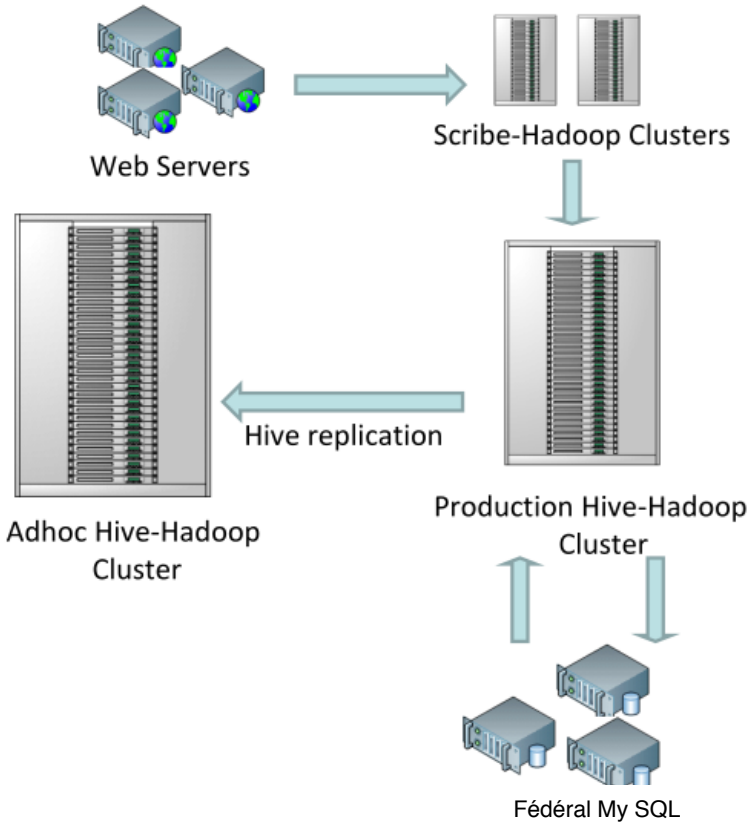

FIGURE 2.21 – Architecture des flux de données de facebook (Thusoo *et al.*, 2010a)

2.5.3.6 Architecture pour les réseaux sociaux LikedIn

Sumbaly *et al.* (2013) ont présenté une architecture d'analyse basée sur Hadoop pour LinkedIn, qui permet aux chercheurs d'extraire les connaissances et de créer des modèles à partir Big Data. Les données entrant dans les fichiers HDFS peuvent être classées en deux catégories : les données d'activité de Kafka (2) et les données de la base de données centrale Oracle (3). Les journaux de changements de la base de données sont périodiquement compactés en séries temporelles pour un accès facile. Les données d'activité consistent en événements de type streaming générés par les services traitant les demandes sur LinkedIn (1). Les événements sont regroupés en thèmes sémantiques et transportés par le système de publication et d'abonnement de LinkedIn, Kafka (1). Ces thèmes sont regroupés dans

une structure de répertoire hiérarchique en fichiers HDFS. Une fois que les données sont disponibles dans Hadoop ETL (Extract, Transform and Load) , il est alors reproduit dans deux cas, l'un pour le développement (5)(Hadoop for developemnt workflows) et l'autre pour la production (5 Hadoop for production workflow). L'architecture utilisée Azkaban, le planificateur de tâches par lots open source de LinkedIn, pour gérer les flux. Azkaban maintient une vision globale des flux des Jobs, ce qui aide la surveillance, le verrouillage des ressources et la collecte des journaux. Chaque Job dans le contexte d'Azkaban est défini dans un fichier de configuration simple de type cle,valeur. Azkaban est un cadre d'exécution général et prend en charge divers types Job tels que MapReduce, Pig, Hive, scripts Shell et autres. Enfin, les résultats de ces flux de Job sont réinjectés dans le système de service en ligne. Les sorties des flux de Kafka (4), des bases de données de type clés valeur par l'utilisation de Voldemort (6) et des cubes OLAP multidimensionnels par Avatara (7) ont une déclaration Pig d'une seule ligne à la fin d'un flux de travail. Avant de redéployer les données vers les systèmes de service en ligne, certains flux de Job peuvent choisir de pousser les données vers des groupes de serveurs Staging Voldemort read-only (8).

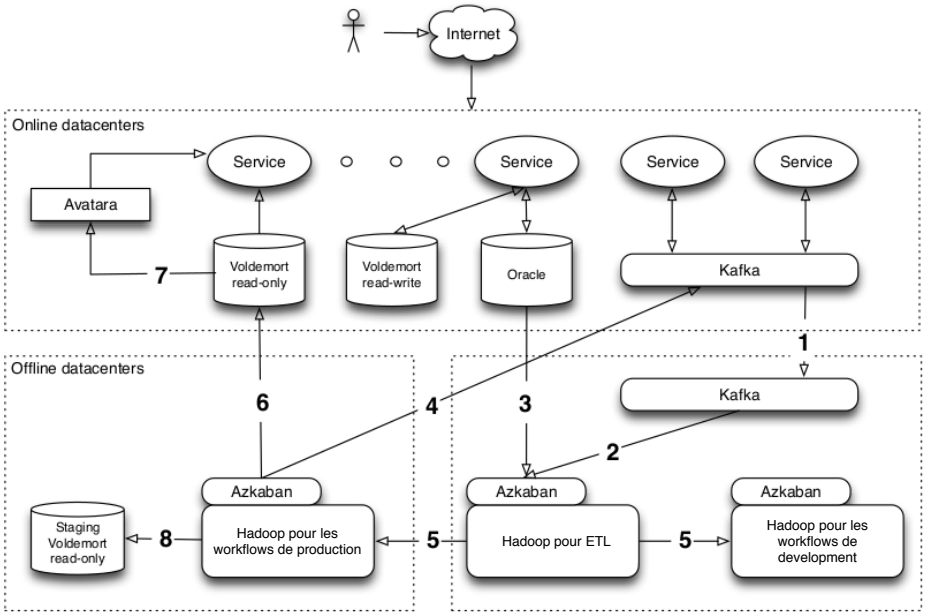

FIGURE 2.22 – Architecture du pipeline de données de LinkedIn (Sumbaly *et al.*, 2013)

2.5.4 Cloud Computing pour l'architecture de Big Data

Développer un Framework qui énumère les alternatives pour la mise en œuvre d'applications Big Data par l'utilisation des services dans le Cloud et identifier les objectifs stratégiques soutenus par ces alternatives. Le Framework pour les Big Data dans le Cloud utilise le modèle de service pour la mise en œuvre du Big Data Pipeline : DaaS (Décision as Service), SaaS (Software as a Service), PaaS (Platform as a Service), IaaS (Infrastructure as a Service), et le modèle de déploiement.

— Le service DaaS : signifie que les processus de décision sont entièrement transférés dans le Cloud. Cela signifie que les services DaaS dans le Cloud peuvent être connectés directement pour créer le Big Data pipeline.

— Le service SaaS : les pipelines Big Data en mode SaaS utilisent des applications virtualisées. Les demandes fournies en mode SaaS contiennent fonctionnalité pour

la mise en œuvre des processus de décision ou mettre en œuvre certaines parties d'un processus de décision. Un pipeline Big Data en mode SaaS, est plus difficile à dimensionner en tant que pipeline DaaS.

— Le service PaaS : le pipeline Big Data activé par le PaaS utilise un logiciel virtuel. Les composants de l'application tels que les bases de données, la file d'attente des messages, etc. La partie virtuelle est plus petite que dans le SaaS.

— Le service IaaS : un pipeline Big Data compatible IaaS utilise des ressources de stockage virtuelle. Par ce moyen, il est possible d'ajouter rapidement des ressources afin de réagir à une demande accrue. Pour permettre une mise à l'échelle, le pipeline monolithique doit mettre en œuvre ses propres mécanismes.

— Modèle de déploiement : la deuxième dimension est le modèle de déploiement pour mettre en œuvre le pipeline Big Data. La définition du Cloud Computing distingue trois modèles de déploiement : privé, public et hybride. Les Cloud privés sont fournis par l'entreprise elle-même. Les services publics dans les Cloud sont fournis par un prestataire de services indépendant. Un Cloud hybride est composé de services fournis à la fois par les services publics et privés.

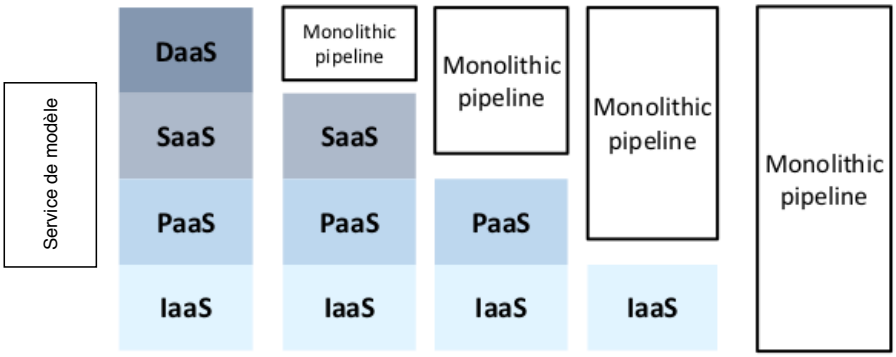

FIGURE 2.23 – Modèle de service Pipeline de Big Data (Schmidt & Möhring, 2013)

2.6 Contribution de Big Data

Dans cette section, nous présentons les contributions récentes pour les Big Data en vues de notre classification. Pour chaque travail, nous illustrons la contribution, la classification de l'architecture utilisée selon notre proposition, les outils et les écosystèmes.

Les Travaux de Big Data			
Référence	Contribution	Classification	Outils
De Maio *et al.* (2017)	Propose la mise en œuvre et le déploiement en ligne de l'analyse de concepts flous temporels sur un système de calcul distribué en temps réel	Application, ville intelligence.	Storm
Huang *et al.* (2017)	Système de découverte d'astéroïdes distribué pour de grandes données astronomiques	Application, astronomie	Spark, Hadoop, Mapreduce

Les Travaux de Big Data			
Référence	Contribution	Classification	Outils
Kranjc *et al.* (2017)	Une plate-forme de calcul distribué pour Machine Learning dans le CLoud	Cloud	Python, Weka, C++, Django
Castiglione *et al.* (2018)	Big Data infrastructure de données utilisée pour interroger, parcourir, analyser et traiter des contenus numériques liés au patrimoine culturel à partir d'un ensemble de données hétérogènes et dépôts distribués.	Cloud,semantique Application	Hadoop Yarn
Maté *et al.* (2016)	Une architecture qui utilise des données énergétiques déjà stockées et des informations externes non structurées pour améliorer l'acquisition de connaissances et permettre aux gestionnaires pour prendre de meilleures décisions.	Application consommation électrique	NoSql, Spark
Bechini *et al.* (2016)	proposer un système de classification basé sur des règles d'association distribuée, conçu selon le modèle MapRedcue	Application	Hadoop Map reduce
Mohapatra *et al.* (2016)	Une architecture Big Data analytique à trois couches est conçue pour analyser les données générées pendant la construction de la barrière et la détection de l'intrusion à l'aide de capteurs de caméra	Application détection d'intrusion	Spark, Zookepeer
Golmohammadi *et al.* (2019)	Proposer une architecture basée sur Hadoop qui intègre l'imagerie par résonance magnétique (IRM) non invasive, la spectroscopie par RM (SRM) ainsi que les résultats de tests neuropsychologiques pour identifier les diagnostics précoces de la maladie d'Alzheimer.	Application, santé	Hadoop
Higashino *et al.* (2016)	Présente CEPSim, un systèmes de traitement d'événements complexes dans des environnements cloud, un simulateur pour les systèmes les événements complexe et en steram dans les environnements en Cloud. Le CEPSim propose un modèle d'interrogation basé sur des graphes acycliques dirigés (DAG) et introduit un algorithme de simulation basé sur un nouveau abstraction.	Application, simulation	Storm

Référence	Contribution	Classification	Outils
Karunaratne *et al.* (2017)	Résoudre le problème du débit limité de la mise en cluster des flux par le développement des algorithmes basés sur le paradigme du micro-clustering qui fonctionnent sur des plateformes de cloud computing	Application, Cloud	Strom
Aufaure *et al.* (2016)	Propose la combinaison des travaux récents sur l'intelligence d'entreprise en temps réel avec la gestion de flux de données sémantiques	Application, sémantique	Spark
Anveshrithaa & Lavanya (2020)	Proposer un modèle de traitement des flux de données en temps réel pour la prévision du trafic des véhicules, ils utilisent les réseaux de mémoire longue et courte durée (LSTM) pour apprendre et se former à partir des données de trafic.	Application, trafic des véhicules, temps réels.	Kafka, Spark, mongoDb.
Otoo-Arthur & van Zyl (2020)	Proposer la mise en place d'un framework de données extensibles et adaptables pour l'apprentissage en ligne BiDeL (big data framework for e-learning). Afin d'améliorer l'apprentissage par l'utilisation de Spark.	Application, e-learning	Hadoop, Sqoop, Flume.
Basanta-Val *et al.* (2020)	Focalisé sur la définition d'un appel de procédure à distance prévisible (RPC) capable de tirer parti de la technologie de traitement des flux distribués. Ce type d'infrastructure permet des calculs parallèles efficaces, réduisant le temps de réponse effectif des invocations de bout en bout de façon linéaire avec le nombre de ressources affectées au système.	Application	Storm.
Hu (2020)	Étudier trois plateformes Apache, à savoir Hadoop, Spark et Storm. Il sélectionne les paramètres à l'aide de diverses approches de sélection des caractéristiques . Les résultats empiriques décrivent réduction significative du temps d'exécution des travaux de Hadoop et Spark et augmentation du nombre de tuples émis pour Storm, montrant ainsi l'optimisation des performances des plateformes de données.	Application, optimisation	Hadoop, Spark, Storm.

Référence	Contribution	Classification	Outils
Um *et al.* (2016)	propose un modèle de traitement d'événements complexes sémantiques pour le raisonnement dans le Big Data	Application, sémantique	Spark, Storm
Kousiouris *et al.* (2018)	Un Framework intégré de gestion du cycle de vie des informations permettant d'exploiter les données des réseaux sociaux pour identifier les événements dynamiques de grande concentration de population dans les applications des villes intelligentes	Application, Ville intelligente	Kafka, Spark
Munro & Clayton (2019)	Proposer une plate-forme de Big Data en temps réel (Kafka) pour les petits drones, le système d'exploitation du robot communiquant à l'aide de services mobiles cellulaires 4G., pour tester les algorithmes d'évitement d'obstacles et dévaluer les performances de communication de bout en bout.	application, drones	Kafka
Kovalchuk *et al.* (2018)	Présentés un système de décision pour ambulance et l'urgence des maladies cardiaque	Application,santé	Hadoop, Spark, Strom, Sql.
Wang *et al.* (2017a)	Analyse efficace du comportement des alarmes pour les réseaux de télécommunications	Application, réseaux de télécommunications	
Mavridis & Karatza (2017)	Étudier l'analyse des fichiers journaux dans le Cloud avec Spark et Hadoop	Application	Spark, Hadoop.

2.7 Conclusion

Dans ce chapitre, nous avons présenté l'architecture Big Data, l'historique et les concepts de base, l'écosystème, et finalement les différentes classifications de ces architectures. Dans les concepts de base, nous avons abordé leurs définitions, les caractéristiques de Big Data, les nomes X.V qui décrivent le volume, la vitesse, la variété, la valeur, la véracité et la vélocité. Ensuite nous avons présenté la spécification d'une architecture Big Data et les travaux des chercheurs pour répondre aux spécifications. Nous avons proposé des classifications pour ces architectures. De plus, nous avons présenté dans la dernière section les contributions récentes pour Big Data afin d'avoir une idée claire sur les pistes de recherche de Big Data. Dans le prochaine chapitre, nous présentons notre contribution sur les systèmes du Big Data.

Chapitre 3

Contribution sur les systèmes du Big Data

Sommaire

3.1 Introduction

Dans ce chapitre, nous présentons notre critique et notre contribution sur les architectures Big Data.Il s'agit de déterminer quelles sont les insuffisances de ces architectures. Comme nous avons déjà discuté dans le chapitre précédent, plusieurs architectures proposées dont l'objectif principal est la création d'une cadre conceptuelle permet de déterminer les besoins et bien exprimer les différents concepts de Big Data . Cependant, ces besoins ne discutent pas l'impact de la relation entre ces architectures avec l'apprentissage et Machine Learning, vue seulement comme un composant dans une architecture. A cet effet, et pour la 1ère contribution (Naoui *et al.*, 2020d), (Naoui *et al.*, 2020b), nous posons la question suivante :

— Existe-il une relation entre l'architecture et les Machines Learning en termes de précision et temps d'exécution ?

Pour répondre à cette question, nous proposons une architecture basée sur l'apprentissage profond distribué (Naoui *et al.*, 2020d). Pour évaluer l'impact d'une telle architecture, nous définissons quatre critères, à savoir, Erreur quadratique moyenne MSE (Mean Square Error), Erreur quadratique moyenne interne IMSE (Intern Mean Square Error), Temps d'exécution du cluster Cte (Cluster execution time) et Temps d'exécution global Gte (Global execution time).

D'autre part, aujourd'hui les sources des données multiples existent dans plusieurs régions. Les spécifications de Big Data cité en section 2.4.1 orientées vers la conception de Big Data. Pour la 2ème contribution (Naoui *et al.*, 2020c), (Naoui *et al.*, 2020a), nous posons la question de la réutilisation et l'exploitation de données Big Data. De plus, l'exploitation des Machines Learning existe dans un processus d'intégration de Big Data depuis diverses sources. Ce processus prend en compte l'apprentissage dans les Machines Learning, leurs fonctionnements et leurs avantages. Pour répondre à cette question de la 2ème contribution, nous avons choisi le domaine de la santé (Naoui *et al.*, 2020c). Les sources de données sont les réseaux sociaux et les données de radiographie des hôpitaux.

3.2 Contribution N°1

3.2.1 Introduction

Aujourd'hui, l'apprentissage approfondi dans le cadre du concept de ville intelligente est devenu un domaine de recherche intéressant. Il fournit des modèles qui peuvent aider les utilisateurs de ce système à prendre la meilleure décision. Chaque seconde de chaque minute, des données quotidiennes d'information sur les villes intelligentes sont saisies pour être analysées. Ces données se caractérisent par leur volume, leur variété et leur rapidité. Le Big Data est basé sur des données massives stockées dans les villes intelligentes. Il a produit de nouvelles méthodes pour le modèle d'apprentissage approfondi. Cependant, il pose des nouveaux défis aux approches traditionnelles. Cette contribution propose un algorithme d'apprentissage profond distribué pour les villes intelligentes (Naoui *et al.*, 2020d) afin de combler les approches traditionnelles dans les axes de taux de prédiction, réduire le temps d'exécution et d'effectuer des algorithmes plus performants. Une architecture multicouche est proposée pour décrire l'apprentissage profond distribué des villes intelligentes dans le système Big Data. Les composants du système proposé sont la couche ville intelligente, la couche Big Data et la couche apprentissage profond. La couche ville intelligente est responsable de la question des composants de la ville intelligente, de ses capteurs et effecteurs de l'internet des objets et de son intégration dans le système. La couche Big Data concerne les caractéristiques

des données et leur distribution dans le système. La couche apprentissage profond est le modèle du système proposé. Elle est responsable de l'analyse des données. L'architecture proposée est appliquée aussi dans les domaines de l'environnement intelligent et de l'énergie intelligente (Naoui *et al.*, 2020d), (Naoui *et al.*, 2020a). Dans le domaine de l'environnement intelligent, nous étudions la prévision du toluène à la ville intelligente de Madrid. Dans le domaine de l'énergie intelligente, nous étudions l'exploitation de l'énergie éolienne sur la ville de l'Australie. L'architecture proposée peut réduire le temps d'exécution et améliorer le modèle d'apprentissage profond tel que l'algorithme mémoire courte à long terme LSTM. L'architecture proposée peut fournir une vue claire de la ville intelligente, du stockage des données et de l'analyse. Les prévisions de toluène dans l'environnement intelligent aident le décideur à assurer la sécurité environnementale. L'énergie intelligente du modèle proposé peut fournir une prévision claire de la production d'énergie éolienne.

Les villes intelligentes sont devenues un domaine de recherche attrayant en raison des multiples domaines qu'elles introduisent, telles que les réseaux intelligents, les transports intelligents, les soins de santé intelligents et les modes de vie intelligents. Les principaux objectifs des villes intelligentes sont de rendre la vie plus indépendante, d'optimiser la consommation d'énergie, de rendre le réseau de transport plus flexible et plus efficace, et de garantir la qualité des soins de santé.

Dans les revues de la littérature, il existe plusieurs définitions des villes intelligentes. Albino *et al.* (2015) ont revu plus de 20 définitions du concept de ville intelligente. Ces définitions mettent l'accent sur l'utilisation des technologies de l'information basées sur la communication en réseau pour connecter entre eux des informations, des capteurs ou des effecteurs physiques, des objets intelligents, afin d'améliorer la qualité de vie dans multiples dimensions telles que la santé, les transports, l'efficacité énergétique, etc. Les principaux processus des villes intelligentes sont :

— Acquisition de données : ce processus indique la technologie utilisée pour connecter des dispositifs (capteur/effecteur/objet intelligent) en réseau (Gubbi *et al.*, 2013). Il permet de collecter des données ou de réaliser un événement.

— Stockage de données : les données collectées par internet des objets sont stockées dans le système.

— Analyse des données : il s'agit de l'analyse des données collectées.

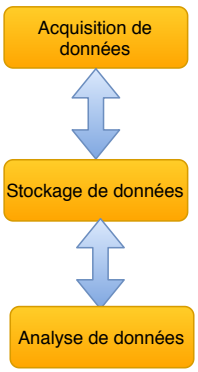

FIGURE 3.1 – Processus des données dans la ville intelligente

Les données des villes intelligentes jouent un rôle important ; elles constituent la base de l'analyse des données. Les données sont collectées auprès de multiples sources différentes et variables, en grande quantité, ce qui donne des données appelées Big Data. Les Big Data se caractérisent par une variété de volume et de vitesse. Le volume des données est

très important ; il peut être mesuré en pétaoctets ou en téraoctets. Certaines organisations quantifient les Big Data par leurs enregistrements, leurs transactions, leurs tableaux ou la durée d'utilisation des données (Russom *et al.*, 2011). La vitesse des données fait référence à la vitesse de génération des données. Elle peut être en lot, en temps quasi réel, en temps réel ou en flux (Russom *et al.*, 2011). La variété des données décrit les multiples sources de données. Elles peuvent être structurées, non structurées ou semi-structurées (Russom *et al.*, 2011). La complexité des Big Data est un défi, notamment en raison du problème de l'analyse des données, qui fait référence à l'utilisation d'outils d'apprentissage automatique pour analyser les Big Data dans les villes intelligentes. L'apprentissage profond fait référence aux méthodes et techniques utilisées pour décrire les modèles de données. Il peut décrire des problèmes de régression ou de classification. Plusieurs chercheurs proposent l'apprentissage profond pour analyser les villes intelligentes. Cependant, ces approches ne traitent pas des caractéristiques des données. Elles se concentrent uniquement sur l'analyse des données. Ainsi, ces approches restent également limitées. De plus, les questions clés de cette étude concernent l'architecture des villes intelligentes, qui peut faciliter l'analyse des Big Data et l'avantage d'apprentissage profond distribué dans les villes intelligentes pour les Big Data.

Les prochaines sections sont organisées comme suit. La section 3.2.2 présente plusieurs travaux connexes, tandis que la section 3.2.3 décrit l'approche proposée, qui repose sur une architecture multicouche. Ensuite, la section 3.2.4 présente la validation et l'expérimentation de l'architecture proposée. Enfin, les conclusions et les travaux futurs sont présentés dans la section 3.2.5.

3.2.2 Travaux connexes

Cette section présente les concepts des algorithmes de Deep Learning, le réseau neuronal à convolution et la mémoire à court terme, plusieurs stratégies de Deep Learning basées sur les Big Data, ainsi qu'un apprentissage machine pour les villes intelligentes.

3.2.2.1 Apprentissage profond

Les algorithmes d'apprentissage profond sont l'ensemble d'algorithmes d'apprentissage machine basés sur des réseaux neuronaux artificiels, qui sont composés de plusieurs couches de traitement. Ces couches peuvent apprendre des données avec plusieurs niveaux de représentation (LeCun *et al.*, 2015).

3.2.2.2 Réseaux de neurones à convolution

— Couche de convolution : le rôle de cette couche dans le traitement de l'image, par exemple, est la réduction de l'image en une forme qui peut être facilement traitée. Il est déterminé en multipliant les fonctions de convolution f et g.

$$(f * g) = \sum_{t} (f(t)g(x+t). \tag{3.1}$$

— Couche de mise en commun : le rôle de cette couche est de réduire la dimension de l'élément convulsé.

— Couche entièrement connectée : la sortie de la dernière couche de mise en commun est l'entrée de la couche entièrement connectée. Elle est responsable de la classification basée sur les couches de prévisualisation.

3.2.2.3 Algorithme mémoire courte à long terme LSTM (Long Short Term Memory)

La mémoire courte à long terme, introduite par Hochreiter (Hochreiter & Schmidhuber, 1997), est un cas particulier de réseau neuronal récurrent introduit par une cellule mémoire pour pouvoir apprendre les dépendances temporelles. La cellule (LSTM) est composée d'un temps t, d'un état Ct et d'une sortie (h_t). En entrée, cette cellule au temps (t) comprend (x_t), (C_{t1}) et (h_{t1}) . La fonctionnalité de LSTM peut être décrite à l'aide des équations suivantes :

$$u_t = \sigma(W^u h_{t-1} + I^u x_t + b^u) \tag{3.2}$$

Mettre à jour la porte (Update gate) H :

$$u_t = \sigma(W^u h_{t-1} + I^u x_t + b^u) \tag{3.3}$$

Oubliez la porte (Forget gate) H :

$$f_t = \sigma(W^f h_{t-1} + I^f x_t + b^f) \tag{3.4}$$

Candidat de cellule (Cell candidate H) :

$$\hat{C} = \tanh(W^C h_{t-1} + I^C x_t + b^C) \tag{3.5}$$

Porte de sortie (Output gate) :

$$o_t = \sigma(W^o h_{t-1} + I^o x_t + b^o) \tag{3.6}$$

3.2.2.4 Apprentissage machine pour les villes intelligentes

— Énergie intelligente : dans la revue de la littérature, il existe plusieurs études relatives aux villes intelligentes. Les domaines activés sont la prévision, l'optimisation et l'efficacité de la consommation d'énergie. Martínez-Álvarez *et al.* (2008) proposé une approche d'apprentissage machine basée sur un algorithme de regroupement pour analyser les données d'information sur l'électricité afin de prévoir le prix de l'électricité sur une courte période. Les auteurs ont utilisé la technique de k-means. Keyno *et al.* (2009) ont proposé un algorithme d'apprentissage machine pour prévoir la consommation d'énergie. Les auteurs ont appliqué un algorithme de regroupement aux données primaires et ont éliminé la variance périodique.

— Environnement intelligent : plusieurs ouvrages ont étudié le concept d'environnement intelligent. Reddy *et al.* (2018) proposé une approche Deep Air pour la prévision de la pollution atmosphérique à la ville de Pékin, en Chine. Les auteurs ont utilisé l'algorithme LSTM. Wang *et al.* (2018) ont proposé une approche hybride qui combine la logique floue, le regroupement semi-supervisé et la classification semi-supervisée pour la pollution de l'air à la ville de Athènes, en Grèce. Wang *et al.* (2017a) ont proposé un cadre hybride basé sur la prédiction spatiale (capturant les algorithmes d'apprentissage approfondi distribué pour les villes intelligentes dans un système de données étendu en utilisant CNN (Convolution Neuronal Network)) et prédicteur temporel (saisie de la relation temporelle en utilisant des séquences pour créer un modèle de séquence avec un mécanisme d'attention simplifié). Les auteurs ont validé l'approche proposée pour la prévision des PM2,5, PM10 et O3 à la ville de Pékin, en Chine.

3.2.2.5 Apprentissage automatique pour l'analyse de Big Data

Qiu *et al.* (2016) Ont étudié l'apprentissage automatique dans l'analyse des Big Data. Les auteurs ont mentionné des stratégies et des algorithmes possibles pour l'apprentissage avec grandes données. Le tableau 3.1 présent plusieurs études de recherche :

TABLEAU 3.1 – Apprentissage automatique pour l'analyse de Big Data

Strategies et Agorithmes	Descriptions	Auteur
Sélection par caractéristiques	Cet algorithme mesure la relation entre les caractéristiques pour réduire la complexité du processus d'analyse des données	(Bengio *et al.*, 2013; Huang & Yates, 2012; Tu & Sun, 2012; Li *et al.*, 2011; Huang & Yates, 2010; Bordes *et al.*, 2012; Dwivedi *et al.*, 2014)
Apprentissage approfondi	Réseau neuronal à convolution,Un réseau de croyance profond,	(Chen & Lin, 2014; Jones, 2014)
Transfert d'apprentissage	Transférer l'apprentissage : L'idée principale de l'apprentissage par transfert est la réutilisation d'un modèle préformé sur un nouveau problème.	(Pan & Yang, 2010; Xiang *et al.*, 2010; Zhang, 2011)
Apprentissage actif	Créer un modèle qui peut étiqueter les données automatique.	(Settles, 2014; Crawford *et al.*, 2013; Haque *et al.*, n.d.)
Apprentissage basé sur le noyau	Les données dans l'espace d'entrée sont projetées sur un l'espace dimensionnel des éléments via une fonction non linéaire	(Ding *et al.*, 2013; Montavon *et al.*, 2013; Slavakis *et al.*, 2009,0; Müller *et al.*, 2001)
Apprentissage distribué et parallèle	La majorité des méthodes proposées se sont concentrées sur la combinaison entre les classificateurs, par exemple les règles de décision, la généralisation par empilement, le méta-apprentissage et les algorithmes de renforcements .	(Chen *et al.*, 2014,0; Leyva *et al.*, 2015; Sarnovsky & Vronc, 2014)

3.2.3 Architecture proposée

Cette section présente notre approche proposée. Tout d'abord, on a décrit une architecture multicouche qui combine les villes intelligentes, l'apprentissage machine distribuée et le système Big Data. Ensuite, un modèle mathématique est formulé pour décrire l'apprentissage dans les machines learning distribuées, ses métriques (erreur quadratique moyenne, erreur quadratique moyenne interne, temps d'exécution du cluster, temps d'exécution global)

pour évaluer ce modèle. Enfin, la fonctionnalité de l'algorithme d'apprentissage profond est présentée dans le système.

3.2.3.1 Architecture multicouche

L'architecture proposée est composée de trois couches : la couche ville intelligente, la couche Big Data et la couche apprentissage profond.

— La couche ville intelligente : concerne la question des villes intelligentes telles que ses composantes intelligente : santé, environnement, énergie, sécurité, bâtiments, résidentiels, administration, transport intelligent et industries (figure 3.2). Un autre élément important de cette couche est l'internet des objets, qui sont des capteurs collectent des données dans la ville intelligente sur la base de la communication en réseau.

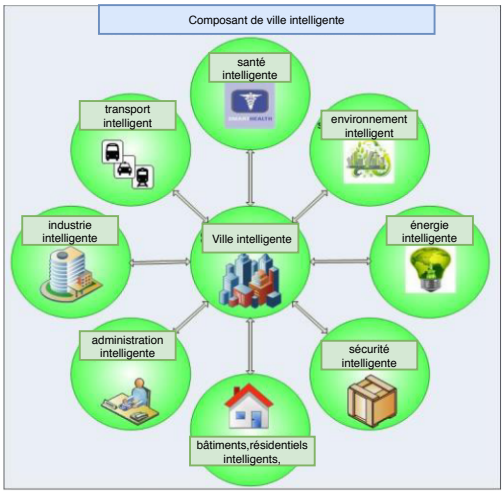

FIGURE 3.2 – Composantes de la ville intelligente (Gaur *et al.*, 2015)

— Couche Big Data : la ville intelligente génère une énorme quantité de données. Ces données sont caractérisées par leur volume, leur vitesse et leur variété. Par conséquent, un système Big Data est nécessaire et responsable du stockage et du traitement des Big Data.

— Couche d'apprentissage profond distribué : prend en considération tous les problèmes liés à l'apprentissage machine, ses types, ses modèles, etc. Les algorithmes d'apprentissage approfondi sont distribués sur le réseau de communication. Cette architecture distribuée est justifiée par les caractéristiques des Big Data qui peuvent collecter une quantité importante de données géographiquement distribuées. L'objectif de cette couche est d'analyser facilement la couche Big Data.

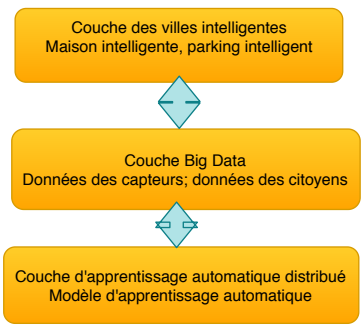

FIGURE 3.3 – Les couches de notre système (Naoui *et al.*, 2020d)

Dans l'architecture proposée, de l'apprentissage profond distribué, une architecture complète qui traite le processus d'apprentissage profond de l'acquisition, du stockage et de l'analyse des données a été proposée. L'effet Big Data au sein de l'architecture est dû aux raisons suivantes :

— Acquisition de données provenant de sources multiples, géographiquement réparties dans les villes intelligentes.

— Le stockage des données est réparti en plusieurs clusters.

— Le modèle d'apprentissage profond analyse les données de plusieurs clusters pour décrire ou prévoir les données.

3.2.3.2 Description mathématique

Soit (D) un ensemble de données $D = \{f(X_1, y_1), (X_2; y_2), (X_i; y_i)(X_{i+1}; y_{i+i})(X_m, y_m)\}$, où X_i sont les ensembles à n dimensions avec les valeurs connues associées y_i, pour une variable de réponse y, et m est le nombre d'ensembles dans D. Le modèle d'apprentissage approfondi est appliqué aux données D. Dans l'apprentissage profond distribué, les données complètes sont divisées en $D = \{D_1, D_2, D_n\}$. Soit C un ensemble de Cluster $C = \{C_1, C_2 ... C_n\}$ chaque Cluster C_j est utilisée pour stocker des données D_j provenant de plusieurs sources de villes intelligentes. Le modèle d'apprentissage profond M est appliqué pour les Clusters afin de créer une vision complète des données. Soit M_j = Modèle(C_j), qui renvoie le Modèle j des données D_j dans le cluster C_j.

Quatre mesures ont été définies pour évaluer le modèle proposé : Erreur quadratique moyenne (MSE), Erreur quadratique moyenne interne du cluster (IMSE), Temps d'exécution du cluster (Cte) et Temps d'exécution global (Gte).

— Erreur quadratique moyenne (MSE) : est calculée avec l'écart-type entre la valeur observée y et la valeur prédite y' des données D, comme suit :

$$MSE = \frac{\sum_{i=1}^{i=m} |y_i - {y'}_i|^2}{m} \tag{3.7}$$

— Erreur quadratique moyenne interne du cluster i (IMSE) : est calculée avec l'écart type des données $D_j = \{[(X_{k_1}, y_{k_1}), (X_{k_1+1}, y_{k_1+1}) ... (X_{k_2}, y_{k_2})]\}$, stockées dans le Cluster j entre la valeur observée y et les valeurs prédites y' .

$$IMSE = \frac{\sum_{k_1}^{k_2} |y_i - {y'}_i|^2}{k_2 + 1 - k_1} \tag{3.8}$$

— Temps d'exécution du cluster (Cte) : c'est le temps établi pour exécuter les tâches du cluster.

— Temps d'exécution global (Gte) : est le temps établi pour exécuter toutes les tâches dans le cas de plusieurs clusters fonctionnant en parallèle ; le temps global est, le cluster qui a le temps d'exécution maximum.

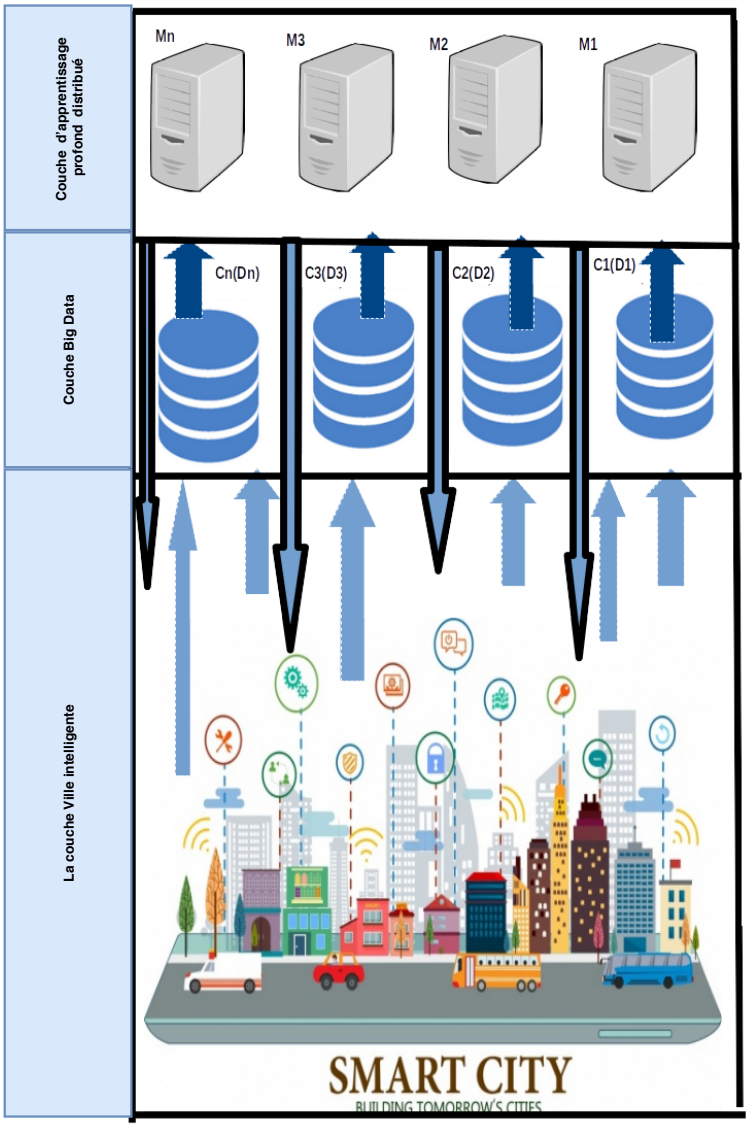

FIGURE 3.4 – Architecture de système (Naoui *et al.*, 2020d)

3.2.3.3 Algorithme d'apprentissage distribué

Algorihtme 7 Apprentissage approfondie distribue

Input : Ensemble de Clusters $=\{C_1, C_2, \ldots, C_n\}$, Ensemble de données dans les Clusters $= \{C_1(D_1), C_2(D_2), \ldots, C_n(D_n)\}$

Output : Ensemble de modèle $Model = \{Model_1, Model_2, \ldots, Model_n\}$ of Model

for $Each(C_i(D_i)$ **do**

  $Model_i \leftarrow C_i(Deeplearning(D_i))$

return $Model_i$

3.2.4 Validation expérimentale

Cette section présente l'expérimentation de l'architecture proposée. Tout d'abord, le paltforme de l'expérimentation est illustré dans la section 3.2.4.1. Ensuite, le modèle d'apprentissage approfondi utilisé dans cette expérimentation est présenté en 3.2.4.2. Les points 3.2.4.3 et 3.2.4.4 présentent les résultats du modèle d'apprentissage approfondi distribué

proposé. Ensuite, au point 3.2.4.5, l'architecture proposée est comparée avec trois modèles d'apprentissage : K plus proche voisin KNN (K Nears Neighbor), arbre de décision (CART) et base naïve (NB). Les paramètres de comparaison sont les paramètres cités au point 3.2.3.2, à savoir l'erreur quadratique moyenne (MSE), l'erreur quadratique moyenne interne (IMSE), le temps d'exécution du cluster (Cte) et le temps d'exécution global (Gte).

3.2.4.1 Framework d'expérimentation

Framework a été implémenté pour tester et valider l'approche proposée. Il est composé comme suit :

— Hadoop paltforme : Hadoop est l'un des paltformes les plus connus pour le système Big Data (Taylor, 2010).

— Python : un langage de programmation qui a beaucoup de succès pour le modèle d'apprentissage des Machines Learning.

— Deux machines ayant les mêmes caractéristiques (DELL INSPIRON N4050 - Intel Core i5-2450M/ 2,50 GHz - RAM 6 Go) ont été utilisées.

3.2.4.2 L'algorithme LSTM

L'algorithme LSTM a été appliqué en tant qu'algorithme d'apprentissage profond. L'architecture de LSTM est composée de trois couches : la couche d'entrée, 4 unités cachées et la couche de sortie.

3.2.4.3 Environnement intelligent

Le processus d'environnement intelligent (Smart Environment) pour la prévision du toluène a été étudié. Le niveau de toluène (méthyl-benzène) a été mesuré en $\mu g/m^3$. L'exposition à long terme à cette substance (présente également dans la fumée de tabac) peut entraîner des complications rénales ou des lésions cérébrales permanentes. Deux études de cas ont été utilisées. Le premier cas représente l'approche proposée ; les données ont été divisées en 2 Clusters et dans le second cas, un seul Cluster a été utilisé (tableau 3.2) (Naoui *et al.*, 2020d).

TABLEAU 3.2 – Cas d'étude, Clusters et données utilisés dans l'environnement intelligent

Cas d'étude	Clusters	Données utilisées
Premier cas d'étude (Case study 01)	Cluter1	Qualité de l'air de Madrid 2016
	Cluter2	Qualité de l'air de Madrid 2017
Deuxième cas d'étude (Case study 02)	un seul cluster	Qualité de l'air de Madrid 2016/2017

La figure 3.5 présente les valeurs réelles du toluène et la figure 3.6 illustre le modèle LSTM de l'algorithme d'apprentissage profond distribué proposé. Les résultats sont les suivants : Erreur quadratique moyenne interne (5,24), temps d'exécution du cluster (561,90) et temps d'exécution global (561,90).

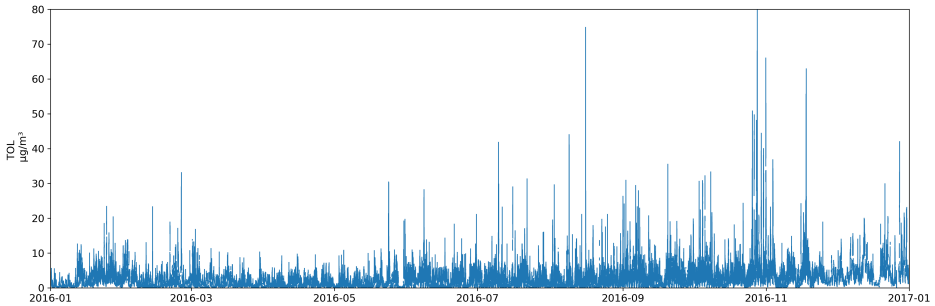

FIGURE 3.5 – Toluène à Madrid en 2016

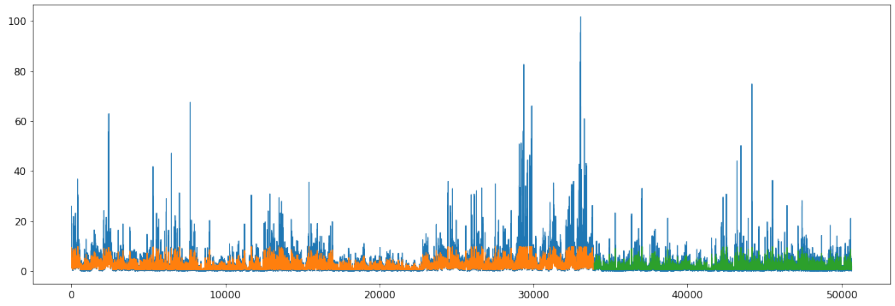

FIGURE 3.6 – LSTM Prévision du toluène en 2016

La figure 3.7 présente les valeurs réelles du toluène et la figure 3.8 illustre le modèle LSTM de l'algorithme d'apprentissage profond distribué proposé. Les résultats sont les suivants : Erreur quadratique moyenne interne (6.10), temps d'exécution du cluster (524.26) et temps d'exécution global (524.26).

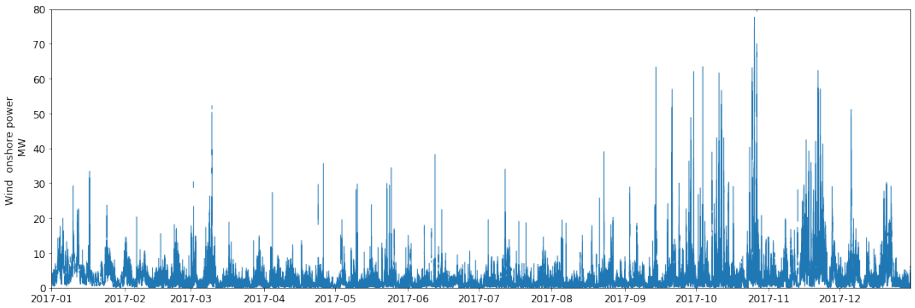

FIGURE 3.7 – Toluène à Madrid en 2017

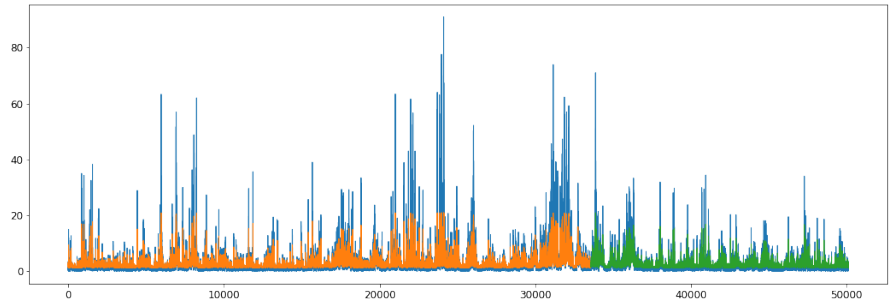

FIGURE 3.8 – LSTM Toluene prediction en 2017

3.2.4.4 Énergie intelligente

La prévision de la consommation d'énergie intelligente joue un rôle important dans les domaines de recherche. Elle offre des modèles qui peuvent aider les utilisateurs à choisir la meilleure décision. Chaque seconde et chaque minute des informations quotidiennes provenant des données énergétiques sont collectées pour être analysées. Les données sur la production d'énergie éolienne en Australie ont été utilisées. Le premier cas représente l'approche proposée. Les données ont été divisées en deux clusters : les données du premier Cluster (du 01-01-2015 au 31-3-2015) et celles du second Cluster (du 31-03-2015 au 2015-06-30). La seule Cluster (du 01-01-2015 au 2015-06-30) est présentée dans le tableau 3.3 (Naoui *et al.*, 2020d).

TABLEAU 3.3 – Cas d'étude, Clusters et données utilisés dans l'énergie intelligente

Cas d'étude	Clusters	Données utilisées
Premier cas d'étude (Case study 01)	Cluter1	Énergie éolienne en Australie (Du 01-01-2015 au 31-3-2015)
	Cluter2	Énergie éolienne en Australie (Du 31-03-2015 au 30-06-2015)
deuxième cas d'étude 02	seul cluster	Énergie éolienne en Australie(Du 01-01-2015 au 30-06-2015)

La figure 3.9 présente les valeurs réelles des prévisions d'énergie éolienne en Australie dans la Cluster 1 (du 01-01-2015 au 31-3-2015) et la figure 3.10 illustre le modèle LSTM de l'algorithme d'apprentissage approfondi distribué proposé. Les résultats sont les suivants : Erreur quadratique moyenne interne (12514.89), temps d'exécution du cluster (50.77) et temps d'exécution global (80.00).

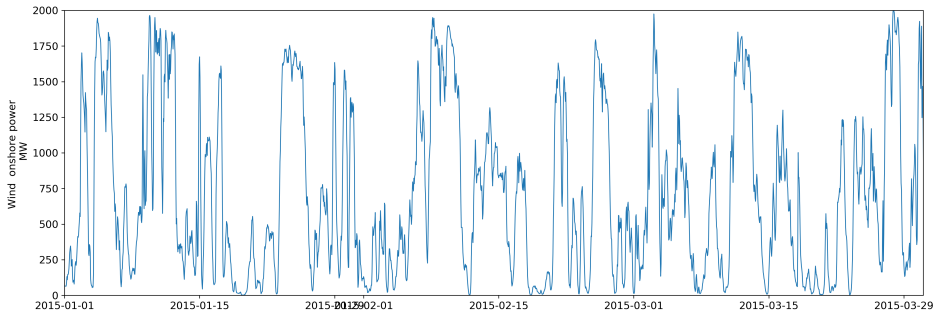

FIGURE 3.9 – Cluster1 :Énergie éolienne

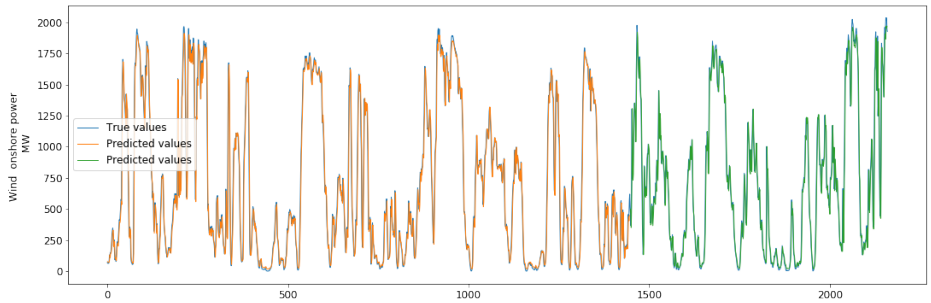

FIGURE 3.10 – Cluster1 : LSTM Prévision de l'énergie éolienne

La figure 3.11 présente les valeurs réelles des prévisions d'énergie éolienne en Australie (du 31-03-2015 au 2015-06-30) et la figure 3.12 illustre le modèle LSTM de notre proposition d'algorithme d'apprentissage approfondi distribué. Les résultats sont les suivants : Erreur quadratique moyenne interne (10976.75), temps d'exécution du cluster (80.00) et temps d'exécution global (80.00).

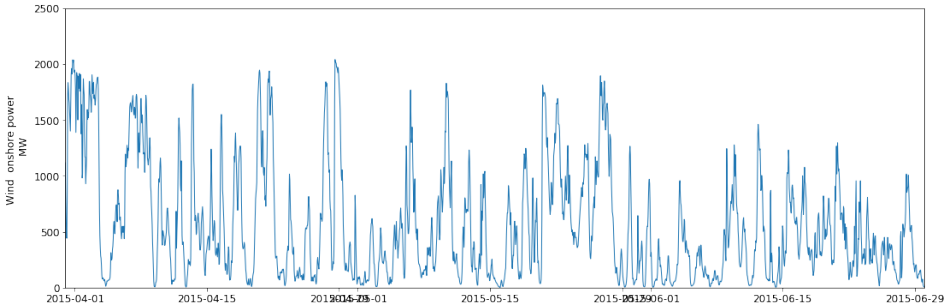

FIGURE 3.11 – Cluster2 :Production de l'énergie éolienne

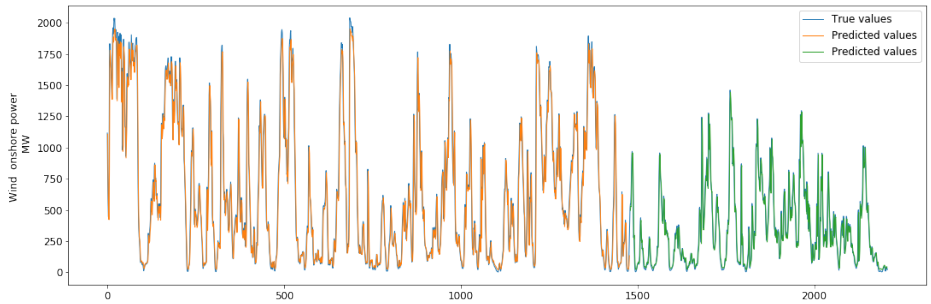

FIGURE 3.12 – Cluster2 :LSTM prévision de l'énergie éolienne

3.2.4.5 L'étude comparative

L'approche proposée est comparée avec trois machines d'apprentissage : le plus proche voisin (KNN), l'arbre de décision (CART) et la prédiction bayésienne (NB). Les paramètres proposés dans la partie 3.2.3.2 ont été calculés, à savoir l'erreur quadratique moyenne (MSE), l'erreur quadratique moyenne interne (IMSE), le temps d'execution des clusters en secondes (Cte) et le temps d'exécution global en secondes (Gte) (tableau 3.4 et tableau 3.5)(Naoui *et al.*, 2020d).

TABLEAU 3.4 – Résultats des modèles de Toluène

Clusters	Model	MSE	IMSE	CTe	GTe
Cluster1(Case study 01)	KNN	-	294.67	13.88	20.12
	CART	-	325.60	2.21	2.21
	NB	-	7497.23	3.10	3.17
	LSTM	-	5.24	561.90	561.90
Cluster2(Case study 01)	KNN	-	340.48	20.12	20.12
	CART	-	395.31	1.98	2.21
	NB	-	8598.44	3.17	3.17
	LSTM	-	6.10	524.26	561.90
Single cluster(Case study 02)	KNN	343.24	-	-	146.13
	CART	360.95	-	-	15.03
	NB	7185.87	-	-	15.66
	LSTM	16.97	-	-	969.72

TABLEAU 3.5 – Résultats des modèles d'énergie éolienne en ville d'Australie

Clusters	Model	MSE	IMSE	CTe	GTe
Cluster1(Case study 01)	KNN	-	663683.50	0.054	0.12
	CART	-	169767.67	0.26	1.28
	NB	-	792401.10	0.47	0.87
	LSTM	-	12514.89	50.77	80.00
Cluster2(Case study 01)	KNN	-	284094.33	0.12	0.12
	CART	-	149426.61	1.28	1.28
	NB	-	143428.25	0.873	0.87
	LSTM	-	10976.75	80.00	80.00
Single cluster(Case study 02)	KNN	365183.56	-	-	0.14
	CART	156986.54	-	-	1.71
	NB	176899.11	-	-	1.03
	LSTM	13284.86	-	-	87.56

a)Comparaison avec le temps d'exécution

Le temps d'exécution global dans environnement intelligent et dans énergie intelligente de chaque modèle KNN, CART, NB et LSTM est le minimum dans les premiers cas d'utilisation. Les données ont été divisées en deux Clusters afin d'exécuter leur tâche en moins de temps qu'un seul Cluster. Ce résultat démontre l'avantage de l'architecture distribuée proposée, dans laquelle les données sont divisées en Cluster. Cette approche permet de réduire le temps global d'exécution.

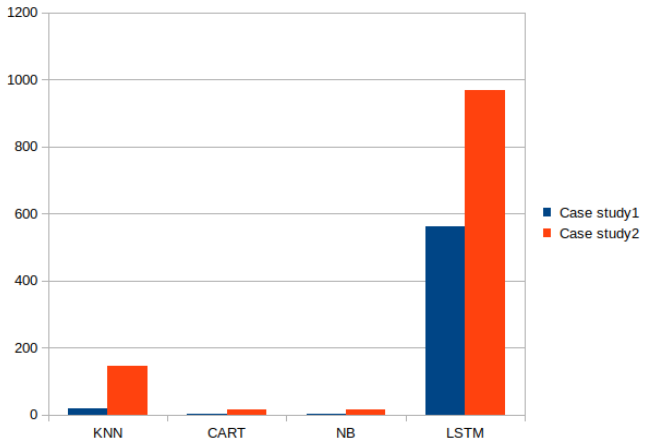

FIGURE 3.13 – Temps d'exécution global de l'environnement intelligent

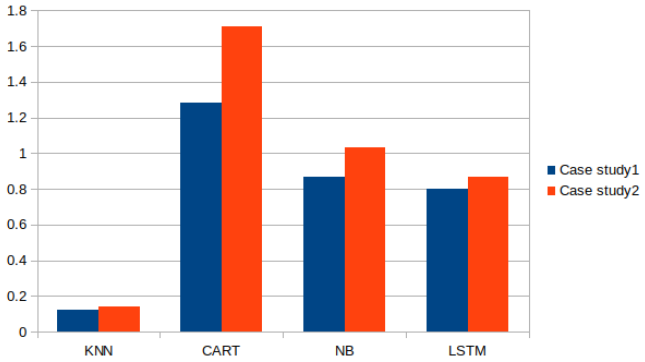

FIGURE 3.14 – Temps d'exécution global de l'énergie intelligente

b)Comparaison avec la capacité de prévisibilité

Le pouvoir de prévisibilité a été évalué par l'erreur quadratique moyenne et l'erreur quadratique moyenne interne. Des résultats différents ont été observés pour les modèles KNN, CART et NB. Par exemple, le modèle KNN dans l'environnement intelligent a enregistré les valeurs suivantes : Cluster1 (294.67), Cluster2 (340.48) et pour un seul Cluster (343.24). Dans l'énergie intelligente, le modèle KNN a enregistré les valeurs suivantes : Cluster1 (663683.50), Cluster2 (284094.33) et pour un seul Cluster (365183.56). Par conséquent, le modèle LSTM est le meilleur modèle et représente le minimum en matière d'environnement intelligent et énergie intelligente dans les différents cas d'utilisation. En outre, le modèle LSTM du cas 1 dans le domaine de l'environnement et de l'énergie intelligents est le minimum par rapport au modèle LSTM du deuxième cas (figure 3.15 et figure 3.16). Ce résultat illustre l'avantage du modèle d'apprentissage approfondi pour l'environnement et l'énergie intelligents. De plus, l'architecture de l'apprentissage approfondi distribué optimise le modèle LSTM.

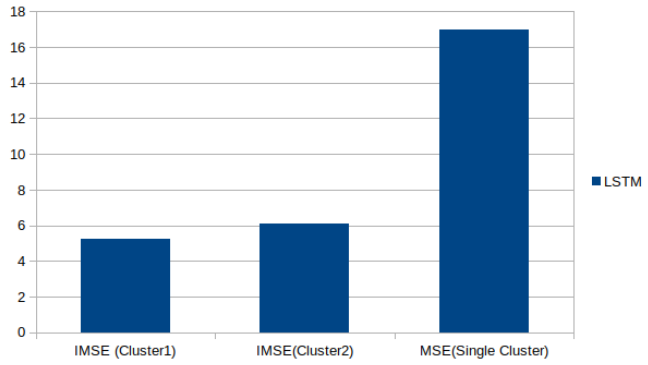

FIGURE 3.15 – La comparaison LSTM entre le MSE et l'IMSE de l'environnement intelligents.

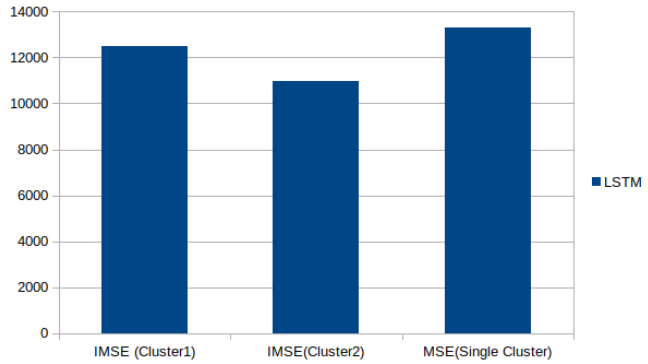

FIGURE 3.16 – La comparaison LSTM entre le MSE et l'IMSE de l'énergie intelligents.

3.2.4.6 Résultats et discussion

Les avantages de l'architecture proposée ont été démontrés selon trois axes :

— Le temps d'exécution global : l'architecture proposée peut réduire le temps d'exécution pour analyser les Big Data dans les villes intelligentes. Elle peut optimiser le temps d'exécution pour les Big Data caractérisés par leur énorme volume de données.

— Meilleur modèle : l'architecture proposée démontre l'importance du modèle d'apprentissage approfondi (LSTM) pour les prévisions énergétiques et environnementales des villes intelligentes.

— Optimisation du modèle : l'architecture proposée peut optimiser et améliorer le modèle d'apprentissage approfondi (LSTM).

3.2.5 Conclusion sur la contribution N°1

Le domaine de recherche des villes intelligentes est important. Plusieurs projets de recherche ont proposé une architecture pour aborder les processus dans les villes intelligentes. En conséquence, notre contribution introduit une architecture distribuée pour un apprentissage approfondi basé sur les Big Data dans les villes intelligentes. L'architecture proposée aborde le processus d'exploration des données dans les villes intelligentes, en particulier l'acquisition, le stockage et l'analyse des données. L'architecture proposée traite les Big Data

dans les villes intelligentes qui sont géographiquement distribuées et caractérisées par leur variété de volume et de vitesse. Par conséquent, des Clusters distribués ont été proposés pour stocker les données. Le modèle d'apprentissage profond peut entraîner les données distribuées à construire le meilleur modèle. Un framework a été développé pour valider l'importance de l'approche proposée. Les composantes de ce framework étaient Hadoop pour le stockage des données et le langage de programmation Python pour développer un modèle d'apprentissage. L'architecture proposée a été appliquée dans l'environnement intelligent pour créer des modèles qui décrivent les prévisions du toluène et l'énergie intelligente pour décrire les prévisions de l'énergie éolienne. Enfin, deux cas ont été comparés ; dans le premier cas, deux Clusters ont été utilisés et dans l'autre cas, une seule Cluster a été utilisée. La comparaison entre l'erreur quadratique moyenne interne et l'erreur quadratique moyenne et le temps d'exécution du cas d'utilisation du taux d'apprentissage a démontré l'importance de l'architecture proposée. Sur la base des résultats de cette étude, dans le cadre de travaux futurs, un autre algorithme d'apprentissage profond tel que le réseau neuronal de convolution et l'auto-codeur sera appliqué.

3.3 Contribution N°2

3.3.1 Introduction

Cette contribution vise à proposer un modèle d'apprentissage approfondi basé sur le Big Data pour le système de santé afin de prédire les données des réseaux sociaux . Les utilisateurs des réseaux sociaux publient quotidiennement de grandes quantités d'informations sur les soins de santé et en même temps, les hôpitaux et les laboratoires médicaux stockent de très grandes quantités de données sur les soins de santé, comme les radiographies.

Nous proposons une architecture qui peut intégrer l'apprentissage profond, les réseaux sociaux et les Big Data. L'apprentissage profond est l'un des domaines de recherche les plus difficiles, et il est de plus en plus populaire dans le secteur de la santé. Il utilise une analyse approfondie pour extraire les connaissances avec une précision optimale. L'architecture proposée se compose de trois couches : la couche d'apprentissage profond, la couche Big Data et la couche des réseaux sociaux. La couche Big Data comprend des données pour les soins de santé, tels que les radiographies. Pour la couche d'apprentissage profond, nous proposons trois modèles de réseau neuronal à convolution sont proposé pour la classification des images radiographiques.

Les systèmes de santé sont apparus comme un domaine d'intérêt pour la recherche, notamment en ce qui concerne les montants importants de données hétérogènes sur les soins de santé. Ces dernières années, les données sur les soins de santé sont devenues de plus en plus complexes parce qu'il existe de nombreux types de données, telles que les données structurées, semi-structurées et non structurées. Il existe également de multiples sources de données sur les soins de santé, telles que les réseaux sociaux, les laboratoires, les médicaments, les instruments, etc. En conséquence, le volume d'informations stockées dans le système de santé est en constante augmentation et les logiciels traditionnels ont du mal à le gérer. Une architecture de données de Big Data peut améliorer le système de santé en fournissant un cadre idéal pour les données, en renvoyant à des données complexes et volumineuses qui peuvent facilement gérer des données volumineuses provenant de sources multiples et offrant une diversité de types de données. Plusieurs études ont porté sur l'utilisation des réseaux sociaux dans le système de santé, pour diverses raisons (Schein *et al.*, 2011) :

— Recrutement pour les essais cliniques.

— Le développement professionnel et la formation des cliniciens.
— Communication et coordination interprofessionnelles.
— Défense de la santé et collecte de fonds pour les organisations de santé.
— Développement d'outils interactifs d'autogestion et de plugin pour les plateformes de médias sociaux populaires.
— Messages de santé publique.
— Surveillance des maladies infectieuses.

Dans cette contribution, nous proposons un modèle d'apprentissage approfondi des problèmes de prédiction des soins de santé dans les réseaux sociaux. L'architecture de ce modèle combine l'apprentissage profond, Big Data et les réseaux sociaux. En outre, la méthode proposée vise à répondre aux questions suivantes :

— Comment stocker les données importantes pour le système de santé dans un système distribué et hétérogène ?.
— Quel est le modèle qui peut décrire les données du système de santé ?.
— Comment les utilisateurs de réseaux sociaux peuvent-ils prévoir ses données, par exemple une image radiographique ?

La présentation de la contribution est structurée comme suit : dans la section 3.3.2, les travaux connexes qui décrivent les différents types de recherche dans le domaine des réseaux sociaux, des soins de santé et de l'apprentissage automatique. La section 3.3.3, présente l'approche proposée. La section 3.3.4, présente la validation du système. Les résultats sont comparés dans la section 3.3.5. Enfin, dans la section 3.3.6, les futurs travaux et une brève conclusion de cette étude de recherche et les perspectives pour les travaux futurs sont présentés.

3.3.2 Travaux connexes

3.3.2.1 Réseaux sociaux et sites pour le secteur de la santé

Aujourd'hui, il existe plusieurs sites et réseaux sociaux dédiés au secteur de la santé. Le réseau Figure1 est un site de réseau social où les utilisateurs peuvent partager leurs connaissances. Par exemple, les utilisateurs peuvent afficher une image radiographique et d'autres utilisateurs peuvent commenter cette image. Open-iBiomedical image est un site contenant des milliers d'images médicales avec toutes les informations sur la maladie et l'âge du patient, son état, son diagnostic et son traitement. C'est l'un des sites les plus importants utilisés pour manipuler du texte et des images.

MedPix est une base de données en ligne gratuite avec un accès libre aux images médicales, aux sujets cliniques, à la fusion d'images, et des métadonnées textuelles comprenant plus de 12 000 scénarios de cas de patients, 9 000 thèmes et environ 59 000 images. Son principal public cible comprend les médecins, les infirmières, les étudiants en médecine et les étudiants en soins infirmiers.

3.3.2.2 Les données des réseaux sociaux pour la prévision des soins de santé

Plusieurs études ont examiné les données recueillies à partir des réseaux sociaux de prédiction des maladies, de l'athérosclérose cardiaque de la santé publique (Achrekar *et al.*, 2011), ou la propagation de la maladie infectieuse (Hirose & Wang, 2012). Ces recherches recueillent des données provenant des réseaux sociaux afin de construire un modèle permettant de prédire les maladies.

3.3.2.3 L'exploration des données pour les soins de santé

Certaines études ont abordé les défis de l'exploration des données dans le domaine des soins de santé (Koh *et al.*, 2011; Bellazzi & Zupan, 2008; Canlas, 2009; Hosseinkhah *et al.*, 2009) tandis que d'autres se sont concentrés sur la prévision des maladies (Kumari & Godara, 2011; Dangare & Apte, 2012; Gupta *et al.*, 2011). En outre, plusieurs études ont proposé des modèles et des machines learning pour le système de santé, tel que l'arbre de décision (Khan *et al.*, 2008; Moon *et al.*, 2012; Chien & Pottie, 2012; Chang & Chen, 2009), la machine à vecteurs de distance (Fei, 2010; Huang *et al.*, 2008; Avci, 2009; Abdi & Giveki, 2013) les réseaux neuronaux (Er *et al.*, 2010; Das *et al.*, 2009; Gunasundari & Baskar, 2009) la méthode bayésienne (Chien & Pottie, 2012; Liu & Lu, 2009) la régression (Gennings *et al.*, 2012; Agarwal *et al.*, 2011), Clustering (Tapia *et al.*, 2009; Escudero *et al.*, 2011; Chipman & Tibshirani, 2006; Chen *et al.*, 2005; Belciug, 2009; Celebi *et al.*, 2005), ou l'algorithme Apriori (Ji *et al.*, 2011; Soni & Vyas, 2010). De plus, plusieurs auteurs ont souligné les avantages de data mining dans le secteur de la santé. Parmi ceux-ci, on peut citer la disponibilité de solutions médicales pour les patients à moindre coût ; la détection et la prédiction de la cause des maladies et l'identification des méthodes de traitement médical ; et un soutien efficace à la prise de décision en matière de santé (Koh *et al.*, 2011).

3.3.2.4 Big Data pour les soins de santé

Les caractéristiques de Big Data sont connues sous le nom de XV, à savoir le volume, la variété, la vitesse, la véracité et la valeur. Ainsi, un modèle basé sur Big Data est un modèle idéal pour le système de santé. Par conséquent, plusieurs études ont examiné le volume des données dans le système de santé, telles que les informations personnelles, les images radiologiques, l'imagerie 3D, les génomiques, le traitement des signaux et la lecture des capteurs biométriques (Widmer *et al.*, 2014; Van Essen *et al.*, 2012). Il existe différents types de données : structurées, semi-structurées et non structurées. La vitesse fait référence à la vitesse à laquelle les données sont générées, par exemple, les données de radiographie. La véracité dans le système de santé signifie que le Big Data fournit certificat sur les diagnostics/traitements. La valeur fait référence à la qualité d'information.

3.3.2.5 Apprentissage approfondi dans le domaine des soins de santé

Les méthodes traditionnelles d'exploration de données tendent à favoriser les données simples et structurées, qui peuvent ne pas être en mesure d'utiliser efficacement la richesse des informations contenues dans les données sur les soins de santé. Le dernier paradigme en matière de technologies d'apprentissage approfondi permet d'extraire efficacement des informations de sources complexes et hétérogènes. Plusieurs recherches dans le domaine d'apprentissage approfondi pour les soins de santé ont proposé un modèle d'apprentissage des données sur les soins de santé pour le diagnostic du cancer (Fakoor *et al.*, 2013), la conception de médicaments (Miotto *et al.*, 2016), la prévision des maladies, suivi du comportement humain (Lipton *et al.*, 2015), et les maladies liées au mode de vie (Phan *et al.*, 2015).

3.3.3 Proposition

3.3.3.1 Architecture de système

La Figure 3.17 présente l'architecture UML(Uniform Modeling Language) du système proposé. Elle est composée de trois couches : une couche Big Data, une couche d'apprentissage profond, et une couche de réseau social.

— La couche Big Data : cette couche est responsable du stockage et de la gestion des données sur la santé. Elle offre aux utilisateurs un framework pour les données qui se caractérise par le volume, la variété, la vitesse, la véracité et la valeur.
— La couche d'apprentissage profond : cette couche offre aux utilisateurs, multiples algorithmes d'apprentissage profond. Elle analyse les données stockées dans la couche Big Data pour fournir des résultats approfondis aux utilisateurs du système.
— La couche réseaux sociaux : elle est le responsable des réseaux sociaux en tant que source d'informations sur les soins de santé. Il existe plusieurs réseaux sociaux qui peuvent être classés en deux catégories selon les informations sur les soins de santé, les réseaux sociaux généraux : contenant des informations générales sur les soins de santé, par exemple, Facebook, Twitter, etc. et les réseaux sociaux sur les soins de santé : qui ne contiennent que des informations sur les soins de santé, par exemple, le site Figure1.

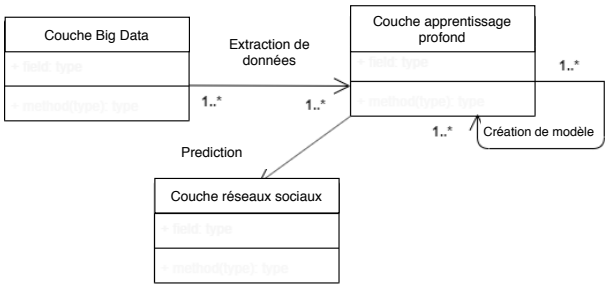

FIGURE 3.17 – Diagramme UML

La Figure 3.18 présente les composantes de l'architecture proposée (Naoui *et al.*, 2020d). Elle est composée de :

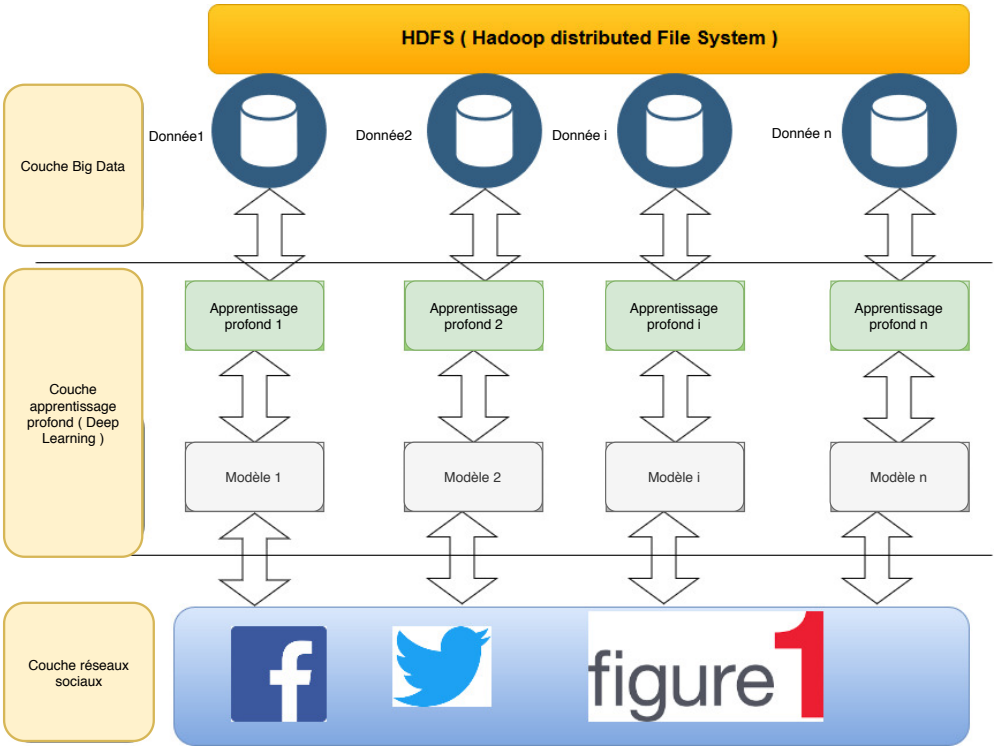

FIGURE 3.18 – Architecture de notre système

— Fichier HDFS (Hadoop Distributed File System) : est un système de fichiers distribués utilisé pour stocker Big Data fonctionnant sur des Clusters de manière fiable. Le HDFS

permet une détection/récupération rapide et automatique des pannes, prend en charge l'accès continu aux ensembles de données et facilite le traitement de grands ensembles des données à partir de la plate-forme hétérogène.

— Données $i \in [1.n]$: elle représente les données stockées de Big Data , les données de soins de santé provenant des données cliniques des hôpitaux, etc.

— Algorithmes d'apprentissage profond $i \in [1.n]$: se sont des tâches d'algorithme d'apprentissage profond.

— Le modèle $i \in [1.n]$: le modèle formé construit avec un algorithme d'apprentissage profond.

— Les réseaux sociaux : les utilisateurs de médias sociaux se connectent au modèle pour prévoir ses données, par exemple, une classification d'une image radiographique.

3.3.3.2 Cas d'utilisation (prédiction de l'image radiographique)

Cette section présente l'étude de cas d'une image radiographique, d'un ensemble de données et plusieurs algorithmes d'apprentissage profond basés sur le réseau neuronal de convolution (CNN) pour prédire une image radiographique. L'algorithme CNN est l'un des algorithmes les plus importants de classification des images. Il est donc appliqué pour classer les maladies à l'aide des images radiographiques. Il existe de multiples et grandes sources d'images, c'est pourquoi les outils Big Data sont proposés pour les stocker et les gérer. Les chercheurs ont identifié de nombreux problèmes dans le diagnostic des images radiologiques, soit en raison de la diversité des diagnostics, soit de la difficulté du diagnostic lui-même. Par conséquent, un apprentissage approfondi peut aider à relever ces défis, car les algorithmes traditionnels ne suffisent plus à les résoudre. Il constitue également un complément précieux dans le domaine médical et aide également les médecins et leurs patients. En outre, un apprentissage approfondi peut aider les utilisateurs à résoudre la confusion qu'entraîne un mauvais diagnostic sur les images radiographiques. L'étude des images radiographiques a pour but d'analyser l'énorme ensemble de données du système de santé. Par exemple, l'institut de la santé américaine NIH (National Insttitu of Health) publiée des milliers d'images en octobre 2017. De plus, les nombreuses maladies pulmonaires et la grande convergence entre elles rendent difficile une personne non compétente pour les distinguer. Ainsi, cette étude utilise les images radiographiques de la radiographie thoracique14, Shenzhen et l'ensemble des données radiologiques du comté de Montgomery.

3.3.3.3 Description de l'ensemble de données

L'institut de la santé américaine NIH est un organisme gouvernemental des institutions américaines qui sont impliquées dans la recherche médicale et biomédicale. Elles dépendent du département de la santé et des services sociaux des États-Unis (Wang *et al.*, 2017b). En octobre 2017, l'institut de santé américaine a ouvert les sources d'images de radiographie pulmonaire nommées Radiographie du thorax14. Les images ont été publiées pour permettre aux médecins de prendre de meilleures décisions de diagnostic pour les patients atteints de maladie pulmonaire 112.120 radiographies pulmonaires PNG (Portable Network Graphics) en 1024 × 1024 résolutions (Wang *et al.*, 2017b). La radiographie du thorax14 comprend 14 maladies : Atélectasie, cardiomégalie, épanchement, infiltration, masse, nodule, pneumonie, pneumothorax, épaississement pleural, consolidation, emphysème, hernie, fibrose et œdèmes.

Radiographie du comté de Montgomery : les images radiographiques de cette collection de données ont été recueillies par Shenzhen Hospital, dans la province de Guangdong de la Chine. Des radiographies ont été obtenues dans le cadre des soins de routine à Shenzhen

Hôpital. La collection contient des images JPEG. Il y a 340 images radiographiques normales et 275 images radiographiques anormales (Jaeger *et al.*, 2014). Les données de Montgomery sont également divisées en deux catégories : normales et anormales.

3.3.3.4 Couche Big Data

Dans la couche Big Data, les images sont stockées dans les nœuds HDFS. L'apprentissage profond s'entraîne sur les données pour créer un modèle. La figure 3.19 montre les composantes de la couche Big Data (Naoui *et al.*, 2020d).

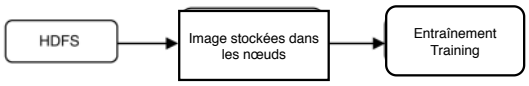

FIGURE 3.19 – Données d'image basées sur le système de fichiers distribué Hadoop

3.3.3.5 Couche d'apprentissage approfondie

Récemment, les modèles CNN ont été très efficaces pour reconnaître et classer les images dans les Big Data en raison de leur capacité d'utiliser des algorithmes de formation hautement évolutifs. Ces algorithmes sont utilisés pour fournir un apprentissage approfondi pour les images dans les réseaux sociaux. Par conséquent, l'apprentissage approfondi peut être utilisé pour analyser les images d'IRM (Magnetic Resonance Imaging) et de radiographie, pour aider diagnostiqué et établir un pronostic. CNN est un réseau de feed-forward et se compose de trois couches : une couche de convolution, couche de mise en commun, et une couche entièrement de connexion (LeCun *et al.*, 1990). Le tableau 3.6 résume les trois modèles CNN proposés pour les problèmes de classification :

TABLEAU 3.6 – Modèle et classification

Modèles	Classification
CNN1	Images radiographiques thoraciques / radiographiques non thoraciques / autres images radiographiques.
CNN2	Normal/ Anormal
CNN3	Classification des maladies

Le modèle CNN1 : ce modèles le plus simple des modèles de classification. La différence entre les images est facile à identifier. L'entrée est une image et la sortie est une classe d'image, qu'il s'agissent d'une radiographie du thorax (ChestX ray), et une radiographie du non thorax (No ChestX ray), ou autre image radiographique (Other X-ray). Six couches de réseau neuronal sont proposées : 2 couches de convolution, 2 couches de regroupement maximum (Max Pooling), et 2 couches entièrement connectées (fully connected) avec 32 matrices de filtrage. La fonction d'activation ReLu est utilisée à la fin de l'analyse neurale avec un taux d'apprentissage de 10e-4. La figure 3.20 présente le modèle l'architecture du modèle CNN1.

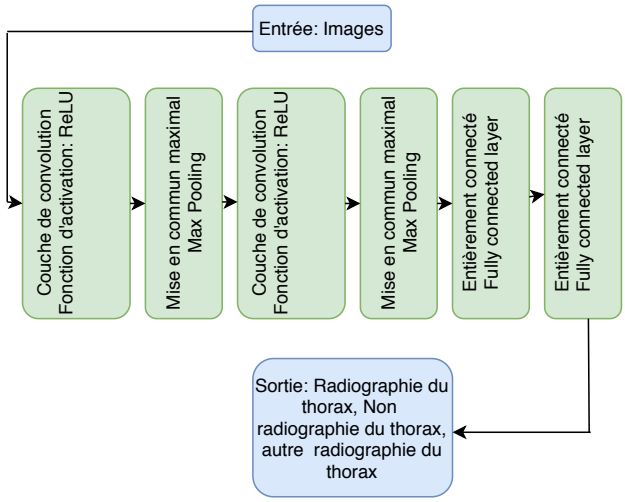

FIGURE 3.20 – Architecture du modèle CNN1

Le modèle CNN2 : pour ce modèle, l'entrée est une image radiographique et la sortie est l'état de santé de l'image : normal ou anormal. Ce modèle CNN est constitué d'un plus grand nombre de couches que l'algorithme précédent : 3 couches de convolution, 3 couches de regroupement moyen (average Pooling), et 3 couches entièrement connectées (fully connected) avec une augmentation du nombre de filtres (jusqu'à 128 filtres). Le modèle a été testé en premier lieu à un taux d'apprentissage de 10e-4 et et en deuxième lieu à un taux d'apprentissage de 10e-5. La figure 3.21 présente l'architecture du modèle CNN2.

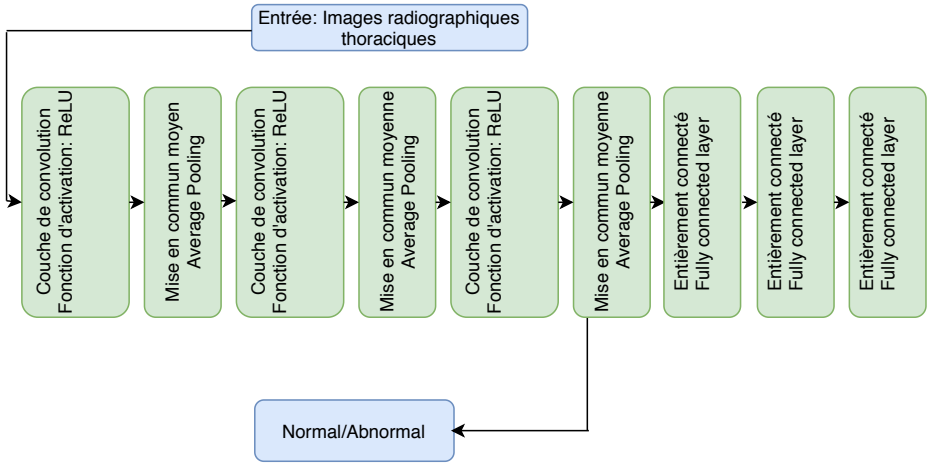

FIGURE 3.21 – Architecture du modèle CNN 2

Le modèle CNN3 : c'est le modèle le plus difficile ; il nécessite un plus grand nombre de couches et de filtres. L'entrée est une image radiographie de l'état d'un patient (anormal) et la sortie est le nom de la maladie. Elle comporte 9 couches et 128 filtres, pour un taux d'apprentissage de 10e-5, en utilisant la fonction ReLu. La figure 3.22 présente l'architecture du modèle de CNN3.

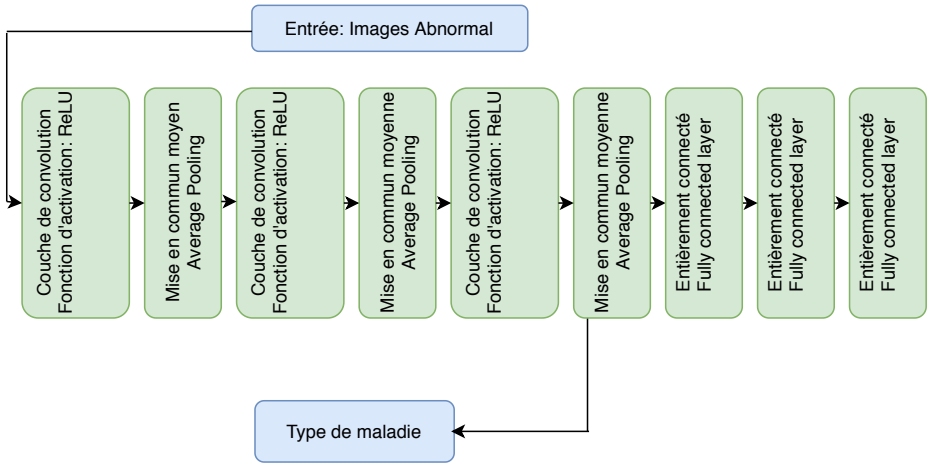

FIGURE 3.22 – Architecture du modèle CNN 3

3.3.3.6 La couche des réseaux sociaux

Twitter est l'un des sites de réseau social les plus populaires. Pour cette étude, le système a téléchargé les images publiées dans les tweets. Ensuite, le modèle a été testé pour répondre dans le commentaire de Twitter de la classe d'images. La figure 3.23 montre le processus de prédiction de la couche des médias sociaux.

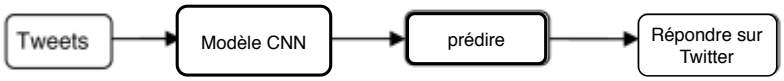

FIGURE 3.23 – Processus de prédiction de la couche de médias sociaux

3.3.4 Validation du modèle proposé

Le modèle proposé a été validé avec Python Hadoop et TensorFlow. Python a accès à des bibliothèques ; il est donc le plus utilisé dans le domaine de l'intelligence artificielle et de l'apprentissage machine. TensorFlow, Tweepy et Pyspark sont tous les deux supportés par Python. Ses interpréteurs sont disponibles pour d'autres systèmes d'exploitation. Les bibliothèques les plus connues dans le domaine de l'intelligence artificielle sont des bibliothèques Python intégrées, telles que TensorFlow, Scikit-learn, et Keras. TensorFlow a été créé pour utiliser le langage Python, bien qu'il existe d'autres programmes utilisés dans ces bibliothèques, mais pas aussi efficaces que Python (Raschka & Mirjalili, 2017). Hadoop est une plateforme logicielle open-source écrite en Java pour stocker et manipuler des données massives dans un format distribué. Par exemple en stockant des données énormes sur plusieurs serveurs et en distribuant ensuite le processus de traitement à ces appareils pour accélérer le résultat du traitement (Grover *et al.*, 2015). TensorFlow, développé par Google et publié en 2015, est un cadre d'apprentissage machine open-source pour le calcul numérique haute performance. De nombreuses entreprises utilisent actuellement TensorFlow comme IBM, Twitter, Intel, Nvidia (Abadi *et al.*, 2016). Dans cette étude, le TensorFlow a été utilisé pour trois algorithmes CNN afin de déterminer le type d'image (radiographie du thorax, pas une radiographie du thorax, une autre radiographie) ; pour déterminer si une personne est infectée ou non (normale ou anormale) ; et pour identifier le type de maladie. Tout d'abord, le CNN1 a été formé pour déterminer le type d'image (radiographie du thorax, pas de radiographie

du thorax, autre radiographie). Ensuite, le CNN2 a été formé à l'aide de radiographies pulmonaires des ensembles de données afin de déterminer si la personne est infectée ou non (normale ou anormale). Enfin, le CNN3 a été formé sur la base de l'ensemble des données de l'institut NIH qui contient environ 112 000 radiographie divisées en 14 classes. Ainsi, TensorFlow a été utilisé pour identifier le type de la maladie. La figure 3.24 présente le processus de prédiction des images.

En outre, le système proposé (Naoui *et al.*, 2020a) montre comment obtenir des images de Twitter via l'API Twitter développée par la même société qui fournit un accès complet aux tweets et aux images. Une fois quelqu'un publie une image de radiographie pulmonaire, il est enregistré, son taux d'infection est obtenu, le type de maladie est déterminé, et le tweet est ensuite répondu avec un commentaire. Les images sont stockées dans plusieurs nœuds conçus par Hadoop, chacun d'eux contenant un large quantité d'images, et le système proposé permettra d'accéder à ces images et de les former pour obtenir le modèle.

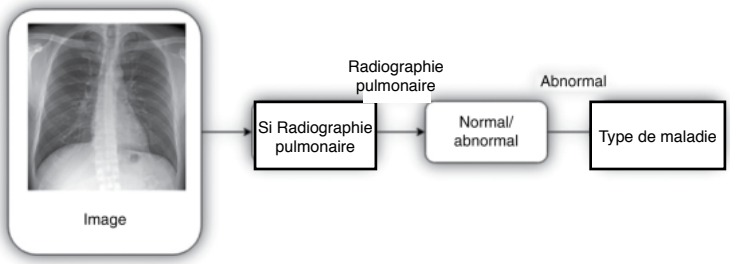

FIGURE 3.24 – Processus de prédiction d'images par le système proposé

Le tableau 3.7 résume les paramètres utilisés pour chaque modèle CNN. La figure 3.25 présent la précision d'entraînement TensorBoard de la classification pour CNN1 et la figure 3.26 pour CNN2.

TABLEAU 3.7 – Paramètre de CNN

CNN	CNN1	CNN2	CNN3
Taille du lot (Batch size)	16	16	16
Taille de l'image (Image size)	256	300	300
Nombre de fichiers dans l'ensemble de formation	2500	2500	112,000
Nombre de couches (Number of layers)	6	9	9
Nombre de fichiers dans l'ensemble de validation	600	500	12000
Nombre d'époque(Number of epoch)	30	20	20
Rythme d'apprentissage (learning rate)	10e-4	10e-4	10e-5
Fonction d'activation (Activation function)	ReLu	ReLu	ReLu

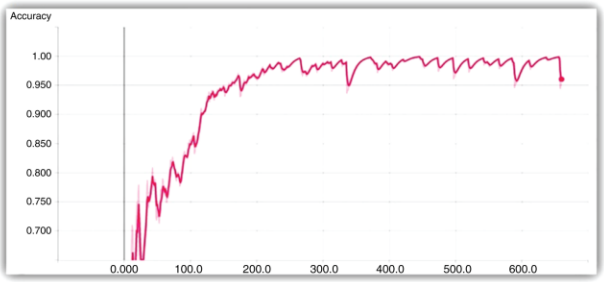

FIGURE 3.25 – Précision d'entraînement TensorBoard de la classification du CNN1

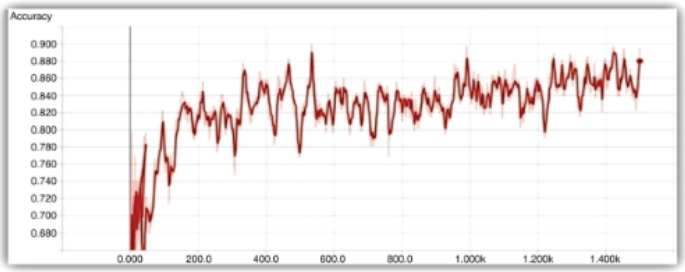

FIGURE 3.26 – Précision d'entraînement TensorBoard de la classification du CNN2

La figure 3.27 montre trois images différentes qui ont été utilisées pour tester le résultat du CNN1. Dans la première image, la valeur de précision de la radiographie du thorax (Chest X ray) est de 99,519%. Pour la deuxième image, pas de radiographie du thorax (No X ray), elle est de 99,97%. Pour la troisième image autre radiographie (Other X ray), la précision est de 99,03%.

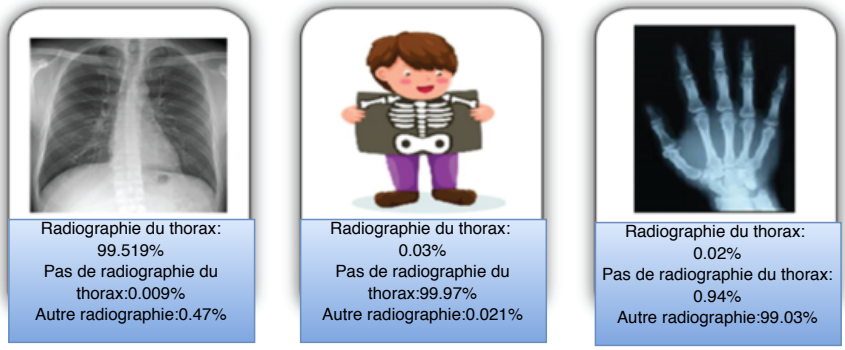

FIGURE 3.27 – Test d'image pour la classification CNN1 des trois images différentes.

La figure 3.28 montre la classification de CNN2 pour la première image. Le résultat est de 67% pour la classe normale.

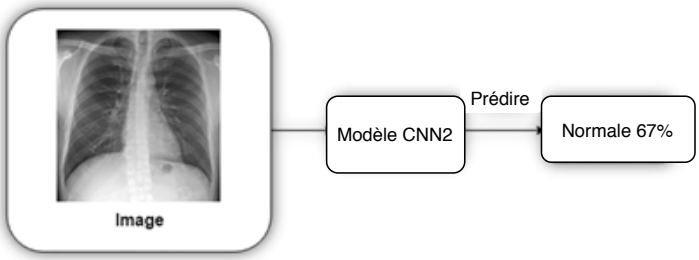

FIGURE 3.28 – Test de la première d'image pour la classification CNN2.

La figure 3.29 illustre la classification de CNN3 d'une image radiographique anormale et le résultat est une pneumonie à 62%.

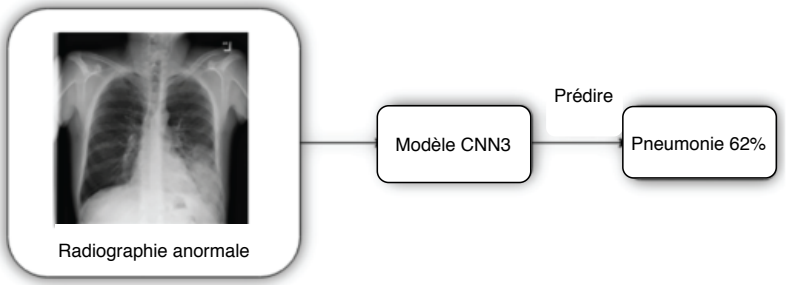

FIGURE 3.29 – Utilisation du modèle CNN3 pour la prédiction de la pneumonie

TABLEAU 3.8 – La précision des types des maladies obtenus

Anormalité	Précision
Atélectasie	0.758
cardiomégalie	0.786
Effusion	0.65
Infiltration	0.8
Masse	0.7
Nodule	0.756
Pneumonie	0.62
Pneumothorax	0.795
Épaississement pleural	0.75
Consolidation	0.645
Emphysème	0.72
Hernie	0.62
Fibrose	0.64
Œdèmes	0.69

3.3.4.1 Prédiction d'image radiographique d'un utilisateur de Twitter

Un utilisateur de Twitter publie une image radiographique et le modèle proposé peut le classer dans les commentaires des utilisateurs. La figure 3.30 montre le commentaire du modèle proposé pour anormal (76%) et normal (24%).

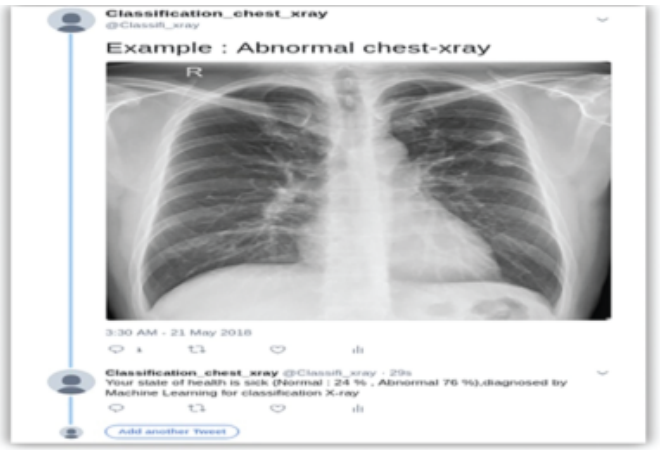

FIGURE 3.30 – Commentaires basés sur la classification d'image du modèle CNN2.

3.3.5 Etude comparative

Le modèle proposé a été comparé aux modèles proposés par Wang *et al.* (2017b), AlexNet, GoogLeNet, VGGNet-16, et ResNet-50. Le tableau 3.9 résume les résultats obtenus.

TABLEAU 3.9 – Comparaison avec d'autres résultats

Anormalité	Wang et al.	AlexNet	GoogLeNet	VGGNet-16	ResNet-50	Précision
Atélectasie	0.707	0.6458	0.6307	0.6281	0.7069	0.758
Cardiomégalie	0.8141	0.6925	0.7056	0.7084	0.8141	0.786
Effusion	0.7362	0.6642	0.6876	0.6502	0.7362	0.65
Infiltration	0.6128	0.6041	0.6088	0.5896	0.6128	0.8
Masse	0.5609	0.5644	0.5363	0.5103	0.5609	0.7
Nodule	0.7164	0.6487	0.5579	0.6556	0.7164	0.756
Pneumonie	0.6333	0.5493	0.599	0.51	0.6333	0.62
Pneumothorax	0.7891	0.7425	0.7824	0.7516	0.7891	0.795
Average of precision	0.6078	0.5582	0.5597	0.5469	0.6078	0.6383

La précision moyenne de huit anomalies montre la performance du modèle proposé avec un taux de 0,6383 par rapport aux autres modèles .Wang *et al.* (2017b), AlexNet, GoogLeNet, VGGNet-16 et ResNet-50. En outre, l'intégration entre l'apprentissage profond, les Big Data et les réseaux sociaux dans le système de santé a constitué une plate forme idéale dans ce travail, car une nouvelle approche d'utilisation de la prédiction de l'apprentissage automatique a été introduite. Jusqu'à présent, les chercheurs sont uniquement concentrés sur les modèles d'apprentissage profond, leurs caractéristiques, leurs cibles et leur précision. Dans cette étude, le modèle d'apprentissage profond a été intégré pour la prédiction des données des réseaux sociaux. Cette intégration peut faciliter l'utilisation du modèle d'apprentissage profond pour les utilisateurs du système des réseaux sociaux. Les avantages de l'intégration de l'apprentissage approfondi dans le diagnostic du système de santé sont la prédiction des diagnostics et des maladies, ainsi que le partage des connaissances d'apprentissage approfondi entre les utilisateurs des réseaux sociaux (patients, médecins, infirmières). Le modèle d'apprentissage profond de cette architecture peut interagir avec les réseaux sociaux en fournissant des commentaires sur les messages des utilisateurs concernant les maladies prévues, ce qui était un nouveau paradigme dans cette proposition. En outre, un grand système de stockage et de traitement des données a été proposé. Le grand système de données stocke des données sur les soins de santé caractérisées par leur volume, leur variété et leur vitesse. Plusieurs modèles d'apprentissage approfondi permettent d'entraîner ses données dans un système central. Le modèle d'apprentissage profond proposé forme trois ensembles de données à partir de nœuds distribués. Ainsi, L'approche de formation distribuée est la plus efficace dans la pratique.

3.3.6 Conclusion sur la contribution N° 2

Cette contribution présente un modèle d'apprentissage approfondi basé sur le Big Data pour le système de santé afin de prédire les données des réseaux sociaux. L'architecture proposée intègre trois couches : couche d'apprentissage profond, couche Big Data ,couche de réseaux sociaux. La couche des grandes données couvre toutes les questions liées aux Big Data, tandis que la couche de l'apprentissage profond est responsable de touts les algorithmes d'apprentissage. La couche des réseaux sociaux est utilisée pour les utilisateurs de réseaux sociaux. Cette contribution décrit également comment utiliser un algorithme d'apprentissage

approfondi pour Big Data sur les réseaux sociaux. Trois algorithmes CNN ont été introduits pour valider la fonctionnalité de l'architecture proposée. Le modèle CNN1 a été utilisé pour prédire le type d'image, tandis que le modèle CNN2 a été appliqué pour déterminer si l'image radiographique du thorax normale ou anormale. Enfin, le modèle CNN3 a été utilisé pour prédire le type de maladies. Les utilisateurs des réseaux sociaux peuvent interagir avec le système proposé pour identifier la classification des images radiographiques. Pour les travaux futurs de cette contribution, d'autres questions liées au système de soins de santé seront examinées pour illustrer les avantages de l'architecture proposée.

3.4 Conclusion

Dans ce chapitre nous avons présenté notre critique sur les spécifications d'architecture Big Data. Nous avons proposé deux contributions. La première est celle de la relation entre la spécification de l'architecture et les Machines Learning. Nous avons démontré que la conception de l'architecture Big Data dépend de Machine Learning. Pour valider cette hypothèse, nous avons défini quatre mesures, et nous avons proposé une Machine Learning distribuée dans les villes intelligentes dans le but de minimisation de coût de traitement et maximisation de taux de prédiction. La deuxième contribution, introduire un processus d'intégration dans les architectures Big data. Cette intégration est justifiée par l'existence des différentes sources de données tell que, les réseaux sociaux et les hôpitaux. Nous avons proposé une architecture qui intègre les réseaux sociaux, les images radiographie de différente source des hôpitaux. Cette architecture nous permettons de la reconnaissance de types d'image et le type de maladie.

Conclusion générale

Dans ce travail, nous avons étudié la relation entre le Big Data, le processus de Data Mining et notamment les Machines Learning. Nous adoptons une architecture multi couches. Cette architecture est la plus utilisée dans l'architecture des systèmes Big data. Une architecture multi couches est une conception connue en dehors de Big data tell que dans les réseaux et les architectures orientent service. Les avantages de cette architecture sont :

— Une conception claire et facile à maîtriser.

— Le traitement de chacun des problèmes dans sa couche. Par exemple, dans le cas de stockage, on ne traite que les problèmes de couche de stockage.

— Diminuer la complexité du système.

— Permettant de faciliter les tâches de développement.

Plusieurs travaux ont été présentés dans le domaine des architectures Big Data, a savoir architecture générale, référentielle, oriente application. Mais ces architectures n'ont pas étudiés la relation entre l'architecture et les Machines Learning en terme de temps d'exécution et le taux d'apprentissage. Ces architectures font un pas vers la conception des architectures Big Data. Pour ces raisons, nous avons proposé une architecture basée sur l'apprentissage profond distribué et faisons une étude empirique dans les villes intelligentes qui permet d'évaluer l'impact d'une architecture distribuée sur le temps d'exécution et le taux d'apprentissage. Les résultats obtenus sont résumés comme suit :

— Le choix de l'architecture dépend de Machine Learning.

— Le choix de l'architecture dépend de temps d'exécution.

— L'architecture distribuée permet de diminuer le coût de traitement.

— L'architecture distribuée permet de maximiser la prédiction de Machine Learning.

— L'architecture distribuée est une architecture idéale pour les villes intelligentes.

De plus, nous avons proposé une architecture qui intègre les réseaux sociaux, Big data dans les hôpitaux. Cette intégration permettant de construire un framework conceptuel pour le partage de Big Data et la construction des modèles pour la reconnaissance des images radiographiques dans le domaine de la santé. Les avantages de cette approche sont :

— Intégration des Big Data exploitable et utilisable.

— Partage de la connaissance construite par une machine Learning.

— Construire un système qui fournira des connaissances pour différents utilisateurs.

Nous avons introduit le système multi agents dans les systèmes Big Data. Le paradigme Mapreduce est modélisé par deux agents. Les agents qui exécutent les tâches Map et un agent qui exécute la tâche reduce. Les avantages de cette modélisation sont :

— Assure l'autonomie entre les tâches du système.

— Le système est tolérance aux pannes.

— Faciliter le traitement dans les systèmes Big Data.

— Introduire l'approche de coopération dans les systèmes Big Data.

— Utiliser le protocole d'interaction FIPA ACL.

Dans le but de proposer une cadre conceptuelle pour l'intégration des déférentes sources de données existantes, nous avons proposé une architecture pour l'intégration des réseaux sociaux, apprentissage profond et Big Data pour la reconnaissance des images radiographiques. Afin de clarifier les apports d'intégration des technologies des internet des objets, Big Data et l'apprentissage profond, nous avons proposé une architecture Big Data pour l'énergie renouvelable.

Enfin, ce travail est un premier pas, qui nécessite des futurs travaux dans deux axes. Le premier axe et celle de cadre théorique par exemple la relation entre l'architecture et une Machine Learning spécifiées. Le deuxième axe c'est l'axe de développement et d'application, dont la question c'est quels sont les outils qui permettent le développement facile de l'architecture proposée.

Références

Abadi, Martín, Barham, Paul, Chen, Jianmin, Chen, Zhifeng, Davis, Andy, Dean, Jeffrey, Devin, Matthieu, Ghemawat, Sanjay, Irving, Geoffrey, Isard, Michael, *et al.* 2016. Tensorflow : A system for large-scale machine learning. *Pages 265–283 of : 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*. 76

Abboud, Yacine. 2018. *Fouille de motifs : entre accessibilité et robustesse*. Ph.D. thesis, Université de Lorraine. 6

Abdi, Mohammad Javad, & Giveki, Davar. 2013. Automatic detection of erythemato-squamous diseases using pso–svm based on association rules. *Engineering applications of artificial intelligence*, **26**(1), 603–608. 71

Abu-Mostafa, Yaser S, Magdon-Ismail, Malik, & Lin, Hsuan-Tien. 2012. *Learning from data*. Vol. 4. AMLBook New York, NY, USA :. vi, 9

Achrekar, Harshavardhan, Gandhe, Avinash, Lazarus, Ross, Yu, Ssu-Hsin, & Liu, Benyuan. 2011. Predicting flu trends using twitter data. *Pages 702–707 of : 2011 ieee conference on computer communications workshops (infocom wkshps)*. IEEE. 70

Agarwal, Sonali, *et al.* 2011. Weighted support vector regression approach for remote healthcare monitoring. *Pages 969–974 of : 2011 international conference on recent trends in information technology (icrtit)*. IEEE. 71

Aggarwal, Charu C. 2015. *Data mining : the textbook*. Springer. 6

Akil, Bilal, Zhou, Ying, & Röhm, Uwe. 2017. On the usability of hadoop mapreduce, apache spark & apache flink for data science. *Pages 303–310 of : 2017 ieee international conference on big data (big data)*. IEEE. 31

Albino, Vito, Berardi, Umberto, & Dangelico, Rosa Maria. 2015. Smart cities : Definitions, dimensions, performance, and initiatives. *Journal of urban technology*, **22**(1), 3–21. 55

Alliance, Open Data Center. 2012. Big data consumer guide. *Open data center alliance*. 37

Angelov, Samuil, Grefen, Paul, & Greefhorst, Danny. 2012. A framework for analysis and design of software reference architectures. *Information and software technology*, **54**(4), 417–431. 34

Anthony, M, & Arjuna, C. 2011. *Introduction to hpcc (high-performancecomputing cluster)*. vi, 2, 35

Anveshrithaa, S, & Lavanya, K. 2020. Real-time vehicle traffic analysis using long short term memory networks in apache spark. *Pages 1–5 of : 2020 international conference on emerging trends in information technology and engineering (ic-etite)*. IEEE. 51

Assunção, Marcos D, Calheiros, Rodrigo N, Bianchi, Silvia, Netto, Marco AS, & Buyya, Rajkumar. 2015. Big data computing and clouds : Trends and future directions. *Journal of parallel and distributed computing*, **79**, 3–15. 1, 25

Aufaure, Marie-Aude, Chiky, Raja, Curé, Olivier, Khrouf, Houda, & Kepeklian, Gabriel. 2016. From business intelligence to semantic data stream management. *Future generation computer systems*, **63**, 100–107. 51

Avci, Engin. 2009. A new intelligent diagnosis system for the heart valve diseases by using genetic-svm classifier. *Expert systems with applications*, **36**(7), 10618–10626. 71

Basanta-Val, Pablo, Fernández-García, Norberto, & Sánchez-Fernández, Luis. 2020. Predictable remote invocations for distributed stream processing. *Future generation computer systems*, **107**, 716–729. 51

Bechini, Alessio, Marcelloni, Francesco, & Segatori, Armando. 2016. A mapreduce solution for associative classification of big data. *Information sciences*, **332**, 33–55. 50

Belciug, Smaranda. 2009. Patients length of stay grouping using the hierarchical clustering algorithm. *Annals of the university of craiova-mathematics and computer science series*, **36**(2), 79–84. 71

Bellazzi, Riccardo, & Zupan, Blaz. 2008. Predictive data mining in clinical medicine : current issues and guidelines. *International journal of medical informatics*, **77**(2), 81–97. 71

Bengio, Yoshua, Courville, Aaron, & Vincent, Pascal. 2013. Representation learning : A review and new perspectives. *Ieee transactions on pattern analysis and machine intelligence*, **35**(8), 1798–1828. 58

Bilal, Muhammad, Oyedele, Lukumon O., Akinadé, Olúgbénga O., Ajayi, Saheed, Alaka, Hafiz, Owolabi, Hakeem, Qadir, Junaid, Pasha, Maruf, & Bello, Sururah A. 2016. Big data architecture for construction waste analytics (cwa) : A conceptual framework. vi, 2, 43

Bordes, Antoine, Glorot, Xavier, Weston, Jason, & Bengio, Yoshua. 2012. Joint learning of words and meaning representations for open-text semantic parsing. *Pages 127–135 of : Artificial intelligence and statistics*. 58

Borthakur, Dhruba, *et al.* 2008. Hdfs architecture guide. *Hadoop apache project*, **53**(1-13), 2. vi, 28, 29

Breiman, Leo. 2001. Random forests. *Machine learning*, **45**(1), 5–32. 15

Breiman, Leo, Friedman, Jerome, Stone, Charles J, & Olshen, Richard A. 1984. *Classification and regression trees*. CRC press. 13

Campesato, Oswald. 2020. *Artificial intelligence, machine learning, and deep learning*. Stylus Publishing, LLC. 20

Canlas, RD. 2009. Data mining in healthcare : Current applications and issues. *School of information systems & management, carnegie mellon university, australia*. 71

Castiglione, Aniello, Colace, Francesco, Moscato, Vincenzo, & Palmieri, Francesco. 2018. Chis : A big data infrastructure to manage digital cultural items. *Future generation computer systems*, **86**, 1134–1145. 50

Celebi, M Emre, Aslandogan, Y Alp, & Bergstresser, Paul R. 2005. Mining biomedical images with density-based clustering. *Pages 163–168 of : International conference on information technology : coding and computing (itcc'05)-volume ii*, vol. 1. IEEE. 71

Ceri, Stefano, Fraternali, Piero, Bongio, Aldo, Brambilla, Marco, Comai, Sara, & Matera, Maristella. 2003. *Morgan kaufmann series in data management systems : Designing data-intensive web applications*. Morgan Kaufmann. 7

Chang, Chun-Lang, & Chen, Chih-Hao. 2009. Applying decision tree and neural network to increase quality of dermatologic diagnosis. *Expert systems with applications*, **36**(2), 4035–4041. 71

Chang, Wo L, Mishra, Sanjay, *et al.* 2015. *Nist big data interoperability framework : Volume 5, architectures white paper survey*. Tech. rept. 37

Chen, Cheng, Zhang, Qingmei, Ma, Qin, & Yu, Bin. 2019. Lightgbm-ppi : Predicting protein-protein interactions through lightgbm with multi-information fusion. *Chemometrics and intelligent laboratory systems*, **191**, 54–64. 16

Chen, Hongmei, Li, Tianrui, Luo, Chuan, Horng, Shi-Jinn, & Wang, Guoyin. 2014. A rough set-based method for updating decision rules on attribute values' coarsening and refining. *Ieee transactions on knowledge and data engineering*, **26**(12), 2886–2899. 58

Chen, Jin, Wang, Cheng, & Wang, Runsheng. 2009. Using stacked generalization to combine svms in magnitude and shape feature spaces for classification of hyperspectral data. *Ieee transactions on geoscience and remote sensing*, **47**(7), 2193–2205. 58

Chen, Tung-Shou, Tsai, Tzu-Hsin, Chen, Yi-Tzu, Lin, Chin-Chiang, Chen, Rong-Chang, Li, Shuan-Yow, & Chen, Hsin-Yi. 2005. A combined k-means and hierarchical clustering method for improving the clustering efficiency of microarray. *Pages 405–408 of : 2005 international symposium on intelligent signal processing and communication systems*. IEEE. 71

Chen, Xue-Wen, & Lin, Xiaotong. 2014. Big data deep learning : challenges and perspectives. *Ieee access*, **2**, 514–525. 58

Chien, Chieh, & Pottie, Gregory J. 2012. A universal hybrid decision tree classifier design for human activity classification. *Pages 1065–1068 of : 2012 annual international conference of the ieee engineering in medicine and biology society*. IEEE. 71

Chipman, Hugh, & Tibshirani, Robert. 2006. Hybrid hierarchical clustering with applications to microarray data. *Biostatistics*, **7**(2), 286–301. 71

Cho, Kyunghyun, Van Merriënboer, Bart, Gulcehre, Caglar, Bahdanau, Dzmitry, Bougares, Fethi, Schwenk, Holger, & Bengio, Yoshua. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arxiv preprint arxiv :1406.1078*. vi, 20

Collobert, Ronan. 2011. Deep learning for efficient discriminative parsing. *Pages 224–232 of : Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 12

Crawford, Melba M, Tuia, Devis, & Yang, Hsiuhan Lexie. 2013. Active learning : Any value for classification of remotely sensed data ? *Proceedings of the ieee*, **101**(3), 593–608. 58

Dangare, Chaitrali S, & Apte, Sulabha S. 2012. Improved study of heart disease prediction system using data mining classification techniques. *International journal of computer applications*, **47**(10), 44–48. 71

Das, Resul, Turkoglu, Ibrahim, & Sengur, Abdulkadir. 2009. Effective diagnosis of heart disease through neural networks ensembles. *Expert systems with applications*, **36**(4), 7675–7680. 71

De Maio, Carmen, Fenza, Giuseppe, Loia, Vincenzo, & Orciuoli, Francesco. 2017. Distributed online temporal fuzzy concept analysis for stream processing in smart cities. *Journal of parallel and distributed computing*, **110**, 31–41. 49

Dean, Jeffrey, & Ghemawat, Sanjay. 2010. Mapreduce : a flexible data processing tool. *Communications of the acm*, **53**(1), 72–77. 28

Deisenroth, Marc Peter, Faisal, A Aldo, & Ong, Cheng Soon. 2020. *Mathematics for machine learning*. Cambridge University Press. 9, 13

Demchenko, Yuri, De Laat, Cees, & Membrey, Peter. 2014. Defining architecture components of the big data ecosystem. *Pages 104–112 of : 2014 international conference on collaboration technologies and systems (cts)*. IEEE. vi, 2, 35, 36

Dietterich, Thomas G. 1998. An experimental comparison of three methods for constructing ensembles of decision trees : Bagging, boosting and randomization. *Machine learning*, **32**, 1–22. 15

Dietterich, Thomas G. 2000. Ensemble methods in machine learning. *Pages 1–15 of : International workshop on multiple classifier systems*. Springer. 15

Ding, Guoru, Wu, Qihui, Yao, Yu-Dong, Wang, Jinlong, & Chen, Yingying. 2013. Kernel-based learning for statistical signal processing in cognitive radio networks : Theoretical foundations, example applications, and future directions. *Ieee signal processing magazine*, **30**(4), 126–136. 58

Duchi, John, Hazan, Elad, & Singer, Yoram. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, **12**(Jul), 2121–2159. 22

Dwivedi, Kartik, Biswaranjan, Kumar, & Sethi, Amit. 2014. Drowsy driver detection using representation learning. *Pages 995–999 of : 2014 ieee international advance computing conference (iacc)*. IEEE. 58

Efron, Bradley, & Tibshirani, Robert. 1986. Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statistical science*, 54–75. 14

El Kassabi, Hadeel T, Serhani, Mohamed Adel, Dssouli, Rachida, & Benatallah, Boualem. 2018. A multi-dimensional trust model for processing big data over competing clouds. *Ieee access*, **6**, 39989–40007. 1, 25

Er, Orhan, Yumusak, Nejat, & Temurtas, Feyzullah. 2010. Chest diseases diagnosis using artificial neural networks. *Expert systems with applications*, **37**(12), 7648–7655. 71

Escudero, Javier, Zajicek, John P, & Ifeachor, Emmanuel. 2011. Early detection and characterization of alzheimer's disease in clinical scenarios using bioprofile concepts and k-means. *Pages 6470–6473 of : 2011 annual international conference of the ieee engineering in medicine and biology society*. IEEE. 71

Fakoor, Rasool, Ladhak, Faisal, Nazi, Azade, & Huber, Manfred. 2013. Using deep learning to enhance cancer diagnosis and classification. *In : Proceedings of the international conference on machine learning*, vol. 28. ACM New York, USA. 71

Fayyad, Usama, Piatetsky-Shapiro, Gregory, & Smyth, Padhraic. 1996. From data mining to knowledge discovery in databases. *Ai magazine*, **17**(3), 37–37. vi, 6, 7

Fei, Sheng-wei. 2010. Diagnostic study on arrhythmia cordis based on particle swarm optimization-based support vector machine. *Expert systems with applications*, **37**(10), 6748–6752. 71

Garg, Nishant. 2013. *Apache kafka*. Packt Publishing Ltd. 32

Garlan, David, & Perry, Dewayne E. 1995. Introduction to the special issue on software architecture. *Ieee trans. software eng.*, **21**(4), 269–274. 33

Gaur, Aditya, Scotney, Bryan, Parr, Gerard, & McClean, Sally. 2015. Smart city architecture and its applications based on iot. *Procedia computer science*, **52**, 1089–1094. vi, 59

Gehring, Jonas, Miao, Yajie, Metze, Florian, & Waibel, Alex. 2013. Extracting deep bottleneck features using stacked auto-encoders. *Pages 3377–3381 of : 2013 ieee international conference on acoustics, speech and signal processing*. IEEE. 19

Gennings, Chris, Ellis, Rhonda, & Ritter, Joseph K. 2012. Linking empirical estimates of body burden of environmental chemicals and wellness using nhanes data. *Environment international*, **39**(1), 56–65. 71

Goldston, David. 2008. Big data : Data wrangling. *Nature*, **455**(7209), 15. 1

Golmohammadi, Meysam, Harati Nejad Torbati, Amir Hossein, Lopez de Diego, Silvia, Obeid, Iyad, & Picone, Joseph. 2019. Automatic analysis of eegs using big data and hybrid deep learning architectures. *Frontiers in human neuroscience*, **13**, 76. 50

Goodfellow, Ian. 2016. Nips 2016 tutorial : Generative adversarial networks. *arxiv preprint arxiv :1701.00160*. vi, 21

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, & Bengio, Yoshua. 2014. Generative adversarial nets. *Pages 2672–2680 of : Advances in neural information processing systems*. 20

Goodfellow, Ian, Bengio, Yoshua, Courville, Aaron, & Bengio, Yoshua. 2016. *Deep learning*. Vol. 1. MIT press Cambridge. vi, 11, 19, 21

Graves, Alex. 2013. Generating sequences with recurrent neural networks. *arxiv preprint arxiv :1308.0850*. 19

Grolinger, Katarina, Mezghani, Emna, Capretz, Miriam AM, & Exposito, Ernesto. 2015. Collaborative knowledge as a service applied to the disaster management domain. *International journal of cloud computing*, **4**(1), 5. 45

Grover, Mark, Malaska, Ted, Seidman, Jonathan, & Shapira, Gwen. 2015. *Hadoop application architectures : Designing real-world big data applications*. " O'Reilly Media, Inc.". 76

Gubbi, Jayavardhana, Buyya, Rajkumar, Marusic, Slaven, & Palaniswami, Marimuthu. 2013. Internet of things (iot) : A vision, architectural elements, and future directions. *Future generation computer systems*, **29**(7), 1645–1660. 55

Gunasundari, Selvaraj, & Baskar, S. 2009. Application of artificial neural network in identification of lung diseases. *Pages 1441–1444 of : 2009 world congress on nature & biologically inspired computing (nabic)*. IEEE. 71

Gupta, Shelly, Kumar, Dharminder, & Sharma, Anand. 2011. Data mining classification techniques applied for breast cancer diagnosis and prognosis. *Indian journal of computer science and engineering (ijcse)*, **2**(2), 188–195. 71

Han, Jiawei, Pei, Jian, Mortazavi-Asl, Behzad, Chen, Qiming, Dayal, Umeshwar, & Hsu, Mei-Chun. 2000. Freespan : frequent pattern-projected sequential pattern mining. *Pages 355–359 of : Proceedings of the sixth acm sigkdd international conference on knowledge discovery and data mining*. 6

Han, Jiawei, Pei, Jian, & Kamber, Micheline. 2011. *Data mining : concepts and techniques*. Elsevier. 6, 7, 13

Haque, Md Muksitul, Holder, Lawrence B, Skinner, Michael K, & Cook, Diane J. Generalized query-based active learning to identify differentially methylated regions in dna. *Ieee/acm transactions on computational biology and bioinformatics (tcbb)*, **10**(3). 58

Hewitt, Eben. 2010. *Cassandra : the definitive guide*. " O'Reilly Media, Inc.". 32

Higashino, Wilson A, Capretz, Miriam AM, & Bittencourt, Luiz F. 2016. Cepsim : Modelling and simulation of complex event processing systems in cloud environments. *Future generation computer systems*, **65**, 122–139. 50

Hirose, Hideo, & Wang, Liangliang. 2012. Prediction of infectious disease spread using twitter : A case of influenza. *Pages 100–105 of : 2012 fifth international symposium on parallel architectures, algorithms and programming*. IEEE. 70

Hochreiter, Sepp, & Schmidhuber, Jürgen. 1997. Long short-term memory. *Neural computation*, **9**(8), 1735–1780. 57

Hoffman, Steve. 2013. *Apache flume : distributed log collection for hadoop*. Packt Publishing Ltd. 30

Hosseinkhah, Fatemeh, Ashktorab, Hassan, Veen, Ranjit, *et al.* 2009. Challenges in data mining on medical databases. *Pages 1393–1404 of : Database technologies : Concepts, methodologies, tools, and applications*. IGI global. 71

Hu, Xiling. 2020. An analysis on task migration strategy of big data streaming storm computing framework for distributed processing. *International journal of information system modeling and design (ijismd)*, **11**(4), 18–35. 51

Huang, Cheng-Lung, Liao, Hung-Chang, & Chen, Mu-Chen. 2008. Prediction model building and feature selection with support vector machines in breast cancer diagnosis. *Expert systems with applications*, **34**(1), 578–587. 71

Huang, Chi-Sheng, Tsai, Meng-Feng, Huang, Po-Hsuan, Su, Li-Ding, & Lee, Kuei-Sheng. 2017. Distributed asteroid discovery system for large astronomical data. *Journal of network and computer applications*, **93**, 27–37. 49

Huang, Fei, & Yates, Alexander. 2010. Exploring representation-learning approaches to domain adaptation. *Pages 23–30 of : Proceedings of the 2010 workshop on domain adaptation for natural language processing*. Association for Computational Linguistics. 58

Huang, Fei, & Yates, Alexander. 2012. Biased representation learning for domain adaptation. *Pages 1313–1323 of : Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*. Association for Computational Linguistics. 58

Hunt, Patrick, Konar, Mahadev, Junqueira, Flavio Paiva, & Reed, Benjamin. 2010. Zookeeper : Wait-free coordination for internet-scale systems. *In : Usenix annual technical conference*, vol. 8. 32

Islam, Mohammad, Huang, Angelo K, Battisha, Mohamed, Chiang, Michelle, Srinivasan, Santhosh, Peters, Craig, Neumann, Andreas, & Abdelnur, Alejandro. 2012. Oozie : towards a scalable workflow management system for hadoop. *Pages 1–10 of : Proceedings of the 1st acm sigmod workshop on scalable workflow execution engines and technologies*. 31

Jacobs, Robert A. 1988. Increased rates of convergence through learning rate adaptation. *Neural networks*, **1**(4), 295–307. 22

Jaeger, Stefan, Candemir, Sema, Antani, Sameer, Wáng, Yì-Xiáng J, Lu, Pu-Xuan, & Thoma, George. 2014. Two public chest x-ray datasets for computer-aided screening of pulmonary diseases. *Quantitative imaging in medicine and surgery*, **4**(6), 475. 74

Jain, Ankit. 2017. *Mastering apache storm : Real-time big data streaming using kafka, hbase and redis*. Packt Publishing Ltd. vi, 32

Jen, Lih-ren, & Lee, Yuh-jye. 2000. Working group. ieee recommended practice for architectural description of software-intensive systems. *In : Ieee architecture*. Citeseer. 33

Ji, Yanqing, Ying, Hao, Tran, John, Dews, Peter, Mansour, Ayman, & Massanari, R Michael. 2011. Mining infrequent causal associations in electronic health databases. *Pages 421–428 of : 2011 ieee 11th international conference on data mining workshops*. IEEE. 71

Jones, Nicola. 2014. Computer science : The learning machines. *Nature news*, **505**(7482), 146. 58

Karau, Holden, Konwinski, Andy, Wendell, Patrick, & Zaharia, Matei. 2015. *Learning spark : lightning-fast big data analysis*. " O'Reilly Media, Inc.". vi, 30

Karunaratne, Pasan, Karunasekera, Shanika, & Harwood, Aaron. 2017. Distributed stream clustering using micro-clusters on apache storm. *Journal of parallel and distributed computing*, **108**, 74–84. 51

Ke, Guolin, Meng, Qi, Finley, Thomas, Wang, Taifeng, Chen, Wei, Ma, Weidong, Ye, Qiwei, & Liu, Tie-Yan. 2017. Lightgbm : A highly efficient gradient boosting decision tree. *Pages 3146–3154 of : Advances in neural information processing systems*. 16

Keyno, HR Sadeghi, Ghaderi, F, Azade, A, & Razmi, J. 2009. Forecasting electricity consumption by clustering data in order to decline the periodic variable's affects and simplification the pattern. *Energy conversion and management*, **50**(3), 829–836. 57

Khan, Muhammad Umer, Choi, Jong Pill, Shin, Hyunjung, & Kim, Minkoo. 2008. Predicting breast cancer survivability using fuzzy decision trees for personalized healthcare. *Pages 5148–5151 of : 2008 30th annual international conference of the ieee engineering in medicine and biology society*. IEEE. 71

Kim, Gang-Hoon, Trimi, Silvana, & Chung, Ji-Hyong. 2014. Big-data applications in the government sector. *Communications of the acm*, **57**(3), 78–85. 25

Kinsta. 2020. *Wild and interesting facebook statistics and facts (2020)*. Last accessed 06 June 2020. 25

Koh, Hian Chye, Tan, Gerald, *et al.* 2011. Data mining applications in healthcare. *Journal of healthcare information management*, **19**(2), 65. 71

Kononenko, Igor, & Kukar, Matjaz. 2007. *Machine learning and data mining*. Horwood Publishing. 6

Kousiouris, George, Akbar, Adnan, Sancho, Juan, Ta-Shma, Paula, Psychas, Alexandros, Kyriazis, Dimosthenis, & Varvarigou, Theodora. 2018. An integrated information lifecycle management framework for exploiting social network data to identify dynamic large crowd concentration events in smart cities applications. *Future generation computer systems*, **78**, 516–530. 52

Kovalchuk, Sergey V, Krotov, Evgeniy, Smirnov, Pavel A, Nasonov, Denis A, & Yakovlev, Alexey N. 2018. Distributed data-driven platform for urgent decision making in cardiological ambulance control. *Future generation computer systems*, **79**, 144–154. 52

Kranjc, Janez, Orač, Roman, Podpečan, Vid, Lavrač, Nada, & Robnik-Šikonja, Marko. 2017. Clowdflows : Online workflows for distributed big data mining. *Future generation computer systems*, **68**, 38–58. 50

Krishnan, Krish. 2013. *Data warehousing in the age of big data*. Newnes. 1, 33

Kumari, Milan, & Godara, Sunila. 2011. Comparative study of data mining classification methods in cardiovascular disease prediction 1. 71

Lam, Chuck. 2010. *Hadoop in action*. Manning Publications Co. 28

Landset, Sara, Khoshgoftaar, Taghi M, Richter, Aaron N, & Hasanin, Tawfiq. 2015. A survey of open source tools for machine learning with big data in the hadoop ecosystem. *Journal of big data*, **2**(1), 24. vi, 27

Larochelle, Hugo, Mandel, Michael, Pascanu, Razvan, & Bengio, Yoshua. 2012. Learning algorithms for the classification restricted boltzmann machine. *Journal of machine learning research*, **13**(Mar), 643–669. 20

LeCun, Yann, Boser, Bernhard E, Denker, John S, Henderson, Donnie, Howard, Richard E, Hubbard, Wayne E, & Jackel, Lawrence D. 1990. Handwritten digit recognition with a back-propagation network. *Pages 396–404 of : Advances in neural information processing systems*. 74

LeCun, Yann, Bengio, Yoshua, & Hinton, Geoffrey. 2015. Deep learning. *nature*, **521**(7553), 436. 56

Levin, BO. 2013. Big data ecosystem reference architecture. *Microsoft corporation*. vi, 2, 36, 37

Leyva, Enrique, González, Antonio, & Perez, Raul. 2015. A set of complexity measures designed for applying meta-learning to instance selection. *Ieee transactions on knowledge and data engineering*, **27**(2), 354–367. 58

Li, Shou-Shan, Huang, Chu-Ren, & Zong, Cheng-Qing. 2011. Multi-domain sentiment classification with classifier combination. *Journal of computer science and technology*, **26**(1), 25–33. 58

Lipton, Zachary C, Kale, David C, Elkan, Charles, & Wetzel, Randall. 2015. Learning to diagnose with lstm recurrent neural networks. *arxiv preprint arxiv :1511.03677*. 71

Liu, Fang, Tong, Jin, Mao, Jian, Bohn, Robert, Messina, John, Badger, Lee, & Leaf, Dawn. 2011. Nist cloud computing reference architecture. *Nist special publication*, **500**(2011), 292. 45

Liu, Kevin FR, & Lu, Che-Fan. 2009. Bbn-based decision support for health risk analysis. *Pages 696–702 of : 2009 fifth international joint conference on inc, ims and idc*. IEEE. 71

Ma, Xiaojun, Sha, Jinglan, Wang, Dehua, Yu, Yuanbo, Yang, Qian, & Niu, Xueqi. 2018. Study on a prediction of p2p network loan default based on the machine learning lightgbm and xgboost algorithms according to different high dimensional data cleaning. *Electronic commerce research and applications*, **31**, 24–39. 16

Mackey, Grant, Sehrish, Saba, & Wang, Jun. 2009. Improving metadata management for small files in hdfs. *Pages 1–4 of : 2009 ieee international conference on cluster computing and workshops*. IEEE. 28

MacQueen, James, *et al.* 1967. Some methods for classification and analysis of multivariate observations. *Pages 281–297 of : Proceedings of the fifth berkeley symposium on mathematical statistics and probability*, vol. 1. Oakland, CA, USA. 18

Martha, Venkata Swamy, Zhao, Weizhong, & Xu, Xiaowei. 2013. h-mapreduce : a framework for workload balancing in mapreduce. *Pages 637–644 of : 2013 ieee 27th international conference on advanced information networking and applications (aina)*. IEEE. 28

Martínez-Álvarez, Francisco, Troncoso, Alicia, Riquelme, José C, & Aguilar-Ruiz, Jesús S. 2008. Lbf : A labeled-based forecasting algorithm and its application to electricity price time series. *Pages 453–461 of : 2008 eighth ieee international conference on data mining*. IEEE. 57

Maté, Alejandro, Peral, Jesús, Ferrández, Antonio, Gil, David, & Trujillo, Juan. 2016. A hybrid integrated architecture for energy consumption prediction. *Future generation computer systems*, **63**, 131–147. 50

Mavridis, Ilias, & Karatza, Helen. 2017. Performance evaluation of cloud-based log file analysis with apache hadoop and apache spark. *Journal of systems and software*, **125**, 133–151. 52

Mezghani, Emna, Exposito, Ernesto, Drira, Khalil, Da Silveira, Marcos, & Pruski, Cédric. 2015. A semantic big data platform for integrating heterogeneous wearable data in healthcare. *Journal of medical systems*, **39**(12), 185. vi, 2, 45, 46

Miotto, Riccardo, Li, Li, Kidd, Brian A, & Dudley, Joel T. 2016. Deep patient : an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, **6**(1), 1–10. 71

Mitchell, Tom M, *et al.* 1997. *Machine learning*. 9

Mohapatra, Suvendu Kumar, Sahoo, Prasan Kumar, & Wu, Shih-Lin. 2016. Big data analytic architecture for intruder detection in heterogeneous wireless sensor networks. *Journal of network and computer applications*, **66**, 236–249. 50

Montavon, Gregoire, Braun, Mikio L, Krueger, Tammo, & Muller, Klaus-Robert. 2013. Analyzing local structure in kernel-based learning : Explanation, complexity, and reliability assessment. *Ieee signal processing magazine*, **30**(4), 62–74. 58

Moon, Sung Seek, Kang, Suk-Young, Jitpitaklert, Weerawat, & Kim, Seoung Bum. 2012. Decision tree models for characterizing smoking patterns of older adults. *Expert systems with applications*, **39**(1), 445–451. 71

Müller, Klaus-Robert, Mika, Sebastian, Rätsch, Gunnar, Tsuda, Koji, & Schölkopf, Bernhard. 2001. An introduction to kernel-based learning algorithms. *Ieee transactions on neural networks*, **12**(2). 58

Munro, Alistair, & Clayton, Gary. 2019. Drone swarms, communications performance and big data. *Pages 1–5 of : 2019 ieee 90th vehicular technology conference (vtc2019-fall)*. IEEE. 52

Naoui, Med Anouar, Lejdel, Brahim, & Ayad, Mouloud. 2020a. Integrating iot devices and deep learning for renewable energy in big data system. *Scientific bulletin*. 2, 3, 54, 55, 77

Naoui, Med Anouar, Lejdel, Brahim, & Ayad, Mouloud. 2020b. Using k-means algorithm for regression curve in big data system for business environment. *Revista cubana de ciencias informáticas*, **14**(2), 34–48. 2, 54

Naoui, Mohammed Anouar, Lejdel, Brahim, Ayad, Mouloud, Belkeiri, Riad, *et al.* 2020c. Integrating deep learning, social networks, and big data for healthcare system. *Bio-algorithms and med-systems*, **16**(1). 2, 54

Naoui, Mohammed Anouar, Lejdel, Brahim, Ayad, Mouloud, Amamra, Abdelfattah, *et al.* 2020d. Using a distributed deep learning algorithm for analyzing big data in smart cities. *Smart and sustainable built environment*. vii, 2, 54, 55, 59, 61, 62, 64, 65, 72, 74

Osman, Ahmed M Shahat. 2019. A novel big data analytics framework for smart cities. *Future generation computer systems*, **91**, 620–633. vi, 2, 40, 42

Otoo-Arthur, David, & van Zyl, Terence L. 2020. A scalable heterogeneous big data framework for e-learning systems. *Pages 1–15 of : 2020 international conference on artificial intelligence, big data, computing and data communication systems (icabcd)*. IEEE. 51

Owen, Sean. 2012. Mahout in action. 31

Pääkkönen, Pekka, & Pakkala, Daniel. 2015. Reference architecture and classification of technologies, products and services for big data systems. *Big data research*, **2**(4), 166–186. vi, 2, 39, 40

Pan, Sinno Jialin, & Yang, Qiang. 2010. A survey on transfer learning. *Ieee transactions on knowledge and data engineering*, **22**(10), 1345–1359. 58

Perera, S. 2013. Thilina gunarathne hadoop mapreduce cookbook. *Birmingham : Packt publishing ltd*, **300**. 28

Phan, NhatHai, Dou, Dejing, Piniewski, Brigitte, & Kil, David. 2015. Social restricted boltzmann machine : Human behavior prediction in health social networks. *Pages 424–431 of : Proceedings of the 2015 ieee/acm international conference on advances in social networks analysis and mining 2015*. 71

Piateski, Gregory, & Frawley, William. 1991. *Knowledge discovery in databases*. MIT press. 6

Polyak, Boris T. 1964. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, **4**(5), 1–17. 22

Qiu, Junfei, Wu, Qihui, Ding, Guoru, Xu, Yuhua, & Feng, Shuo. 2016. A survey of machine learning for big data processing. *Eurasip journal on advances in signal processing*, **2016**(1), 67. 1, 58

Quinlan, J. Ross. 1980. An introduction to knowledge-based expert systems. *Australian computer journal*, **12**(2), 56–62. 13

Quinlan, JR. 1993. C4. 5 : Programs for machine learning morgan kaufmann publishers san francisco. *Ca google sch*, 155–164. 13

Rajaraman, Anand, & Ullman, Jeffrey David. 2011. *Mining of massive datasets*. Cambridge University Press. 1

Raschka, Sebastian, & Mirjalili, Vahid. 2017. *Python machine learning*. Packt Publishing Ltd. 76

Rathore, M Mazhar, Ahmad, Awais, Paul, Anand, & Jeon, Gwanggil. 2015. Efficient graph-oriented smart transportation using internet of things generated big data. *Pages 512–519 of : 2015 11th international conference on signal-image technology & internet-based systems (sitis)*. IEEE. vi, 2, 43, 45

Reddy, Vikram, Yedavalli, Pavan, Mohanty, Shrestha, & Nakhat, Udit. 2018. *Deep air : Forecasting air pollution in beijing, china*. 57

Requeno, José Ignacio, Gascón, Iñigo, & Merseguer, José. 2018. Towards the performance analysis of apache tez applications. *Pages 147–152 of : Companion of the 2018 acm/spec international conference on performance engineering*. 31

Russom, Philip, *et al.* 2011. Big data analytics. *Tdwi best practices report, fourth quarter*, **19**(4), 1–34. 56

Sang, Go Muan, Xu, Lai, & de Vrieze, Paul. 2016. A reference architecture for big data systems. *Pages 370–375 of : 2016 10th international conference on software, knowledge, information management & applications (skima)*. IEEE. vi, 2, 39

Sarnovsky, Martin, & Vronc, Michal. 2014. Distributed boosting algorithm for classification of text documents. *Pages 217–220 of : 2014 ieee 12th international symposium on applied machine intelligence and informatics (sami)*. IEEE. 58

Schein, Rebecca, Wilson, Kumanan, & Keelan, Jennifer E. 2011. *Literature review on effectiveness of the use of social media : a report for peel public health*. [Region of Peel], Peel Public Health. 69

Schmidt, Rainer, & Möhring, Michael. 2013. Strategic alignment of cloud-based architectures for big data. *Pages 136–143 of : 2013 17th ieee international enterprise distributed object computing conference workshops*. IEEE. vi, 2, 49

Settles, Burr. 2014. Active learning literature survey. 2010. *Computer sciences technical report*, **1648**. 58

Shafer, Jeffrey, Rixner, Scott, & Cox, Alan L. 2010. The hadoop distributed filesystem : Balancing portability and performance. *Pages 122–133 of : 2010 ieee international symposium on performance analysis of systems & software (ispass)*. IEEE. 28

Shvachko, Konstantin, Kuang, Hairong, Radia, Sanjay, & Chansler, Robert. 2010. The hadoop distributed file system. *Pages 1–10 of : 2010 ieee 26th symposium on mass storage systems and technologies (msst)*. Ieee. 28

Simonyan, Karen, & Zisserman, Andrew. 2014. Very deep convolutional networks for large-scale image recognition. *arxiv preprint arxiv :1409.1556*. 19

Slavakis, Konstantinos, Theodoridis, Sergios, & Yamada, Isao. 2009. Adaptive constrained learning in reproducing kernel hilbert spaces : the robust beamforming case. *Ieee transactions on signal processing*, **57**(12), 4744–4764. 58

Slavakis, Konstantinos, Bouboulis, Pantelis, & Theodoridis, Sergios. 2012. Adaptive multiregression in reproducing kernel hilbert spaces : The multiaccess mimo channel case. *Ieee transactions on neural networks and learning systems*, **23**(2), 260–276. 58

Soni, Sunita, & Vyas, OP. 2010. Using associative classifiers for predictive analysis in health care data mining. *International journal of computer applications*, **4**(5), 33–37. 71

Statista1. 2019. *Number of internet of things (iot) connected devices worldwide in 2018, 2025 and 2030*. Last accessed 06 June 2020. 25

Stéphane, Tufféry. 2012. *Data mining et statistique décisionnelle : l'intelligence des données*. Editions Technip. 6

Sumbaly, Roshan, Kreps, Jay, & Shah, Sam. 2013. The big data ecosystem at linkedin. *Pages 1125–1134 of : Proceedings of the 2013 acm sigmod international conference on management of data*. vi, 2, 47, 48

Sun, Xiaolei, Liu, Mingxi, & Sima, Zeqian. 2018. A novel cryptocurrency price trend forecasting model based on lightgbm. *Finance research letters*. 16

Supriya, BN, Prakash, S, & Akki, CB. 2016. A survey on big data architectures and standard bodies. *Pages 391–400 of : Proceedings of the international congress on information and communication technology*. Springer. vi, 2, 37, 38

Swarna, C, & Ansari, Zahid. 2017. Apache pig-a data flow framework based on hadoop map reduce. *International journal of engineering trends and technology*, **50**(5), 271âĂŞ275. 30

Tapia, José Juan, Morett, Enrique, & Vallejo, Edgar E. 2009. A clustering genetic algorithm for genomic data mining. *Pages 249–275 of : Foundations of computational intelligence volume 4*. Springer. 71

Taylor, Ronald C. 2010. An overview of the hadoop/mapreduce/hbase framework and its current applications in bioinformatics. *Page S1 of : Bmc bioinformatics*, vol. 11. BioMed Central. 62

Thusoo, Ashish, Shao, Zheng, Anthony, Suresh, Borthakur, Dhruba, Jain, Namit, Sen Sarma, Joydeep, Murthy, Raghotham, & Liu, Hao. 2010a. Data warehousing and analytics infrastructure at facebook. *Pages 1013–1020 of : Proceedings of the 2010 acm sigmod international conference on management of data*. vi, 2, 46, 47

Thusoo, Ashish, Sarma, Joydeep Sen, Jain, Namit, Shao, Zheng, Chakka, Prasad, Zhang, Ning, Antony, Suresh, Liu, Hao, & Murthy, Raghotham. 2010b. Hive-a petabyte scale data warehouse using hadoop. *Pages 996–1005 of : 2010 ieee 26th international conference on data engineering (icde 2010)*. IEEE. 30

Ting, Kathleen, & Cecho, Jarek Jarcec. 2013. *Apache sqoop cookbook : Unlocking hadoop for your relational database*. " O'Reilly Media, Inc.". 31

Tu, Wenting, & Sun, Shiliang. 2012. Cross-domain representation-learning framework with combination of class-separate and domain-merge objectives. *Pages 18–25 of : Proceedings of the 1st international workshop on cross domain knowledge discovery in web and social network mining*. ACM. 58

Uddin, Muhammad Fahim, Gupta, Navarun, *et al.* 2014. Seven v's of big data understanding big data to extract value. *Pages 1–5 of : Proceedings of the 2014 zone 1 conference of the american society for engineering education*. IEEE. 1, 25

Um, Jung-Ho, Lee, Seungwoo, Kim, Tae-Hong, Jeong, Chang-Hoo, Song, Sa-Kwang, & Jung, Hanmin. 2016. Semantic complex event processing model for reasoning research activities. *Neurocomputing*, **209**, 39–45. 52

v. d. Veen, J. S., v. d. Waaij, B., Lazovik, E., Wijbrandi, W., & Meijer, R. J. 2015. Dynamically scaling apache storm for the analysis of streaming data. *Pages 154–161 of : 2015 ieee first international conference on big data computing service applications*. 32

Van Essen, David C, Ugurbil, Kamil, Auerbach, Edward, Barch, Deanna, Behrens, TEJ, Bucholz, Richard, Chang, Acer, Chen, Liyong, Corbetta, Maurizio, Curtiss, Sandra W, *et al.* 2012. The human connectome project : a data acquisition perspective. *Neuroimage*, **62**(4), 2222–2231. 71

Vapnik, Vladimir N. 1995. The nature of statistical learning. *Theory*. 16

Vavilapalli, Vinod Kumar, Murthy, Arun C, Douglas, Chris, Agarwal, Sharad, Konar, Mahadev, Evans, Robert, Graves, Thomas, Lowe, Jason, Shah, Hitesh, Seth, Siddharth, *et al.* 2013. Apache hadoop yarn : Yet another resource negotiator. *Pages 1–16 of : Proceedings of the 4th annual symposium on cloud computing*. 29

Wang, Hao, Zhuang, Bojin, Chen, Yang, Li, Ni, & Wei, Dongxia. 2018. Deep inferential spatial-temporal network for forecasting air pollution concentrations. *arxiv preprint arxiv :1809.03964*. 57

Wang, Jiantao, He, Caifeng, Liu, Yijun, Tian, Guangjian, Peng, Ivy, Xing, Jia, Ruan, Xiangbing, Xie, Haoran, & Wang, Fu Lee. 2017a. Efficient alarm behavior analytics for telecom networks. *Information sciences*, **402**, 1–14. 52, 57

Wang, Xiaosong, Peng, Yifan, Lu, Le, Lu, Zhiyong, Bagheri, Mohammadhadi, & Summers, Ronald M. 2017b. Chestx-ray8 : Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. *Pages 2097–2106 of : Proceedings of the ieee conference on computer vision and pattern recognition*. 73, 80

White, Tom. 2012. *Hadoop : The definitive guide*. " O'Reilly Media, Inc.". vi, 27

Widmer, Antoine, Schaer, Roger, Markonis, Dimitrios, & Müller, Henning. 2014. Gesture interaction for content–based medical image retrieval. *Pages 503–506 of : Proceedings of international conference on multimedia retrieval*. 71

Witten, Ian H, & Frank, Eibe. 2002. Data mining : practical machine learning tools and techniques with java implementations. *Acm sigmod record*, **31**(1), 76–77. 7

Xiang, Evan Wei, Cao, Bin, Hu, Derek Hao, & Yang, Qiang. 2010. Bridging domains using world wide knowledge for transfer learning. *Ieee transactions on knowledge and data engineering*, **22**(6), 770–783. 58

Yin, Hongzhi, Sun, Yizhou, Cui, Bin, Hu, Zhiting, & Chen, Ling. 2013. Lcars : a location-content-aware recommender system. *Pages 221–229 of : Proceedings of the 19th acm sigkdd international conference on knowledge discovery and data mining*. 25

Zhang, Jian. 2011. Deep transfer learning via restricted boltzmann machine for document classification. *Pages 323–326 of : 2011 10th international conference on machine learning and applications and workshops*, vol. 1. IEEE. 58

Zhang, Tong. 2001. An introduction to support vector machines and other kernel-based learning methods. *Ai magazine*, **22**(2), 103. 16

Zhou, Zhi-Hua. 2012. *Ensemble methods : foundations and algorithms*. CRC press. 13

الملخص

شهدت السنوات الأخيرة نمواً سر يع في البيانات المستعملة وذلك بسبب المعلومات المتداولة في الشركات وبيانات مستخدمي الشبكات الاجتماعية وكذا تقنية إنترنت الأشياء واستخدام الأجهزة المحمولة. في هذا الصدد، طرح الباحثون المهتمون العديد من الأسئلة التي من شأنها يتم استخراج المع ارف من البيانات الضخمة. يعرف الباحثون البيانات الضخمة كونها تتميز بالحجم الكبير، التنوع، الصحة، القابلية للتحصيل مع السهولة وكذا السرعة. تتمثل رؤيتنا في معالجة الرابط بين التنقيب في البيانات والبيانات الضخمة. في هذا العمل، قدمنا بنية متعددة الطبقات لاستخراج المعارف من البيانات الضخمة. تسمح هذه البنية بمعالجة فعالة لعملية استخراج المعارف. تتميز الهندسة التي اقترحناها بكونها متعددة الطبقات. ولهذ الغرض قدمنا مساهمتين. الأولى هو استخدام خوارزميات عميقة موزعة لتحسين محورين مهمين للغاية وقت التنفيذ ودقة عمليات تنقيب البيانات. المساهمة الثانية ترتكز على دمج مصادر البيانات المختلفة لتحقيق هدف محدد.

الكلمات المفتاحية : البيانات الضخمة، التنقيب في البيانات ، هندسة البيانات الضخمة، التعلم العميق الموزع.

Abstract:

During this decade, the growth of data is rapid due to the information circulating in companies, the data of social media users, the technology of the Internet of Things, and the use of mobile devices. This growth raises many questions for researchers to extract knowledge from large masses of data. This research discipline is Data Mining for Big Data. Several researchers define Big Data by data characterized by its volume, variety, velocity, veracity, volatility, and value. Our vision is to address the link between Data Mining and Big Data. In this work, we propose a multi-layered architecture for data mining in Big Data systems. This architecture allows the efficient processing of the data mining process. Our architecture has two contributions. The first is the use of deep and distributed algorithms to improve two critical areas of data mining process runtime and learning rate. In the second contribution, we focus on the integration of various data sources.

Keywords: Big Data, Data Mining, Architecture Big Data, Distributed Deep learning.