Order N^{o}: ../../..

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
UNIVERSITY OF KASDI MERBAH OUARGLA

Faculty of New Information Technologies and Communication
Department of Computer science and technology of Information

THESIS

Thesis submitted in partial fulfillment of the requirements for the degree of 3rd **Cycle LMD Doctorate**

Option : **Artificial intelligence**

By : **Salah Eddine HENOUDA**

Theme

Une approche par l'apprentissage approfondi pour l'extraction des connaissances dans les big data

Publicly defended on :25/02/2024 before the jury composed of :

Belal KHALDI	*MCA*	*at Ouargla University*	*President*
Fatima Zohra LAALLAM	*Pr.*	*at Ouargla University*	*Supervisor*
Okba KAZAR	*Pr.*	*at Biskra University*	*Co-supervisor*
Khaled REZEG	*Pr.*	*at Biskra University*	*Examiner*
Oussama AIADI	*MCA*	*at Ouargla University*	*Examiner*
Mohammed El amine ABDERRAHIM	*MCA*	*at Ouargla University*	*Examiner*

Acknowledgements

I would like to express my deepest gratitude to my family for their unwavering love, encouragement, and support throughout my academic journey. Their patience, understanding, and sacrifices have been invaluable, and I am profoundly grateful for their constant belief in me.

I am indebted to my supervisors Pr. Fatima Zohra LAALLAM and Pr. Okba KAZAR for their guidance, expertise, and mentorship. Their insightful feedback, constructive criticism, and dedication have been instrumental in shaping this thesis and my overall growth as a researcher.

I extend my appreciation to the faculty members and staff at Kasdi merbah university for their contributions to my education and research experience. Their passion for knowledge and commitment to excellence have inspired me to strive for academic excellence.

I would also like to thank my friends and colleagues for their support, encouragement, and camaraderie throughout this journey. Their friendship and collaboration have made the academic pursuit more enjoyable and rewarding.

Lastly, I acknowledge the contributions of all individuals and organizations whose work has influenced and enriched this thesis.

Abstract

In the realm of predictive analytics, two major issues capture attention: mobility prediction and breast cancer diagnosis. Although these problems may seem unrelated, they are crucial in their respective domains. Mobility prediction is essential for effective urban planning, while accurate diagnosis of breast cancer can save lives. In this thesis, we tackle two problems: (i): Mobility Prediction (MP) problem in big data, i.e, next location prediction of mobile users. (ii): Breast Cancer (BC) classification problem in high dimensionality datasets. We propose four contributions which are: (i): mini survey: we presented some of the well-known solutions used for reduce datasets high dimensionality, and we provided a comparison between the presented solutions. (ii): we introduced a practical comparison using two different classifiers (Multi Layer Perceptron (MLP) and Support Vector Machine (SVM)) combined with five different dimensionality reduction techniques, in order to understand the affect of high dimensionality on classifying Breast Cancer (BC). The results showed that using dimensionality reduction techniques increased the classification accuracy in some cases. The former, also showed that choosing the wrong combination of dimensionality reduction algorithms may lead to a worse results. The following accuracies are some of the results that we got: MLP-PCA, and MLP-ISOMAP outperformed simple MLP model with 0.7% of classification accuracy. SVM-PCA, and SVM-ISOMAP outperformed simple MLP with 0.3%. An example of choosing a bad combination of a dimensionality reduction algorithm and an MLP classifier is illustrated in the following example: MLP-AE, and MLP-RFE decreased the classification accuracy by 1.7% compared to simple MLP (iii): WP-BERTA: our proposed solution for next location prediction problem. The former is a combination of Bertwordpiece embedding algorithm and Transformer Roberta algorithm. (iv): WP-CamemBERT: our proposed solution for next location prediction problem. The former is a combination of Bertwordpiece embedding algorithm and Transformer CamemBERT algorithm. We showed that our solutions (WP-BERTA, and WP-CamemBERT) outperformed state-of-the-art solutions by increasing the next location prediction accuracy by at least 3% compared to the state-of-the-art algorithms.

Keywords Machine Learning; Deep Learning; transformer; Wi-Fi; mobility traces; Next location prediction; Neural Networks; Big Data; Dimensionality Reduction; Breast Cancer prediction; Medical datasets.

ملخص

في مجال التحليل التنبؤي، تستحوذ مشكلتان رئيسيتان على الاهتمام: التنبؤ بالتنقل وتشخيص سرطان الثدي. على الرغم من أن هذه المشاكل قد تبدو غير مرتبطة، إلا أنها أساسية في مجالاتها المختلفة. التنبؤ بالتنقل أمر أساسي للتخطيط الحضري الفعال، بينما يمكن أن ينقذ التشخيص الدقيق لسرطان الثدي حياة الناس. في هذه الرسالة: نعالج مشكلتين هما (1): مشكلة التنبؤ بالتنقل (MP) في البيانات الضخمة، أي التنبؤ بالموقع التالي لمستخدمي الهاتف المحمول. (2): مشكلة تصنيف سرطان الثدي (BC) في مجموعات البيانات عالية الأبعاد. نقترح أربع مساهمات وهي:(1): استطلاع صغير (mini survey): قدمنا بعض الحلول المعروفة التي تستخدم لتقليل مجموعات البيانات ذات الأبعاد العالية، وقدمنا مقارنة بين الحلول المقدمة.(2): قدمنا مقارنة عملية باستخدام مصنفين مختلفين (Multi-Layer Perceptron (MLP) و Support Vector Machine(SVM) جنبا إلى جنب مع خمس تقنيات مختلفة لتقليل الأبعاد، من أجل فهم تأثير الأبعاد العالية على تصنيف سرطان الثدي (BC). أظهرت النتائج أن استخدام خوارزميات تقليل الأبعاد زاد من دقة التصنيف في بعض الحالات. كما أظهرت الدراسة السابقة أن الاختيار الخاطئ لتركيبة خوارزميات تقليل الأبعاد مع خوارزميات التنبأ قد يؤدي إلى نتائج أسوأ. تظهر النتائج التالية دقة التصنيف المتحصل عليها من خلال دمج خوارزميات تقليل البعاد مع خوارزميات التنبأ حيث :تفوق كل من ACP-PLM و PAMOSI-PLM علي النموذج البسيط PLM بنسبة دقة تصنيف قدرها ٧.٠٪. وكذلك تفوق ACP-MVS و PAMOSI-MVS علي PLM البسيط بنسبة ٣.٠٪. في حين تظهر النتائج ان التركيبة السيئة لخورزميات تقليل الابعاد مع PLM البسيط والمتمثلة في PLM: EA-PLM و EFR-PLM خفضت دقة التصنيف بنسبة ٧.١٪ مقارنة ب PLM البسيط. (3): WP-BERTA: حلنا المقترح لمشكلة التنبؤ بالموقع التالي لمستخدم الهاتف المحمول. هذا الحل هو عبارة عن مزج طريقتين هما تضمين الأجزاء الكلماتية Bertwordpiece وهندسة Transformer Roberta. (4): WP-CamemBERT:حلنا المقترح لمشكلة التنبؤ بالموقع التالي. هذا الحل هو عبارة عن مزج طريقتين هوما تضمين الأجزاء الكلماتية Bertwordpiece وهندسة Transformer CamemBERT. لقد أظهرنا أن حلولنا (WP-BERTA و WP-CamemBERT) تفوقت على أحدث الحلول من خلال زيادة دقة التنبؤ بالموقع التالي لمستخدم الهاتف المحمول بنسبة 3٪ على الأقل مقارنة بأحدث الأساليب. الكلمات المفتاحية: التعلم الآلي; التعلم العميق; تقليل الأبعاد ; الشبكات العصبية; التنبؤ بسرطان الثدي; مجموعات البيانات الطبية ، المحولات. واي فاي؛ آثار التنقل التنبؤ بالموقع التالي ؛ البيانات الضخمة.

Résumé

Dans le domaine de l'analyse prédictive, deux problématiques majeures retiennent l'attention : la prédiction de la mobilité et le diagnostic du cancer du sein. Bien que ces problèmes puissent sembler sans lien, ils sont cruciaux dans leurs domaines respectifs. La prédiction de la mobilité est essentielle pour une planification urbaine efficace, tandis qu'un diagnostic précis du cancer du sein peut sauver des vies. Dans cette thèse, nous abordons deux problèmes qui sont :(1) : Problème de prédiction de mobilité (MP) dans le big data, c'est-à-dire la prédiction de l'emplacement suivant des utilisateurs mobiles.(2) : Problème de classification du cancer du sein (BC) dans les bases de données de haute dimensionnalité. Nous proposons quatre contributions qui sont : (1) : mini Survey : nous avons présenté certaines des solutions bien connues qui sont utilisées pour réduire les bases de données de haute dimensionnalité, et nous avons fourni une comparaison entre les solutions présentées. (2) : nous avons introduit une comparaison pratique en utilisant deux classificateurs différents (Multi-Layer Perceptron (MLP) et Support Vector Machine (SVM) combinés avec cinq techniques différentes de réduction de dimensionnalité, afin de comprendre l'effet de la haute dimensionnalité sur la classification du cancer du sein (BC). Les résultats ont montré que l'utilisation des algorithmes de réduction de la dimensionalité a augmenté la précision de classification dans certains cas. Cependant, il a également été démontré que le choix d'une mauvaise combinaison d'algorithmes de réduction de la dimensionalité peut conduire à de moins bons résultats. Les précisions suivantes sont quelques-uns des résultats que nous avons obtenus : MLP-PCA et MLP-ISOMAP ont surpassé le modèle MLP simple avec une précision de classification de 0,7%. SVM-PCA et SVM-ISOMAP ont surpassé le MLP simple de 0,3%. Un exemple de choix d'une mauvaise combinaison d'algorithme de réduction de la dimensionalité et d'un classifieur MLP est illustré dans l'exemple suivant : MLP-AE et MLP-RFE ont diminué la précision de classification de 1,7% par rapport au MLP simple. (3) : WP-BERTA : notre solution proposée pour le problème de prédiction de prochaine emplacement a visiter par un utilisateur. Le premier est une combinaison de la méthode d'intégration Bertwordpiece et de l'architecture Transformer Roberta.(4) : WP-CamemBERT : notre solution proposée pour le problème de prédiction de prochaine emplacement a visiter par un utilisateur. Le premier est une combinaison de la méthode d'intégration Bertwordpiece et de l'architecture Transformer CamemBERT. Nous avons montré que nos solutions (WP-BERTA et WP-CamemBERT) surpassaient les solutions de l'etat de l'art en augmentant la précision de la prédiction de prochain emplacement a visiter d'au moins 3 % par rapport aux méthodes de l'etat de l'art.

Mots clés Apprentissage automatique ; apprentissage approfondi ; réduction de la dimensionnalité ; réseaux de neurones ; Prédiction du cancer du sein, Bases de données médicales, transformateur ; Wi-Fi ; les traces de mobilité ; Prochaine prédiction de l'emplacement ; Big Data.

Contents

List of Tables

List of Figures

acronyms

AA: Average accuracy

AE: Auto encoder

AI: artificial intelligence

ANN: Artificial neural network

AP: Access point

BC: Breast cancer

CHAID: CHI-squared automatic interaction detection

CMAC: Cerebellar model articulation controller

CNN: Convolution neural network

DBN: Deep belief network

DL: Deep learning

DR: Dimensionality reduction

DRT: Dimensionality reduction technique

DT: Decision tree

GA: Genetic algorithm

GNA: Generative adversarial network

GONN: Genetically optimized neural network

HDD: high dimensional data

HDFS:	Hadoop distributed file system
ID3:	Iterative dichotomiser 3
ISOMAP:	Isometric feature mapping
KNN:	K-nearest neighbor
KS:	Kappa statistic
LHNFC:	Linguistic hedges neuro-fuzzy classifier
LSTM:	Long-short term memory
L-SVM:	Linear support vector machine
LH:	Linguistic hedge
K-SVM:	K-means and SVM combination algorithm
MDS:	Multidimensional scaling
ML:	Machine learning
MLP:	Multi layer perceptron
MP:	Mobility prediction
MSE:	Mean square error
NB:	Naive bayes
NN:	neural network
NLU:	Natural language understanding
NLP:	Natural language processing
NMF:	Non-negative matrix factorization
OA:	Overall accuracy
PCA:	Principal component analysis
PLM:	Permutation language modeling

RF:	Random forest/ Rotation forest
RNN:	Recurrent neural network
RBM:	Restricted boltzman machine
RFE:	Recursive feature elimination
SVD:	Singular value decomposition
SVM:	Support vector machine
SNMTF:	Spatial non-negative matrix tri factorization
SVCMAC:	Self validation cerebellar model articulation controller
ST-RNN:	Spatial temporal recurrent neural network
STF-RNN:	Space time feature based recurrent neural network
STS-LSTM:	Spacial-temporal-semantic-neural network
TSNE:	T-distributed stochastic neighbor embedding
WP-Berta	Word peace roberta
WDBC:	Breast cancer wisconsin
ZB:	Zettabyte

Chapter 1

General Introduction

Contents

1.1 Context

In the last few years, and because of the large production of heterogeneous information, i.e. structured, unstructured and semi-structured data, big data has emerged as one among the most important technical words in multiple domains. In this regard, the use of traditional mechanisms and theories for storing, processing, and analyzing big data has become difficult or impossible. Thus, the necessity for providing new ways to deal with big data is needed [1, 2]. Therefore, machine learning (ML) and data mining techniques perfectly replaced the traditional analytic algorithms and succeeded in multiple domains to extract hidden patterns and analyze big data for extracting value, i.e. big data analytics. It should be noted that the key success behind the previous mentioned techniques, i.e. (ML) and data mining techniques, is the nature of big data which is highly in dimensions and composed of non linear data [3].

1.2 The problems addressed in the in the thesis

1.2.1 Mobility prediction problem

In now days, we live an incredible raise in mobile technologies, in which they have become a part of our lives. According to [4], about 4 billion of mobile users spend more than six hours on the net, while the use of mobile phones and social media applications on mobile devices have attained respectively 67%, 42% in the world in 2019. Thus, a necessity of providing better user experience with personalized services of better performance is needed.

Understanding users movements can help improving such services. It is important to know that user movements which are constrained by some conditions and usual users habits, are not totally randomized [5]. It is also important to know that the privacy of mobile users is important. Thus, relying only on the user's context to make service personalization and provide better user experience is needed [6].

One common solution for the previous two problems, i.e. maintaining privacy and understand user's movements, is mobility prediction. The latter rely only on the user's historical context, i.e. maintaining privacy, to make predictions, i.e. extract hidden patterns of movements. So, mobility prediction can be defined as: predicting a user movements, i.e. learning and inference, using his historical personal context, i.e. previous knowledge [5].

It is hard to accurately predict users mobility because of their behaviour. The latter which is related to various dimensions, i.e. multi-dimensional problem, notably the temporal and spacial dimensions, changes automatically concerning the user and varies from one user to another [7].

1.2.2 Breast cancer classification in high dimensional tabular datasets problem

Breast Cancer (BC) is one of the life-threatening and dreaded cancers among women in the world. [8] asserts that this type of cancers occurs in the lining cells of the ducts or lobules in the glandular tissue of the breast. Fundamentally, in the end of 2020, (BC) turns out to be the most frequently occurring cancer worldwide [8]. The fact that, in the last 6 years, 7.8 million women were diagnosed with (BC) necessitates the inclusion of artificial intelligence technologies such as Machine Learning (ML). These technologies typically seek to predict and understand the disease better in order to avoid deaths and life loses. In essence, several research studies were carried out for the sake of diagnosing (BC). Examples of such studies

are [9, 10, 11, 12] and [13].
In the last few years and with the large production of data, high dimensional data (HDD) have been applied in numerous fields such as: education, social media, web and medicine, etc [14]. Using (HDD) for (BC) classification is challenging because of the curse of dimensionality [14, 15, 16]. The latter, which introduces a large search space affects negatively the models performance (classification accuracy and pattern recognition) by increasing the models computations and result more complexity [15, 16].

1.3 thesis structure

The rest of the thesis is splited into three chapters that are:

- Chapter 2: Big Data and Machine Learning Essentials,
- Chapter 3: Contributions, which includes related work. The former is made up of the following sections:
  - Section 3.1: A Comparative study for dimensionality reduction techniques for big data.
  - Section 3.2: On the effectiveness of Dimensionality Reduction Techniques on High Dimensionality Datasets.
  - Section 3.3: Next location prediction using Transformers.
  - Section 3.4: WP-CamemBERT for next location Prediction.

At the end, a General Conclusion is presented as chapter 4.

Chapter 2

Big Data and Machine Learning Essentials

Contents

2.1 Introduction

This chapter illustrates machine learning and big data fundamentals. It presents big data definition and characteristics, followed by describing its architectures and its different data types. This chapter also explains big data storage and processing concepts and technology, followed by introducing the term big data analytics and its techniques. Then, big data tools and applications are presented in section 2.2.6 followed by challenges faces by big data technologies. Next, we introduce the basics of ML as well as DL which is a subclass of it. First, we provide ML definition in section 2.3.1, followed by section 2.3.2, which outlines the tasks that a ML algorithm can finish. The learning paradigms as well as some of the well-known ML algorithms are presented respectively in sections 2.3.3, and 2.3.4. Section 2.3.5 illustrates deep learning importance and the challenges that motivate it. Sections 2.3.6, 2.3.7, and 2.3.8 provide some of DL basics (definition, learning process in DL, and design architectures). Finally, a conclusion summarizes the content of the chapter is presented at the end.

2.2 Big data

2.2.1 Big data definition

The term ”big data” refers to the different techniques that are used to analyze or extract value and useful information from too large or complex data. The latters They have been gathered from various sources (sensors, devices, etc..) are very challenging to process with the classical processing technologies [17].

2.2.2 Big data characteristics

A big data is described by one or more of the following characteristics. The latters are also called v's of big data, as figure 2.1 shows.

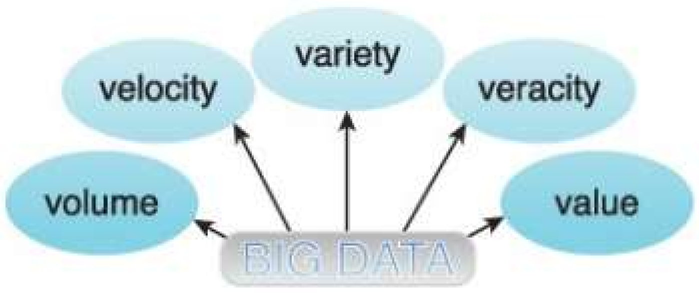

Figure 2.1: V's of big data [18].

- **Volume** It is referring to the large amount of data which is growing daily in a exponential manner, and needs different big data solutions and techniques to be processed and stored. This voluminous data came from distinct sources like social media, sensors, online transactions, etc..) [18]. Figure 2.2 demonstrates the amount of data created daily by some organizations and world wide users.

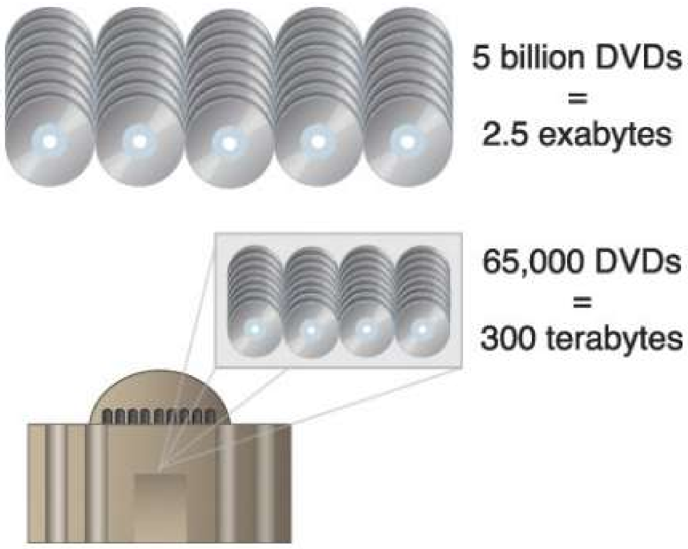

Figure 2.2: the daily created amount of data by some organizations and world wide users [18].

- **Velocity** Big data which is voluminous, can be generated and accumulated within a brief period of time. The term velocity refers to speed rate of data generating, data analyzing, and result returning (data processing) [18]. Figure 2.3 demonstrates and example of the amount of data generated in one minute period of time, and its relation with data velocity.

- **Variety** It means heterogeneous data that composes big data (Unstructured, semi-structured, and structured data). Big data variety introduces some challenges like: data integration, data storage, and data processing [18]. Figure 2.5 presents an example of high data variety that came from distinct sources.

- **Veracity** presents the untrustworthiness in some sources of data [19]. Today, a lot of data sources are available. This makes the data precision, quality, and trust these sources a big issue for big data [20].

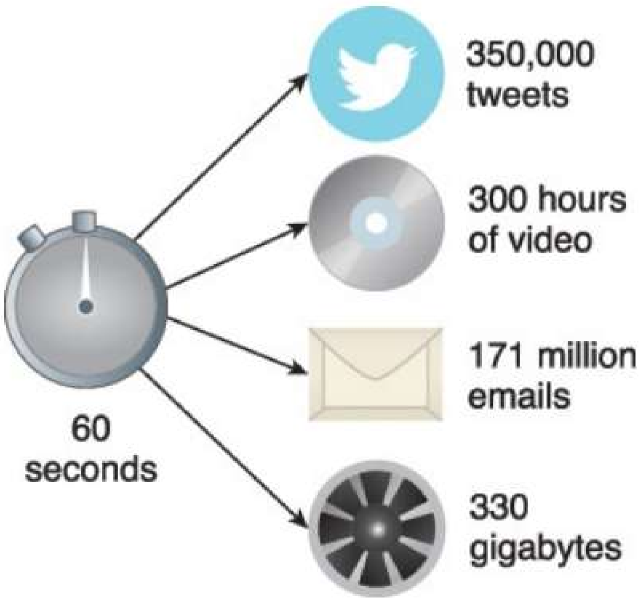

Figure 2.3: Example of big datasets generated in one minute with high velocity [18].

- **Value** It refers to the meaningful and useful information and patterns extracted from big data. The two big data characteristics value and veracity are related to each other. The more the quality and fidelity big data have, the more value it contains. Also the time affects big data value characteristic. The more time it needs to be processed, the less value it contains [18]. Figure 2.4 demonstrates how big data veracity and the time to generate analytical results, affects big data value.

2.2.3 Types of data used in big data

Human interactions with machines like online services, and machine programs, hardware devices are the two sources of data that consist big data [18]. Therefore, the following list presents the different data types that are used in big data.

- **Structured data** is the data that corresponds to some data structures, models, or schemes. It is used usually to catch relation between entities [18].

- **Unstructured data** It is a data that does not correspond to any data structure, model, or scheme like figure 2.5 shows. Unstructured data forms the majority type of all types of data (80% of data enterprises are unstructured) [18]. It can be stored using relational databases (as binary large object), or using NoSQL databases. It can not queried or processed directly using SQL [18].

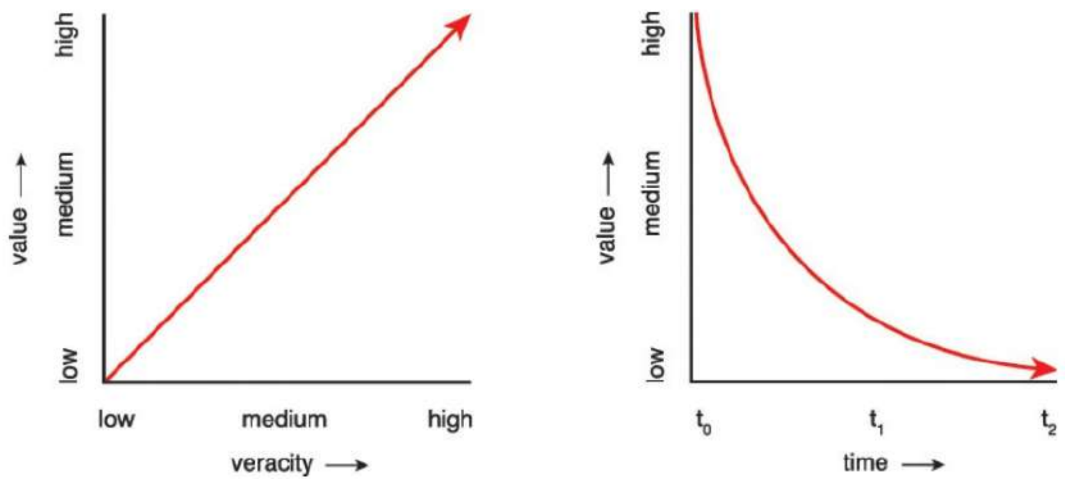

Figure 2.4: The relationship between time, data veracity, and data value [18].

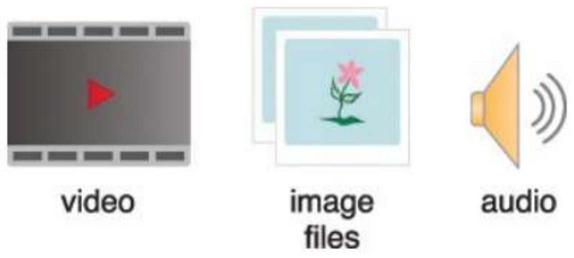

Figure 2.5: Some types of unstructured data [18].

- **Semi-structured data** Usually, this type of data is hierarchical or graph based. Thus, an implicit level of structure exists. Jason and XML files are generally used for representing semi-structured data [18].

2.2.4 Big data architecture components

All big data architectures are developed to control data during storage, processing, and analysis. Kappa, internet of things IOT, and lambda are famous big data architectures [17]. In the list that follows, we present the main components of a big data architecture.

- **Data sources** Sources of big data can be from one unique source or from multiple sources. the following are some sources of data: data stores, static files, and real-time sources [21].
- **Data storage** Data is stored in a distributed manner (distributed files). This method can store all data formats, with different and high data volumes. The

former is called data lake [17].

- **Batch processing** Usually and because of the high data volumes, batch jobs are used to treat the large data (filtering, aggregation, analysis, and preparation) [22].
- **Real-time message ingestion** Capturing and storing real-time messages (from real-time sources) is important for stream processing. This data store is used from big data solutions as messages buffer, queuing semantics, and for reliable delivery [23].
- **Stream processing** Is the process of treating real-time data at the same time it was created or received [23].
- **Analysis and reporting** Is the process of extracting useful ideas and insights from big data. Mathematical, statistical, and some visualisation techniques are used to complete that process [24].
- **Analytical data store** The analytical data store is an optimized data base that stores data in a structured format. The latter can help in the analysis process (querying the data) [17].
- **Orchestration** A big data solution is composed of a set repeated processing operations, encapsulated in workflows. To automate the latters, an orchestration technology is needed [17].

2.2.5 Big data analysis techniques

Big data analytics: is a set of analytical algorithms and processes aims at facilitating and making more effective decisions by extracting useful insights and knowledge from big data [25].

This section and the following list summarize the basic techniques of data analysis as mentioned in [18].

- **Quantitative analysis** This technique quantifies the relationships and useful patterns in a given dataset. The samples used in this type of data analysis are large, i.e. the use of statistical methods to extract large number of observations. Thus, the captured observations and results in one sample can be generalized to the hole dataset. Figure 2.6 describes quantitative analysis.
- **Qualitative analysis** This technique describes the qualities of data using textual words. Thus, numerical computations or comparisons can not be used in the type of analysis. Unlike quantitative analysis, this technique uses small samples.

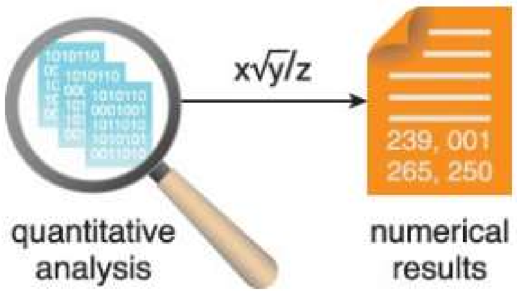

Figure 2.6: Quantitative analysis and its numerical output [18].

Therefore, we can not generalize the observations for the hole dataset. Figure 2.7 illustrates qualitative analysis.

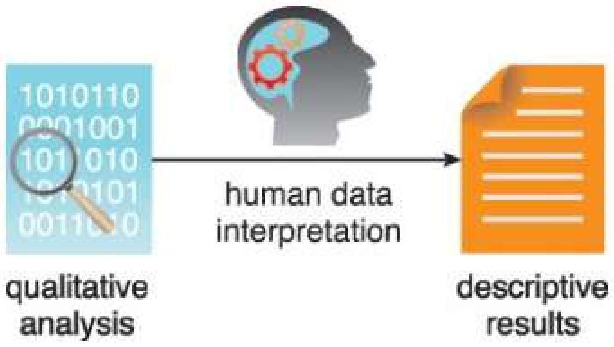

Figure 2.7: Qualitative analysis and its descriptive outputs [18].

- **Statistical analysis** It refers to the use of statistical methods to analyse and describe a dataset. It can be quantitative or qualitative. A/B testing, correlation, and regression are 3 type of statistical analysis.
- **Machine learning** Is the process of combining human knowledge and machine powerfulness to automate solving problems without human interventions. ML has multiple techniques like: classification and clustering.
- **Semantic analysis** Is the process of extracting useful patterns and insights from speech and textual data. Its goal is to make machines understand speech and textual data as humans do.
- **Visual analysis** Is the process of extracting useful patterns and insights through the use of graphical representations. understanding graphical representations is easier and quicker than understanding textual data. We mention the following

types of visual analysis. Heat maps, time series plots, network graphs, and spatial data mapping.

2.2.6 Big data tools and applications

There are a lot of tools and frameworks to use for processing and managing big data. In this section, we illustrate some of them which are presented in [26].

Hadoop ecosystem

Hadoop is composed of a set of frameworks. Each of the latters is specialized in completing a specific task. Hadoop frameworks are combined together and relying to each other to generate the final output [26]. The following sections demonstrate some of the key components of hadoop ecosystem.

Distributed file system component Distributed file system is the main component in the hadoop ecosystem. It is used to store semantics in the data cluster, data of any size, and files of any format. It provides data availability and protect from data loss. Two types of distributed file system are: hadoop distributed file system (HDFS), and distributed file system with posix compliance [26].

Distributed processing components Distributed processing components are distributed processing engines used to support various and distinct applications [26]. four categories of distributed processing components are presented as follows:

- first type is used only by hadoop ecosystem and its applications like hadoop Map Reduce and TeZ [26].
- Another type that is compatible only with hadoop and used by only a single application like Cloudera Impala distributed processing engine [26].
- Flink and spark are examples of the third type. The latter is compatible with multiple big data technologies as hadoop, and used by distinct applications [26].
- The last type is compatible with multiple big data technologies, and used only by a single application type like apache Drill and apache storm [26].

Application components Distributed processing components are used by application components to support distinct applications [26]. Various types of application components exist, we mention the following categories:

- **SQL components** This type of application components support SQL for processing and querying big data. Hive and drill are famous examples of it [26].

- **Data flow components** Are used to describe data transformations through data flow processing pipelines. Pig and cascading are two famous examples of data flow components [26].

- **Graph processing components** Graph processing components are used to process data which is represented with graph architecture (nodes and edges) like social networks and payment transactions. Giraph is a famous open source application used to process this type of data [26].

- **Modeling components** Modeling components are used to create predictive models in hadoop. Both statistical methods and machine learning algorithms are used for creating this type of models. Apache mahout is an open source example application used as modeling components on hadoop [26].

NoSQL databases

NoSQL databases have emerged in order to replace traditional relational databases. The latters were incapable of handling the high write/read requirements of web applications. MongoDB, casandra, and Hbase are three famous examples of NoSQL databases [26].

In memory databases

Contrasted to the database management systems which use the disk for storing data, in memory database management systems use the main memory of the device for storing and processing data. Thus, very fast processing of the data with no input/output to/from disk. There are a lot of in memory open source databases, we mention aerospike, hazelcat, and gemfire as examples. [26].

Streaming event processing technologies

Streaming processing technologies treat large amount streaming data which is continuously produced. The former use producer/consumer concept in treating streaming data. Each producer/consumer agent process in a specific manner streaming data and passes the results to another agent. There are many applications used for processing this kind of data, we mention flume, storm, and kafka as an open source examples [26].

Search engines

This type of search engines is based on distributed indexing mechanism, where big data can be indexed at a high speed rate. The generated index are distributed and stored over many data nodes. Apache solr, elasticsearch, and sphinx search are open source examples of big data search engines [26].

2.3 Machine Learning

2.3.1 Machine learning definition

Machine learning (ML) is a set of algorithms that can learn useful patterns automatically from data in order to complete a specific task, for example: classification task. An algorithm learned useful patterns with specific experience A, only if its performance enhanced with that experience [27].

2.3.2 Machine learning tasks

The thing that makes (ML) algorithms interesting is that they can solve hard problems that are exceedingly tough to resolve using fixed programs. This section aims at presenting some common tasks, i.e. models objective, that (ML) algorithms can solve and realise. In [27], the authors presented some tasks, we mention the following:

- **Classification:** Let K be the set $K = \{1, ..., m\}$ of m categories, i.e classes. Let X be a set of inputs $X = \{X_1, ..., X_n\}$, where $X_i, i = 1, .., n$ is a vector of features. Classification task may be described as the assignment procedure of some inputs $X_i, i = 1, ..n$ to a specific category $k_j, j = 1, .., m$. The former is defined mathematically as the function $f(x) = y$, where $y \subset \{1, .., m\}$.

- **Classification with missing inputs:** It is hard to classify some inputs with missing values, and rather than defining only one classification function, it is needed to define a set of classification functions to classify the input X with its missing inputs.

- **Regression:** This task is close to classification. The only difference is in the output format, in which the function used in regression may outputs any numerical number $X, where X \subset R$. Essentially, This regression is used in prediction relation tasks.

- **Transcription:** In this task, the (ML) model has a goal of transforming unstructured data into discrete text data.

- **Machine translation:** As its name, the (ML) model here tries to transform a sequence of words or symbols in a specific language into another equivalent sequence in other language.

- **Structured output:** In this type of task, the (ML) model outputs any data structure. The latter consists of elements with important relations between them.

- **Anomaly detection:** The (ML) model tries in this type of task to detect abnormal or non typical events and objects.
- **Synthesis and sampling:** The (ML) model has a goal of generating new synthesised data, i.e. new samples of data, that is similar to training data. The model here is called generative model.
- **Imputation of missing values:** The (ML) model in this type of task tries to predict missing parts in the new data inputs.
- **Denoising:** In this type of task, the (ML) model tries to clean the corrupted data and predict new clean version of it.
- **Dimensionality reduction:** aims to simplify complex data by reducing the number of input variables while preserving the essential information and structure within the dataset.

2.3.3 Learning paradigms

There are a lot of learning paradigms in the literature. The following sections introduce three well known of them that are: reinforcement learning, supervised learning, and unsupervised learning.

Supervised learning

A learning paradigm is called supervised when the training data that will be fed to the ML model, contains the correct outputs that should the ML model results when a specific input is given. The outputs in the training data are called labels, and the ML model corrects its weights based on the provided labels [28]. The following examples represent some problems that are solved through supervised learning paradigm.

Hand-written-digits classification problem [29]. In this problem, the ML model classifies each hand written image into one of the 10 presented categories $\{0, .., 9\}$. The learning is supervised because the data is labeled and for each input image, and output (category) from 0 to 9 is provided.

Breast Cancer Wisconsin Diagnostic (WDBC) classification problem [30]. In this problem, the ML model tries to categorize the input features into one of the two provided classes, B for benign, and M for malignant.

Unsupervised learning

A learning paradigm is called unsupervised when the training data that will be fed to the model is unlabeled. Unlabeled data is a data that does not contain any outputs that the ML model can use in the training process to supervise its learning. Thus, the ML model here is obliged to detect useful patterns from only the given input examples. The process of classifying unlabeled data is called clustering [28]. Figure 2.8 illustrates an example of clustering (unsupervised learning) some coins based only on two input features: mass, and size.

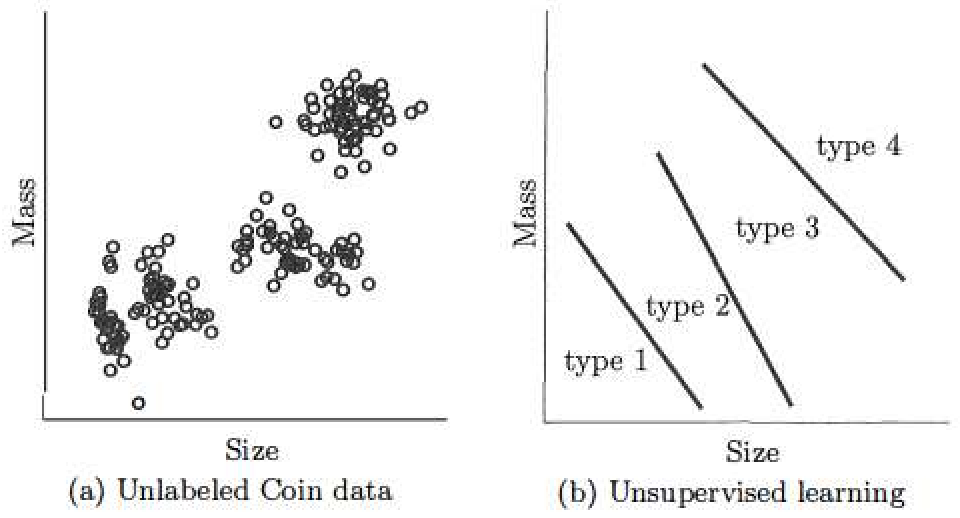

Figure 2.8: Coin classification through unsupervised learning [28].

Reinforcement learning

This Paradigm is based on the reward, penalty strategies. So, the agent,i.e. learner and decision maker, will be rewarded or punished based on its output actions with the environment. So, the agent will learn how to interact with its environment and chooses the optimal action for each situation [31, 28]. Figure 2.9 shows the interaction between reinforcement learning and its environment.

2.3.4 Some of machine learning algorithms

This section introduces the most common ML algorithms used in the literature.

- **Decision tree (DT).** A decision tree is a graph that is made up of nodes, and edges. Each node demonstrates a specific decision or condition, and each edge demonstrates the output of that decision or condition. The final nodes are called leaf nodes and they represent the final decision or class label. Thus, the process of classifying a given instance example will go down from the root node following some decisions/ conditions to the leaf node (class), which is the

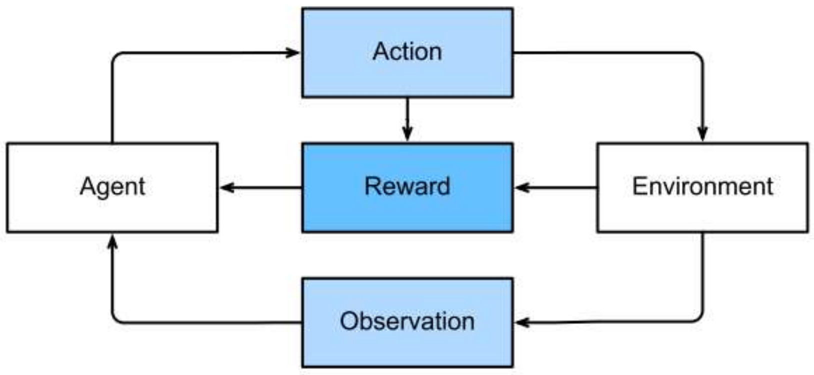

Figure 2.9: Reinforcement learning - environment interactions [32].

most convenient class for that input example [33]. The following list displays some of decision tree algorithms. [33].

- Classification and regression tree (CART).
- Iterative dichotomiser 3 (ID3).
- CHI-squared automatic interaction detection (CHAID).

Figure 2.10 shows an example of how decision tree algorithm works

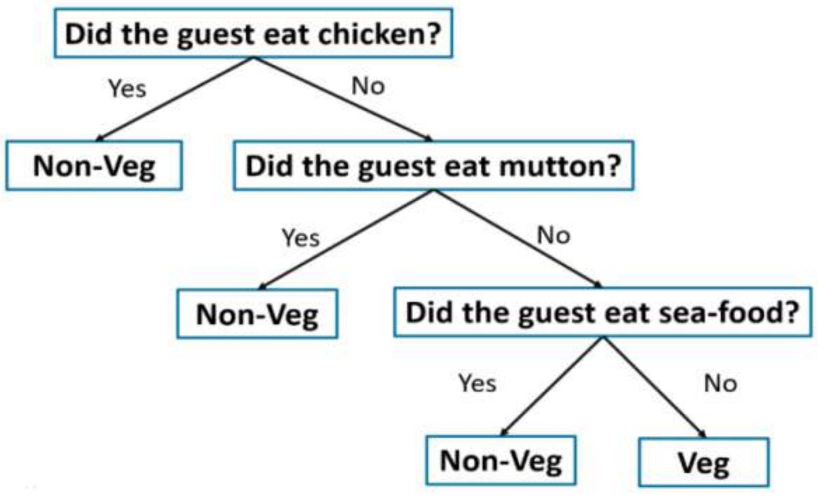
Did the guest eat chicken?
Yes
No
Non-Veg
Did the guest eat mutton?
Yes
No
Non-Veg
Did the guest eat sea-food?
Yes
No
Non-Veg
Veg

Figure 2.10: Decision Tree example [34]

- **Support vector machine (SVM).** SVM is used for regression/classification tasks through supervised learning paradigm. It represents data in n-dimensional space, where n refers to the number of features. Classes that are represented in that n-dimensional space are separated using margin calculation. Therefore, SVM separates classes using hyperplane, i.e. straight line or kernel trick for non linear problems [33, 34]. Figure 2.11 demonstrates how SVM separates two classes using a straight line.

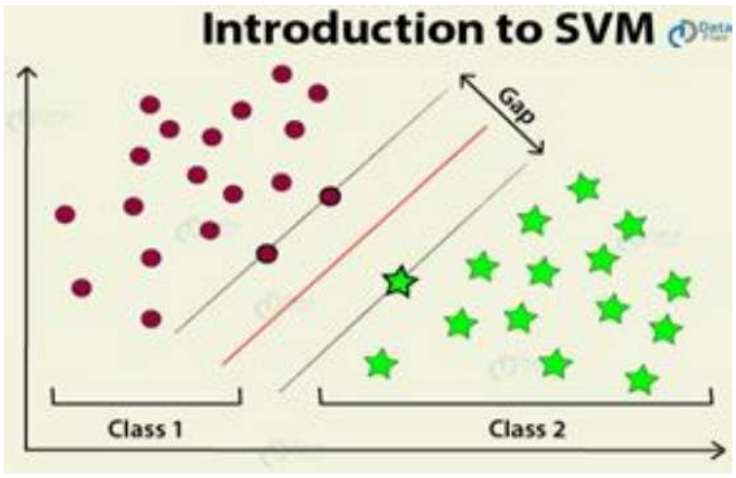

Figure 2.11: SVM example [34].

- **Naive bayes (NB).** It is a probabilistic ML algorithm according to the theorem of bayes of probability. It is used for clustering/ classification tasks, where it assumes that the present features in the dataset are unrelated. Thus, it classifies an input X to the class that has the maximum posterior probability [33]. Figure 2.12 demonstrates the posterior probability formula.

$$\underset{\text{Posterior Probability}}{P(c\,|\,x)} = \frac{\overset{\text{Likelihood}}{P(x\,|\,c)}\,\overset{\text{Class Prior Probability}}{P(c)}}{\underset{\text{Predictor Prior Probability}}{P(x)}}$$

$$P(c|X)=P(x_1|c)\times P(x_2|c)\times\cdots\times P(x_n|c)\times P(c)$$

Figure 2.12: NB posterior probability formula [34]

- **Principal component analysis (PCA).** PCA is utilized usually for feature selection and dimensionality reduction purposes. It works by choosing k orthogonal linearly uncorrelated vectors that fits the data the most, and then performing a change of basis of the data on the chosen vectors. The latters are called principal components. so PCA is the process of transforming data into lower dimensional data through orthogonality [34]. The following example represents the PCA of iris dataset that consists of 3 types of iris flowers and 4 attributes.

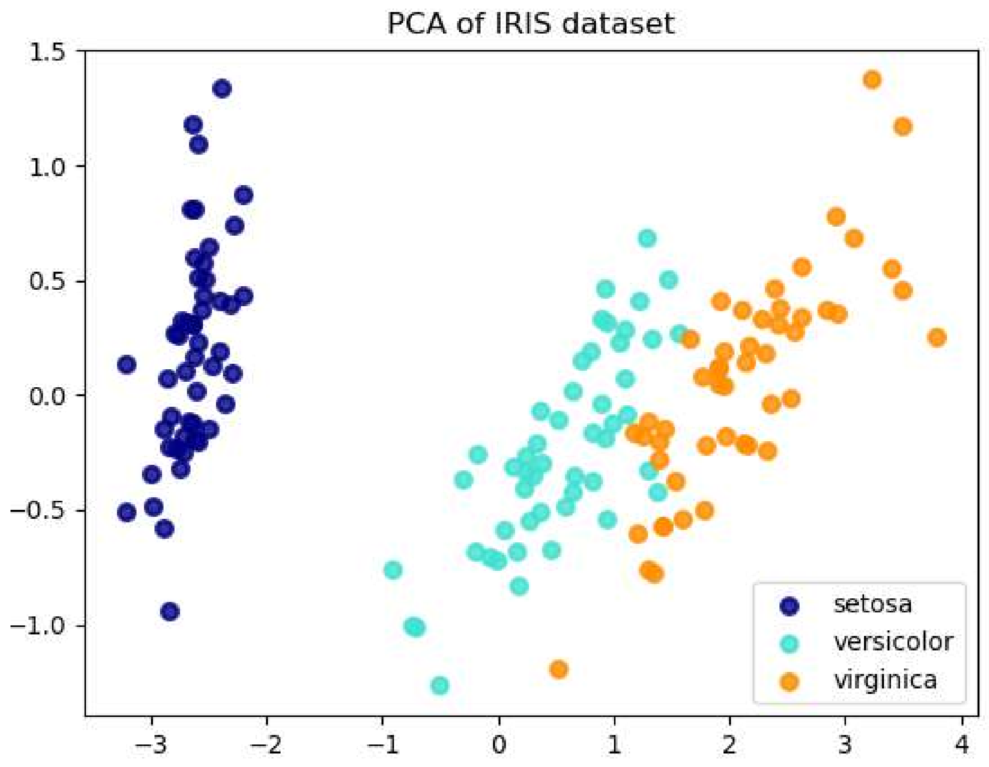

Figure 2.13: PCA of iris dataset [35].

- **K-means clustering.** It is an unsupervised learning algorithm used for clustering analysis. Its basic idea is to define k center points for k clusters, and then assigns each point in the dataset to the cluster that has the minimum mean. The algorithm repeats the process of rechoosing the clusters's centers and continue clustering until all the data is clustered [34].

- **K-nearest neighbor (KNN).** It is a supervised learning algorithm that is used for both classification and regression applications.
  In classification, the algorithm assigns an input X to the most common class

between its k-nearest neighbors.
In regression, the algorithm outputs for an input X, the average value of its k-nearest neighbors values.
Figure 2.14 demonstrates iris dataset [35] classification example, using KNN algorithm with k=1. Iris dataset consists of 3 classes and 4 attributes.

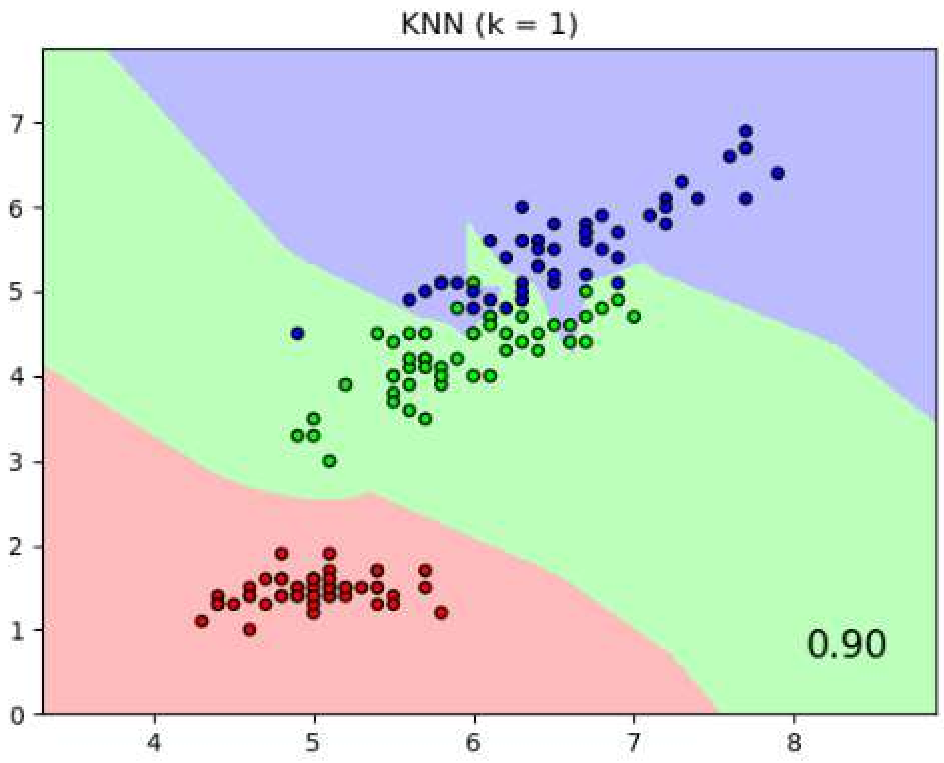

Figure 2.14: K-nearest neighbor with k=1 [35].

- **Random forests. (RF)** In fact, random forests RF are ensemble of decision trees combined together to form a forest. Each DT in the RF outputs a specific results. Therefore, the RF algorithm combines those results to generate a general output. RF are used usually for classification / regression tasks [33]. Figure 2.15 shows an illustration of the RF's general architecture.

- **Regression.** Regression algorithms like linear regression, logistic regression, and polynomial regression [33] are predictive algorithms that are used to find the relation between the independent input features and the output target. Their objective function is to decrease the distance between the data points and the function curve or line [33].

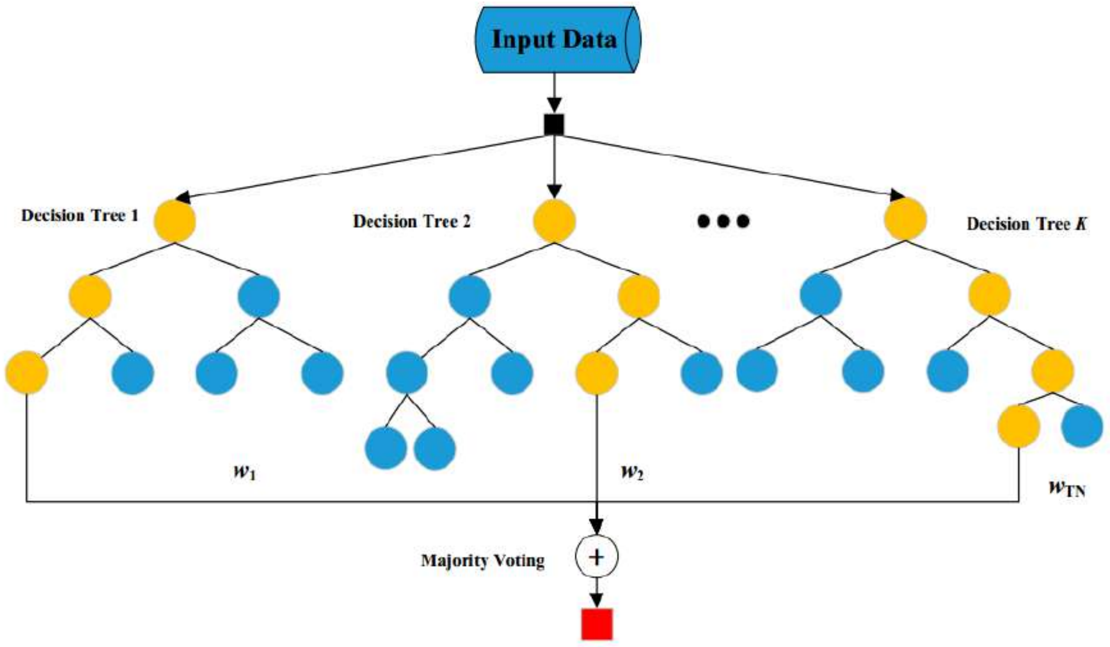

Figure 2.15: RF general architecture [36].

2.3.5 Challenges motivating deep learning

The traditional ML algorithms failed in solving both: central problems and in generalization, where the data examples are small compared to data dimensions (the provided examples does not cover all regions). Deep learning overcomes those drawbacks by a set of assumptions. The former supposes that the data is composed hierarchically of a set of features. Contrary to machine learning, the previous assumptions help deep learning in generalization by gaining exponentially relations between the provided examples and the existing regions [27].

2.3.6 Deep learning definition

Deep learning is a sub class of machine learning algorithms that uses artificial neural networks (ANN) to learn meaningful patterns in hierarchically manner. Deep learning algorithms attempts to learn higher representations from lower representations in several levels (hierarchy). Deep learning algorithms can be utilized in supervised, unsupervised, or semi- supervised learning paradigms [37].

2.3.7 Learning process in neural networks

The process of learning in neural networks consists of two phases. The first phase is called forward propagation. In the latter and after an initialization of the model's weights, the model uses a set of inputs X combined with the initialized weights to produce (predict) some outputs. After that, a cost function is applied to compute the error resulted in the firs phase. Note that making a neural network model learns means that a change in the model's weights must be applied to minimize the cost function (error). Therefore, a back propagation in the second phase is applied to propagate the error through the network, and to compute the influence of each weight on the cost function. This influence is computed via partial derivatives of the cost function with respect to the weights. Finally, adjusting the weights in proportion to the cost function using a specific algorithm like stochastic gradient descent [32].

2.3.8 Architectures design

Three main categories of artificial neural networks (ANN) are presented in the following section.

- **Convolution neural networks (CNNs)** A CNN is a type of ANNs that consists of stacked convolution and pooling layers one over the other. At the end of the network, a fully connected layer is presented generally to perform classifications. Convolution layers apply a set of filters on the inputs in order to extract useful patterns and create a feature map from the obtained patterns. While pooling layers are used to reduce computations and data dimensions by summarizing a set of features in a specific location in the feature map into one value in the next layer [38]. The following well-known examples of architectures are CNN based architectures proposed in the literature: AlexNet, ResNet, and googleNet [38]. Figure 2.16 shows the CNN's general architecture.

- **Pretrained unsupervised networks**

  - **Autoencoders (AEs)** An AE is a feed forward neural network that uses unsupervised learning in processing data (unlabeled data). AEs are used for dimensionality reduction tasks. The former composed of two parts: an encoder which is used to compress the input data into data of smaller dimensions, and a decoder which is responsible for reconstructing the input data by decompressing the outputs of the encoder [38]. Figure 2.17 shows the AE's general architecture.

  - **Generative adversarial networks (GANs)** GANs models use unsupervised learning in order to perform their tasks. Their main goal is to produce

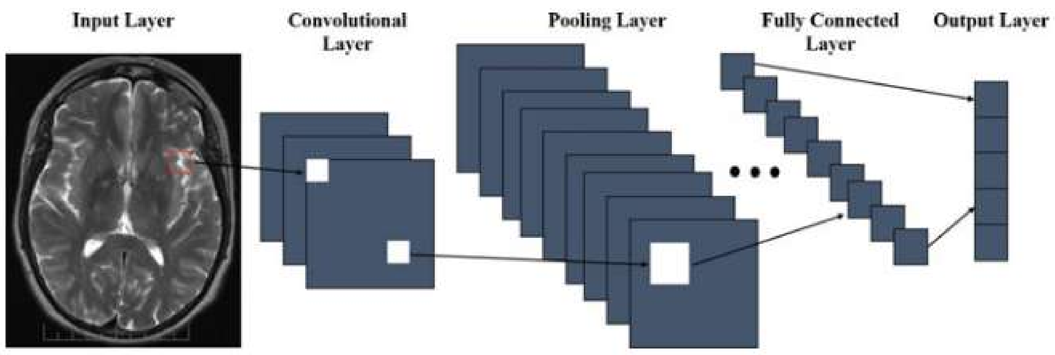

Figure 2.16: CNN's General architecture [38].

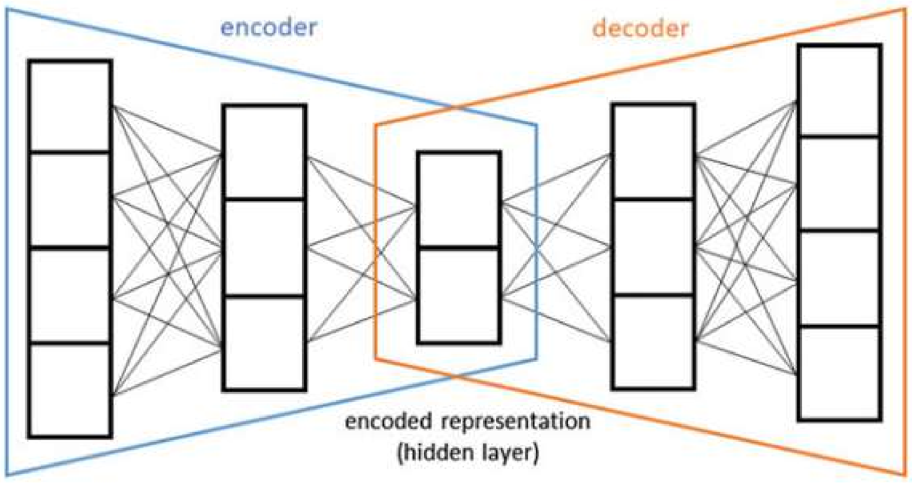

Figure 2.17: General architecture of an AE [38].

synthesized data that is similar to the provided training data. GANs are composed of a discriminator and a generator that are trained in parallel. The discriminator is used for distinguishing between real and synthesized data that is produced by the generator. The latter's goal is to augment the error rate of the discriminator, by producing synthesized data that is so close to the real one, i.e. the discriminator can not differentiate between synthesized and real data [38]. Figure 2.18 shows a general representation of a GAN model.

– **Deep belief networks (DBNs)** A DBN composed of multiple hidden layers. Each hidden layer is a stochastic recurrent neural network that is named RBM (restricted boltzman machine). Each of the latter is connected with two other RBMs, one is visible and the other is hidden [38], as figure 3.5

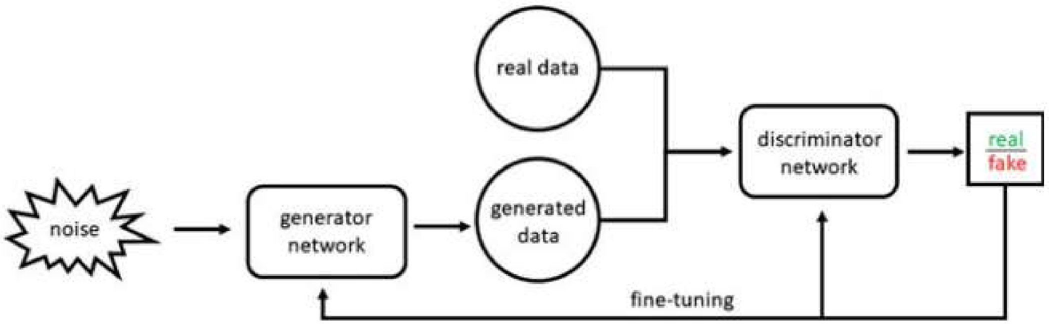

Figure 2.18: GNA's general representation [38].

shows.

- **Recurrent networks** Recurrent models are used for processing and treating sequential data. In this category we mention two structures that are: long-short term memory networks (LSTMs), and recurrent neural networks (RNNs).
  - **Recurrent neural networks RNNs** RNNs differs from other architectures by using internal memory. The latter is used to tel which information is important to keep compared with the past information. The output of an RNN will be fed as an input alongside the inputs. Therefore, the former affects the current decision of the RNN model [38]. Figure 2.19 shows the genera architecture of stacked RNN model.

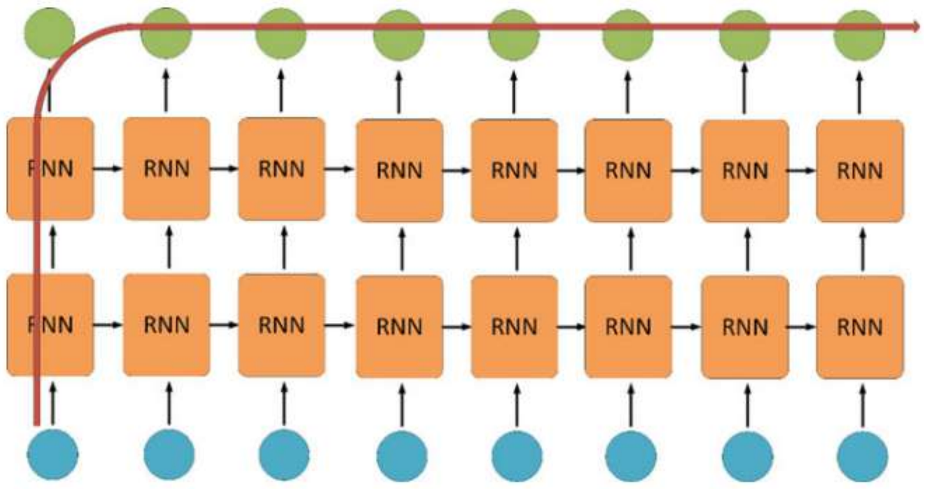

Figure 2.19: Caption

  - **Long-short term memory LSTMs** An LSTM is a variation of the traditional RNN model. it was proposed to solve some RNN's limitations like the vanishing gradients problem and for handling larger data. It is composed of a cell which consists of an input gate, and a forget gate [38].

- **Transformers** Transformers were first proposed by [39]. The former treat sequential data in parallel based on the multi-head attention mechanism. Each attention head performs a different attentions on the input sequence. A transformer consists of an encoder and a decoder as figure 3.15 shows. Transformer outperforms the traditional recurrent models like RNNs because it work faster (parallel architecture), and treat all the sequence parts at the same time (simultaneously observe the input sequence) because of the attention mechanism [39].

2.4 Conclusion

In this chapter we presented the basics of big data and machine learning. First, we defined the term big data and demonstrated its characteristics (v's). Next, we presented the data types that are used in big data, followed by presenting the big data architecture components. Also, big data analytics was defined and its different analysis techniques were provided. Then, a quick overview on big data tools that are used in the industry like hadoop ecosystem was provided. After that, we introduced the basics of machine learning ML. We provided some definitions and presented the learning paradigms used during the learning process. We also presented and explained some famous algorithms that are used in this domain area like support vector machine SVM. We particularly focused on deep learning DL. Therefore, we illustrated some of the ML challenges that motivates DL. Also, we provided DL definition as well as explained its learning process which is based on two passes (forward and backward propagation). And we conclude the chapter by introducing the main DL architectures that are proposed in the literature.

Chapter 3

Contributions

Contents

3.1 Comparative study for dimensionality reduction techniques for big data

3.1.1 Introduction

Big data has been recently considered as a buzzword, especially with the large production of information in an exponential manner [1, 2]. In this regard, studies expect that, in 2020, the amount of information produced would exceed 35 ZB(zettabyte) [40, 41]. Therefore, the use of traditional theories and technology with big data has become even more difficult [1, 40, 19]. For this reason, seeking new ways to deal with big data has become essential. It is essential to emphasize that among the problems that face big data is high dimensionality [20]. The latter makes the processes of analyzing big data and extracting a value from it (hidden patterns and insights) a difficult task. This paper, therefore, will serve to introduce some of the recent solutions that can be used to reduce high dimensionality in big data. The outlines of this paper organized as follows: section 3.1.2, introduces the term big data as well as its V's (characteristics), the main challenges facing each V, and the dimensionality reduction definition. Section 3.1.3, presents some of the recent solutions that were proposed to solve the high dimensionality problem for big data. After that, a comparative table is presented in section 3.1.4, that describes the previous techniques and the data-sets used to test the performance of each one. Finally, a conclusion and future work are presented in section 3.1.5.

3.1.2 BACK GROUND

We present in this section some definitions related to big data which includes the definition, its characteristics, the problem in hand and its solution which is dimensionality reduction.

3.1.2.1 BIG DATA

A lot of people think that big data is a large data-set that characterized only by high volume. But the reality is that volume is just one of its characteristics. For example [42] defines it as, "big data is high-volume, high-velocity, high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation".

In addition, Tech-America Foundation asserts big data as, large volume with high variety and high velocity that requires advanced technologies to be stored, captured, managed, and analyzed [19]. Similarly, [43] defines it as the data-sets that require a lot of time to be captured, processed, and managed using the traditional information technologies. In this respect, others define big data regardless of its volume, as data-sets that have complex structure and need complex operations to be processed [1]. For example, a graph of several terabytes may be considered as big data while a simple data-set with several petabytes may not.

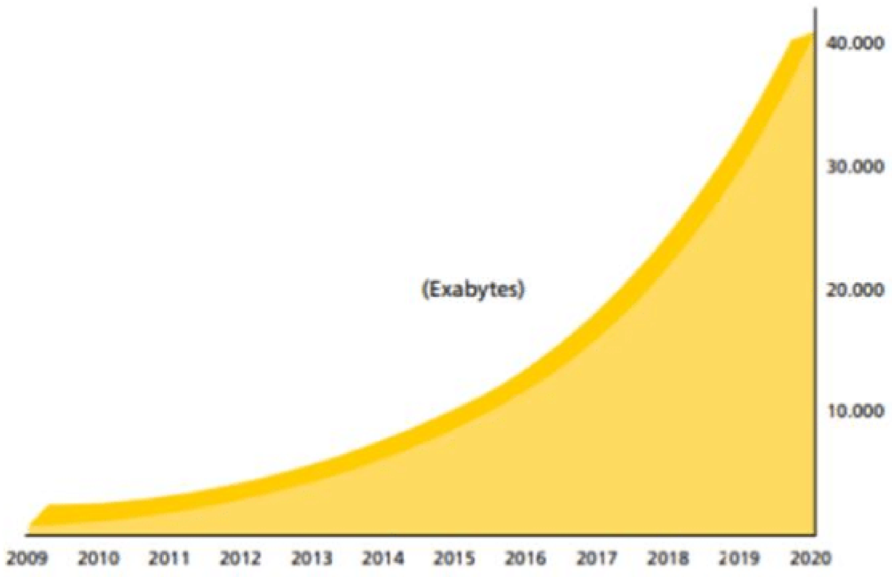

Figure 3.1: Data growth from the beginning of 2010 to the end of 2020 [41]

3.1.2.2 DIMENSIONALITY REDUCTION

Rather than dealing with a significant number of dimensions in which high computation is needed or some dimensions are low in value or insignificant, it is preferable to deal only with few and essential dimensions. For that reason, dimensionality reduction is defined as: "Dimensionality reduction is an effective solution through finding meaningful low-dimensional structures hidden in their high-dimensional observations" [20].

3.1.3 RELATED WORK

In this section, we present some of the recent dimensionality reduction techniques proposed for big data.

3.1.3.1 MODIFIED PRINCIPAL COMPONENT ANALYSIS

In this sub-section, we introduce the classical and the modified (PCA) as well as the motivation behind the modified (PCA). The two algorithms are used to reduce dimensions by extracting the principal components.

- **CLASSICAL (PCA)** PCA is a statistical algorithm that explains a large number of correlated explanatory variables by a small number of uncorrelated components using a linear combination of the explanatory variables for each component [44]. The principal components are a result of singular value decomposition (SVD) of the explanatory variables like the following [44]:
  Let X be the matrix of $n*p$ of explanatory variables and X_S its standardization.
  Let the (SVD) of X_S is: $SVD = UDV'$ where:
  U is an $n*p$ orthogonal matrix satisfying: $U'U = I_P, V$ is an $p*p$ orthogonal matrix satisfying: $V'V = I_P, D$ is a $P*P$ diagonal matrix for singular values.
  The columns of UD are the principles components.

- **MODIFIED (PCA)** The classical PCA needs to load all the data on memory to be applied. Thus, when it comes to big data, memory barriers pop up. For that reason, [44] proposed a modified PCA based on scanning data by rows to overcome the memory barriers. *Scanning data by rows* loads individual rows to the memory and provides a summary information. They provided two distributions:
  - **Single processor**. if the data size does not exceeds the hard disk memory size, an algorithm of modified (SVD) of two rounds is proposed:
    - **First round** Computes D_x, V_k of the optimal K for every $K \leq P$ (using scanning data by rows).
    - **Second round** uses the previous D_k, V_k of first round with scanning data by rows to compute U_k and get the principal components.
  - **Parallel distribution** if the data size exceeds the hard disk memory size, a distributed (SVD) algorithm based on map reduce is proposed:
    - **First round**
      - Computes in parallel (using scanning data by rows and mapReduce) D_k, V_k of the optimal K for every $K \leq P$.
      - Computes the total (SVD) modified using the Reduce function.

 * **Second round** scans data by rows for each processor and uses the results (D_k, V_k) of the first round, to compute U_k and the principal components using map and reduce.

3.1.3.2 SIMPLICIAL NON-NEGATIVE MATRIX TRI FACTORIZATION (SNMTF)

In this sub-section, we present the classical (NMF) which used to reduce dimensions by factorize a matrix into a product of smaller matrices. Also, we present a new proposed (SNMTF) which is based on the classical (NMF) to reduce dimensionality with some modification.

- **(NMF)** Non negative matrix factorization(NMF) models are linear models for dimensionality reduction by transforming data into low dimensions of latent components, in which a given matrix V is factorized into a product of two matrices W^T and F with no negative values for all matrices [45].

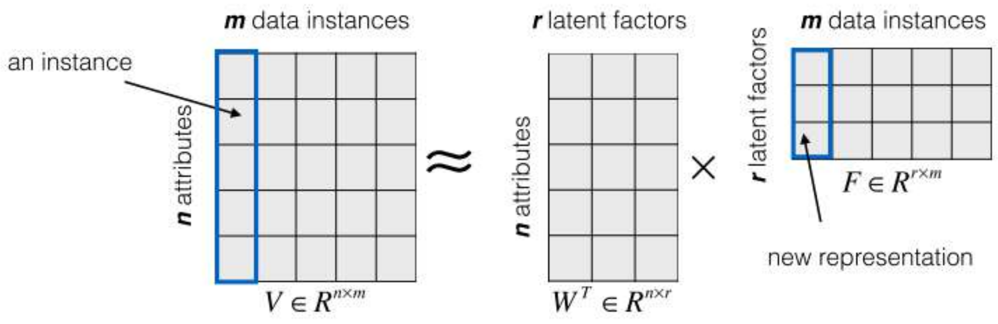

Figure 3.2: Nonnegative matrix factorization [45]

- **NEW (SNMTF) MODEL** In [45], the authors based on the classical (NMF) and because it does not explain consistently the roles of latent components, they propose a new (SNMTF) model which factorizes a given matrix V into product of three matrices W^TDF, where D is positive diagonal matrix, $\sum_{k=1}^{r} W_k i = 1 \forall i$, $\sum_{k=1}^{r} F_k j = 1 \forall j$. Scaling the two factor matrices F and W via the diagonal matrix D, can lead to inconsistency of interpretability.

  To solve this problem, they add an assumption which equals all the weights in the diagonal matrix $\lambda_1 = ...\lambda_r = \lambda$ [45]. And for decreasing the iteration complexity, attain a linear convergence and easily control sparsity, they proposed a combined algorithm of three-block alternating direction (used to decompose the product W^TDF into three blocks W, D, F) and Frank-Wolfe's scheme (used to solve the sub-problems in each block W, F). D solved using the previous assumption. The new proposed model (SNMTF) outperformed

(PCA) and the existing dimensionality reduction of (NMF) in the literature [45].

3.1.3.3 STACKED AUTOENCODERS

Stacked autoencoders are a class of neural network (NN). They are identified by the presence of the same number of neurons in the input and output layers and a lower number of neurons in the hidden layers (symmetric architecture).

The dimensions of the data are decreased as they pass through the hidden layers, [46] presented and compared the performance of stacked autoencoders which are powerful dimensionality reduction technique for linear and/or non-linear problems, versus classical (PCA) that we previously presented. The authors found that autoencoder has reduced the reconstruction error by ten times than (PCA). Actually, an autoencoder with just one hidden layer outperformed (PCA) in reducing data. The following figure shows a general architecture of a stacked autoencoder as presented in [46].

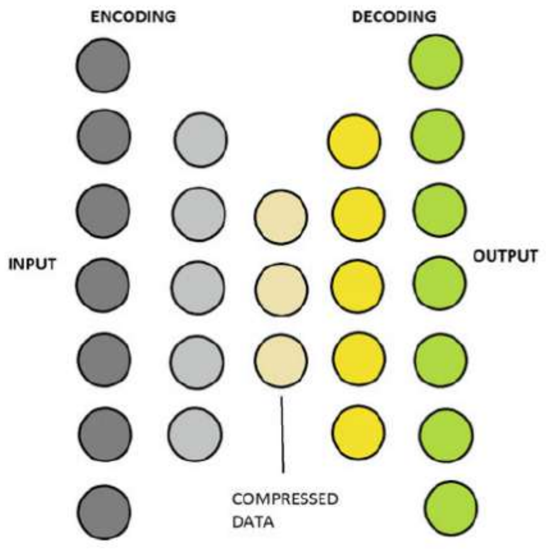

Figure 3.3: General architecture of stacked autoencoder (AE) [46].

3.1.3.4 LINGUISTIC HEDGES NEURO-FUZZY CLASSIFIER WITH FEATURE SELECTION (LHNFCSF)

In [47], the authors presented a combination of neural networks and fuzzy inference systems (neuro-fuzzy) based on linguistic hedges with feature selection algorithm, for both dimensionality reduction and classification. The motivations behind this combination were first neural networks are strong in learning and classification, while fuzzy inference systems are strong with issues of reasoning with high semantic level. The second motivation was to overcome the drawbacks of both neural networks like black box behavior, and fuzzy inference systems like selecting suitable membership values.

In fact, they compared the performance of two models of neuro-fuzzy systems. The first model was the combination without feature selection LHNFC, and the second model (LHNFCSF) was the combination with feature selection. The selected features in the second model were the features that have the biggest hedge value for any class or the biggest hedge value for each class.

(LHNFCSF) has reduced the dimensions approximately to the half for all cases without losing accuracy for classification during training or testing comparing with (LHNFC).

The following figure shows the general architecture of neuro-fuzzy systems with (LHs).

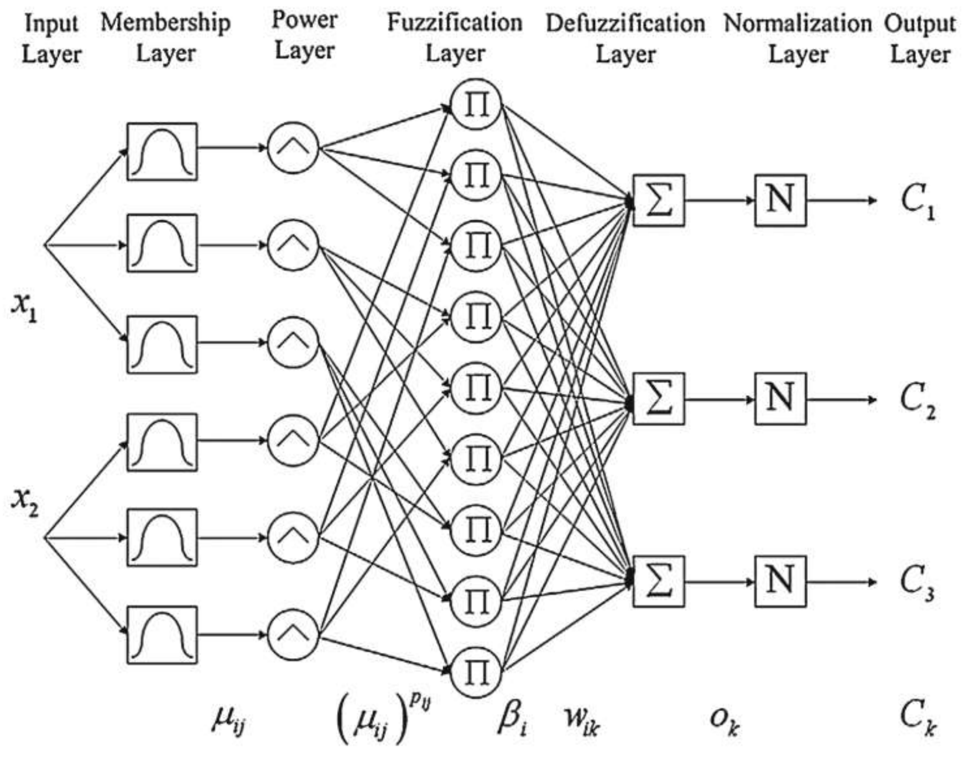

Figure 3.4: General architecture of neuro-fuzzy classifier with (LHs) [47].

- $x1, x2$: inputs.

- U_ij: the membership grade of ith rule and jth feature. Calculated by the Gaussian function [47].
- $\alpha_{ij} = (U_{ij})^{p_{ij}}$: modified membership grade based on (LHs) of each fuzzy set, where p_{ij} denotes the (LH) value of the ith rule and jth feature [47].
- B_i: degree of fulfillment (firing strength) of the rule i for D number of features: $\Pi_{j=1}^{D}\alpha_{ij}$.
- O_k: output weights which is calculated by: $\sum_{i=1}^{U} B_i w_{ik}$, where w_{ik} represents the degree of belonging to the kth class that is controlled with the ith rule, and normalized if the summation > 1 [47].
- C_k: the existing classes.

The algorithm used to reduce the dimension (feature selection) is the following [47].
First, the equation used for the selection value of the jth feature is:

1. Describe only one fuzzy rule for each class using Gaussian distribution.
2. Set $P_{ij} = 0.5$, for $i = 1,2,...,K$ and $j = 1,2,...,D$, where K is the number of classes and D is the number of features.
3. Set the number of selected features (L).
4. Train the neuro-fuzzy classifier with LHs. In training $0 \leq P_{ij} \leq 1$.
5. For $i = 1$ to K. Find the jth feature that satisfies the maximum p value for the ith class. Take the jth feature into the individual discriminative features set.
6. The $(L - K)$ features, which have the biggest P value, are selected as common discriminative features.
7. There are L discriminative features. The new training X new and testing data are created by the selected features from the original data (feature dimensions are reduced from the original number to L).

3.1.3.5 DEEP BELIEF NETWORK (DBN)

(DBNs) are a class of neural networks. A (DBN) consists of multiple hidden layers where each layer is an (RBM) (Restricted Boltzman Machine), which is also a class of neural networks. Each (RBM) is connected with two layers, a hidden layer and a visible layer, and so on [48].

In [48], the authors proposed a new framework which consists of two (DBNs). The first (DBN) used to reduce the dimension of spectral bands on all pixels of the input (reduce the number of features) while the second (DBN) used as feature extractor (extract spectral-spatial features) and classifier at the same time because of its last layer which is a discriminative type.

(NNs) are strong models dealing with labeled or unlabeled data. The proposed model outperformed (PCA), and the results showed that it gave better class sparsity, and thus, better and more stable classification accuracy [48].

The following figure shows the general architecture of a (DBN), which is consist of 3 stacked (RBMs). In the first (RBM) from Figure 3.5, visible layer is x and hidden layer is (h1). For the second (RBM), hidden layer in the first (RBM) be the visible layer and so on.

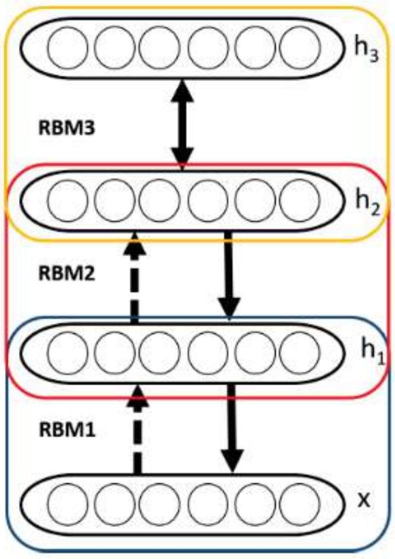

Figure 3.5: General architecture of (DBN) with 3 hidden Restricted Boltzman Machines (RBM1), (RBM2) and (RBM3) [48].

3.1.4 DISCUSSION

In this section, we introduce a comparative table that explains some details for each algorithm, as well as, some figures that explain the obtained results.

It is apparent that 60% of the presented studies used images as case of study while 40% of them used different types of text files as Figure 3.6 shows.

Based on the results obtained in [45, 46, 48], we present the following figures that demonstrate the performance of the (SNMTF), (DBN), and the stacked autoencoder (AE) respectively compared with (PCA).

Tables 3.1 and 3.2 presents some details, as well as the advantages and disadvantages for each algorithm.

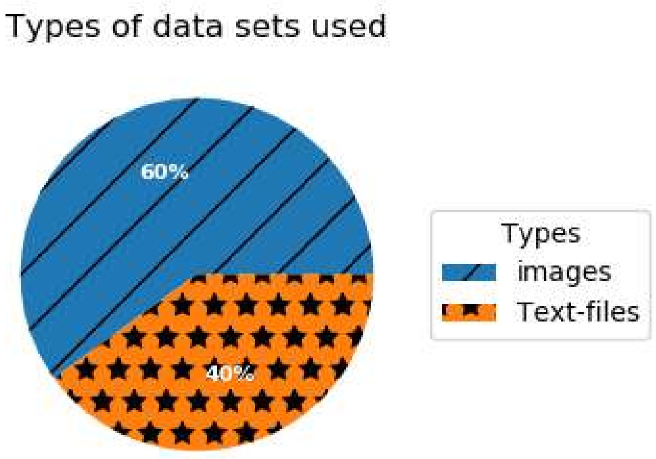

Figure 3.6: Types of data-sets used for each study.

Figure 3.7 shows that (SNMTF) outperformed (PCA) and gave less inaccuracy classification for all the data-sets used (Faces, digits and Tiny-images).

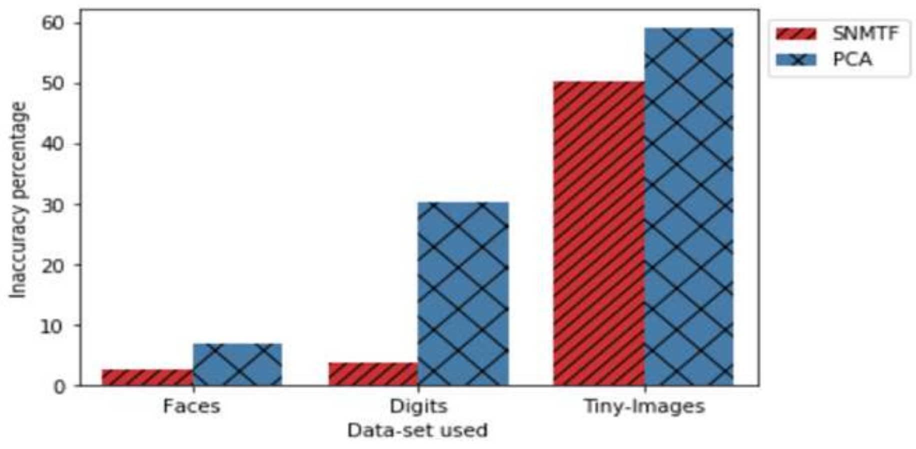

Figure 3.7: Inaccuracy percentage of classification for different data-sets using (SNMTF), (PCA) as dimensionality reduction.

Figure 3.8 demonstrates that (DBN) gives better classification accuracy (overall accuracy (OA), average accuracy (AA)) and kappa statistic (KS), for all experiments than (PCA).

In all cases and for the same number of components, the autoencoder performs better than PCA and produce less reconstruction error [46].

	PCA modified [44]	Neuro-fuzzy [47]	Stacked autoencoder [46]	SNMTF [45]	DBN [48]
Model	Single or parallel distribution	NNs + Fuzzy inference system	NNs	Modified NMF	NNs
Metrics	Means square error (MSE)	Accuracy	Reconstruction error, (MSE)	Sparsity, Inaccuracy	Accuracy kappa statistics (2 types), value
Method based	Scan data by rows and map-reduce for parallel distribution	Linguistic hedges with feature selection	/	modified NMF	/
Data sets	3 synthesized data-sets [44] +1999 KDD of California university [51]	Breast cancer diagnostic+ prognosis+ Erythemato squamous+ Thyroid disease [51]	5 synthesized data-sets (Gaussian distribution)[46] +ozone level detection + gas drift concentration [51]	Faces [49]+ digits [29]+ Tiny images [50]	Indian pines hyperspectral images [48]
Results	Reduced dimensions more than half with minor error	100% train accuracy 97% test accuracy	Reduced reconstruction error 10 times than PCA	Less inaccuracy classification for all data-sets compared to PCA	Outperformed PCA
Advantages	Handle all data sizes even if it exceeds the memory	The combination overcomes drawbacks of NNs and fuzzy inferences systems	Independent of linearity /non linearity, labeled/ unlabeled data	Low complexity, controlled sparsity	Supports any type of data

Table 3.1: Comparative table of the presented techniques used for dimensionality reduction for big data PART.1

Disadvantages	Non linearity difficulties	Training takes, time, extensive exploration to find the right combination	Training takes time, extensive exploration to find the right combination	Lot of constraints to overcome the model's problems and make the model works perfectly	Training takes time, extensive exploration to find the right combination
Classifier	Logistic, linear, log-linear regression	LHNFCSF	/	Gradient boosting classifiers	DBN

Table 3.2: Comparative table of the presented techniques used for dimensionality reduction for big data PART.2

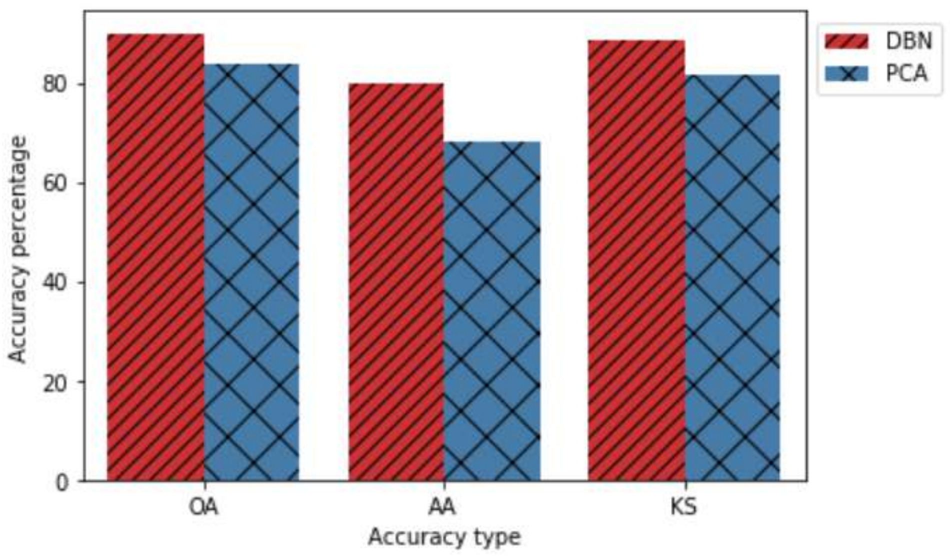

Figure 3.8: Different types of classification accuracy using DBN, PCA as dimensionality reduction.

3.1.5 Conclusion

In this thesis, we introduced the big data term with its multiple definitions from various sources, as well as its characteristics (Vs). Furthermore, we briefly discussed the major issues for each characteristic. This paper highlights some of the available techniques for dimensionality reduction for big data. All the presented techniques have a good performance for reducing dimensions of a given data-set; however, the most interesting class of them is (NNs). (NNs) are considered strong techniques for dimensionality reduction because, they yield the advantage of dealing with all types of data. In this respect, our future work will be specialized in studying this class and may propose a new model for dimensionality reduction using (NNs).

3.2 On the effectiveness of Dimensionality Reduction Techniques on High Dimensionality Datasets

3.2.1 Introduction

Breast Cancer (BC) is one of the most serious illnesses that affects women. It mostly affects the cells that line the ducts or lobules in the glandular tissue of the breast [8]. As mentioned by [8], The most prevalent cancer worldwide at the end of 2020 is (BC). A total of 7.8 million living women have been diagnosed with (BC) in the last six years. Consequently, it is needed to incorporate innovations in artificial intelligence (AI), such as machine learning (ML) for a deeper knowledge of the illness and early detection. In this regard, a series of measures and therapies can be used to reduce mortality and loss of life. As a result, several researches for (BC) detection were carried out, including as [9, 10, 11, 12] and [13]. However, using (ML) algorithms on high-dimensional datasets may result in excessive complexity. Thus, The purpose of this work is to present a realistic comparison of five well-known dimensionality reduction algorithms (DRTs) and investigate their performance using Support Vector Machine (SVM) and Multi Layer Perceptron (MLP), which are two well-known classifiers in the field.
the rest of the sections is organized as follows: Section 3.2.2 initially outlines the major contributions for (BC) classification and, then, describes the basic techniques and functions utilized in our studies. Section 3.2.3 shows and assesses the dataset utilized in our research. Section 3.2.4 assesses the performance of the given (DRTs) when used with (SVM) and (MLP) classifiers. Section 3.2.5 aims at reporting the findings and drawing conclusions.

3.2.2 Materials and Method

3.2.2.1 Related Works

Several techniques have been proposed for (BC) diagnosis. [52] and [11] are some examples. They concentrated on the K-nearest neighbor (KNN) algorithm for classification of (BC). In particular in [11], for resolving the problem, the researchers presented a new hybrid solution based on a Fuzzy-artificial immune system and the (KNN) algorithm. The presented technique had a classification accuracy of 99.14%. [11].
Other research focused on labeling (BC) using the support vector machine (SVM) technique. In [53], the researchers employed Linear Support Vector Machine (L-SVM) as a reference algorithm and compared it to numerous machine learning (ML) techniques used for the same purpose., i.e (BC) classification. The given (L-SVM) achieved 96% of accuracy.

In [54], a mixture of k-means and SVM techniques was proposed named (K-SVM). For each class, K-means is utilized to identify hidden patterns while SVM is employed as a (BC) tumor classifier. The latter uses the dataset's characteristics alongside with features derived by K-means algorithm to classify the tumor.
Others concentrated on ensemble algorithms for constructing a model made up of numerous (ML) models. In [13], The authors proposed a novel architecture made up of numerous (SVM) model structures. To reap the benefits of each individual (SVM) model, they were all hybridized together. The suggested algorithm integrated two forms of (SVM) called C-SVM and V-SVM, as well as six different types of kernel functions.
In [55], The researchers proposed including a voting algorithm between several (ML) classifiers into a single model for (BC) classification. The voting algorithm picks the best model combination for maximum classification accuracy.
In [56],The authors developed a novel architecture comprised of a Genetic Algorithm (GA) for extracting the most useful and relevant characteristics and a Rotation Forest (RF) for classification. The findings showed that the suggested algorithm achieved good classification accuracy.
Other works, such as [57, 10, 52] stressed the use of deep learning models (DNN) to tackle the problem of (BC) classification. For example, [58] introduced a Deep Belief Network-based (DBN-NN) architecture for (BC) classification. The presented architecture consists of two stages. The first one is training the (DBN) unsupervisedly while the second is training supervisedly another (DBN),i.e. the same design, but with the first stage's weights as input.
In [59], The authors presented a Neural network (NN) Self-Validation Cerebellar Model Articulation Controller (SVCMAC). In reality, SVCMAC is a hybrid of the Cerebellar Model Articulation Controller (CMAC) and the Self-Validation Algorithm. The latter is utilized to select the model's best parameters.
In [12], The authors presented a novel design for neural networks (NNs) called the Genetically Optimized Neural Network Model (GONN). The researchers enhanced Genetic programming by incorporating novel crossover and mutation algorithms in order to get an optimum (GONN) architecture. The proposed algorithm significantly improved classification accuracy. (GONN) achieved 98.24% by dividing the data into 50% training and 50% test and validation.

3.2.2.2 Data Normalization

For the goal of normalizing the dataset, we used the Z-score normalization algorithm in this study. The Z-score can be described as the difference between the raw data x_f and the feature's mean value μ, divided by the original feature's standard deviation. σ [53]. The normalization method employed is described in the following equation:

$$x_{new} = \frac{x_f - \mu_f}{\sigma_f} \tag{3.1}$$

3.2.2.3 Dimensionality Reduction algorithms

In this section, we provide a brief description of the used algorithms in our experiments for reducing high dimensionality.

3.2.2.3.1 Auto Encoders (AE) Stacked autoencoders belong to the neural network (NN) class. They distinguish themselves from one another by having an identical number of neurons in the layers of input and output and a smaller number of neurons in the hidden layers (symmetric architecture). The dimensions of the data are lowered as they transit through the concealed layers. [46].

3.2.2.3.2 T-Distributed Stochastic Neighbor Embedding (T-SNE) [60] is the first to propose T-Distributed Stochastic Neighbor Embedding (T-SNE). Actually, (T-SNE) uses basic gradients with a symmetrized (SNE) cost function. (T-SNE) computes similarity using a Student-t distribution rather than a Gaussian distribution.

3.2.2.3.3 Recursive Feature Elimination (RFE) (RFE) is a dimensionality reduction algorithm that operates by eliminating the least significant features in a subset of features recursively until the required number of features k, i.e. the most significant k features, is attained [61].

3.2.2.3.4 Isometric Feature Mapping (Isomap) The researchers present the Isometric Feature Mapping (Isomap) algorithm for decreasing high dimensionality in [62]. The algorithm given consists of three steps: (1) determines the neighbors points using the distance between the points; (2) (Isomap) aims to give an approximation of the geodesic distances between the points by computing the shortest paths distances between the points; and (3) (MDS) algorithm is used to construct d-dimensional embeddings.

3.2.2.3.5 Principal Component Analysis (PCA) (PCA) is an algorithm for unsupervised machine learning. [63] introduced it for the first time. It has been widely employed in a variety of sectors, including engineering science, biology, and physics, to name a few. The (PCA) algorithm's main purpose is to reduce the dimensionality of datasets by turning correlated characteristics into a set of uncorrelated principal components. This algorithm consists of four major steps:

1. Covariance matrix calculation.
2. Obtaining the eigenvalues and eigenvectors of the covariance matrix.
3. Sorting eigenvalues and their related eigenvectors.
4. obtaining the top k principal components.

3.2.3 Dataset Evaluation

We demonstrated our studies using WDBC dataset (diagnostic) from the (UCI) machine learning repository [30]. The dataset consists of 596 cases separated into 2 categories (37.3 percent malignant, 62.7 percent benign). Each raw data set is made up of thirty two characteristics, as defined in the (UCI) repository and illustrated in Table 3.3. Please see [30] for further details.

id	diagnosis	radius-mean	texture-mean	perimeter-mean
842302	M	17.99	10.38	122.8
area-mean	smoothness-mean	compactness-mean	concavity-mean	concave points-mean
1001	0.1184	0.2776	0.3001	0.1471
symmetry-mean	fractal-dimension-mean	radius-se	texture-se	perimeter-se
0.2419	0.07871	1.095	0.9053	8.589
area-se	smoothness-se	compactness-se	concavity-se	concave points-se
153.4	0.006399	0.04904	0.05373	0.01587
symmetry-se	fractal-dimension-se	radius-worst	texture-worst	perimeter-worst
0.03003	0.006193	25.38	17.33	184.6
area-worst	smoothness-worst	compactness-worst	concavity-worst	concave points-worst
2019	0.1622	0.6656	0.7119	0.2654
symmetry-worst	fractal-dimension-worst			
0.4601	0.1189			

Table 3.3: A raw data sample with thirty-two characteristics

3.2.4 Result and Discussion

For (BC) classification, we employed two well-known models from the literature To evaluate the efficiency of the five presented dimensionality reduction strategies, namely (T-SNE, RFE, Isomap, PCA, and AE). These are the models: Multi Layer Perceptron (MLP) and Support Vector Machine (SVM) [53]. Each classification model was used alongside with each dimensionality reduction algorithm, and many tests were carried out.

As a consequence, we split our data in half for training and half for validation and testing.

Both the normalization and (DRTs) models were solely trained on training data to prevent information leakage and to assess the models' generalization power. The trained models were then utilized to transform the testing data.

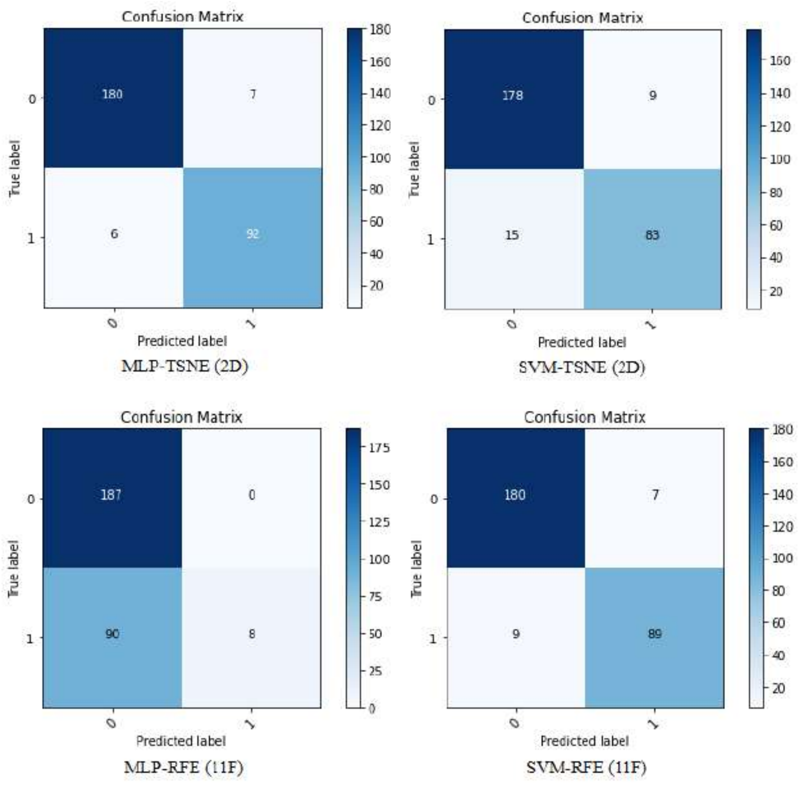

Figure 3.9: MLP and SVM confusion matrices using RFE and T-SNE dimensionality reduction algorithms.

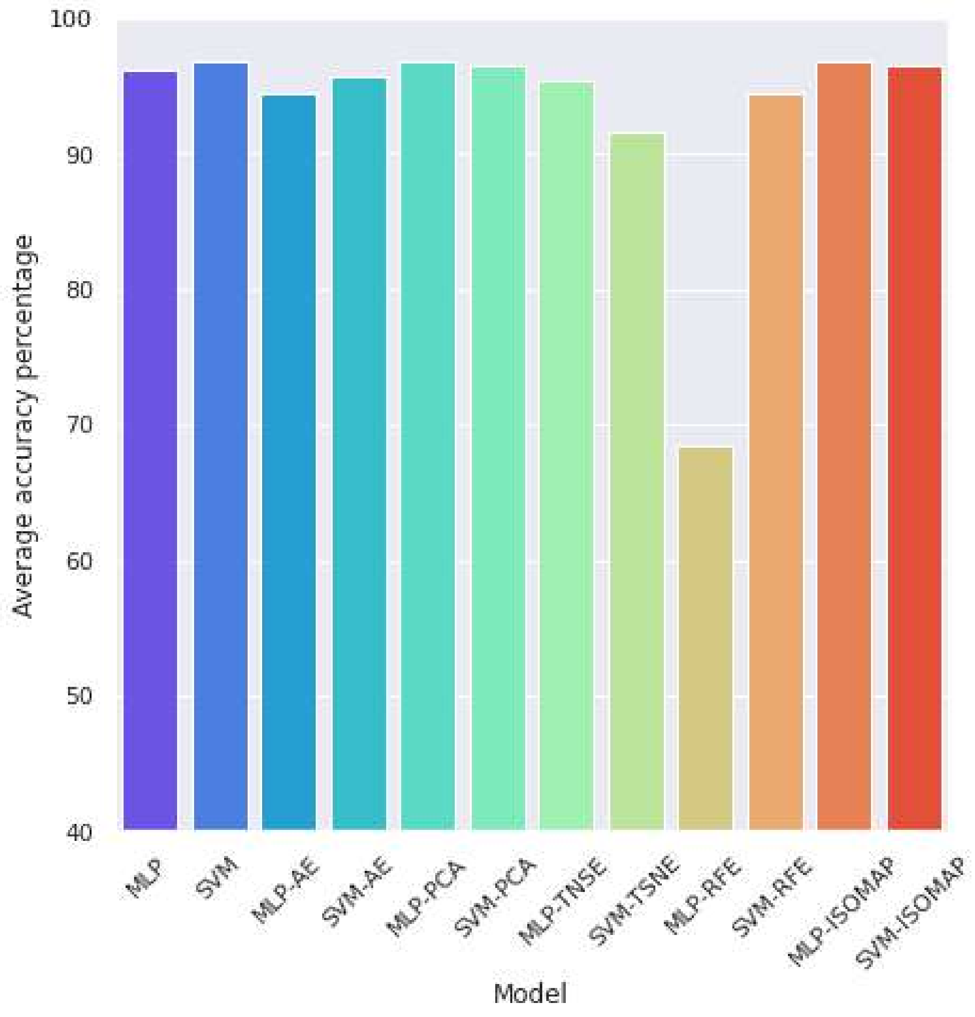

Figure 3.10: All 12 models' average accuracy percentage.

We used confusion matrices to examine the effectiveness of all the given models and to identify the true and false predicted classes, as shown in Figures 3.9 and 3.14. It is crucial to remember that a correct forecast is a true forecast, whether it is positive or negative. Figures 3.9 and 3.14 show that without using dimensionality reduction algorithms (DRTs),(SVM) outperformed (MLP) by making two true predictions out of a total of 285 predictions for (SVM). Using the five given (DRTs) improved correct predictions for almost every (MLP) scenarios by at least two predictions. Only when (RFE) and (AE) were used as (DRTs) was there a drop of at least 5 correct predictions in comparison to basic (MLP). The most reliable prediction results were obtained by (MLP-Isomap) and (MLP-PCA) (276 correct predictions and just 9 wrong predictions, a gain of 2 correct predictions over simple (MLP)). (MLP-TSNE) had two less correct predictions than simple (MLP). Nonetheless, combining the given (DRTs) algorithms with (SVM) showed only a reduce of correct predictions by at least 1 prediction in comparison to simple (SVM).

	MLP	MLP-AE	MLP-PCA	MLP-Isomap	MLP-TSNE	MLP-RFE
MLP	///	**+1.7**	-0.7	-0.7	**+0.7**	**+27**
MLP-AE	-1.7	///	-2.4	-2.4	-1	**+25**
MLP-PCA	**+0.7**	**+2.4**	///	0	**+1.4**	**+28**
MLP-Isomap	**+0.7**	**+2.4**	0	///	**+1.4**	**+28**
MLP-TSNE	-0.7	**+1**	-1.4	-1.4	///	**+27**
MLP-RFE	-27	-25	-28	-28	-27	///
SVM	**+0.7**	**+2.4**	0	0	**+1.4**	**+28**
SVM-AE	-0.3	**+1.4**	-1	-1	**+0.3**	**+27**
SVM-PCA	**+0.3**	**+2.1**	-0.3	-0.3	**+1**	**+28**
SVM-Isomap	**+0.3**	**+2.1**	-0.3	-0.3	**+1**	**+28**
SVM-TSNE	-4.5	-2.8	-5.2	-5.2	-3.8	**+23**
SVM-RFE	-1.7	0	-2.4	-2.4	-1	**+25**
//	//	//	//	//	//	//
	SVM	SVM-AE	SVM-PCA	SVM-Isomap	SVM-TSNE	SVM-RFE
SVM	///	**+1**	**+0.3**	**+0.3**	**+5.2**	**+2.4**
SVM-AE	-1	///	-0.7	-0.7	**+4.2**	**+1.4**
SVM-PCA	-0.3	**+0.7**	///	0	**+4.9**	**+2.1**
SVM-Isomap	-0.3	**+0.7**	0	///	**+4.9**	**+2.1**
SVM-TSNE	-5.2	-4.2	-4.9	-4.9	///	-2.8
SVM-RFE	-2.4	-1.4	-2.1	-2.1	**+2.8**	///

Table 3.4: Differences in average accuracy between all the 12 presented models

in 3.10 and Table 3.4, we examined the average accuracy percentage of all the twelve given models (simple (MLP), simple (SVM), and ten classifiers combined with (DRTs) with eleven decreased features as input). We found that (MLP-Isomap), (MLP-PCA), and basic (SVM) had the best accuracy by at least 0.3% of prediction accuracy, followed by (MLP-TSNE), and finally (MLP-AE). (MLP-RFE) performed the poorest after (MLP-AE). The former achieved 68% prediction accuracy with a difference of $\simeq$ 25% in comparison to the model that gave the the least accurate forecast.
We observe that combining (SVM) with the five given (DRTs) failed to improve overall accuracy. In comparison to the simple (SVM), the combinations (SVM-PCA) and (SVM-Isomap) with 11 features as input have marginally lower prediction

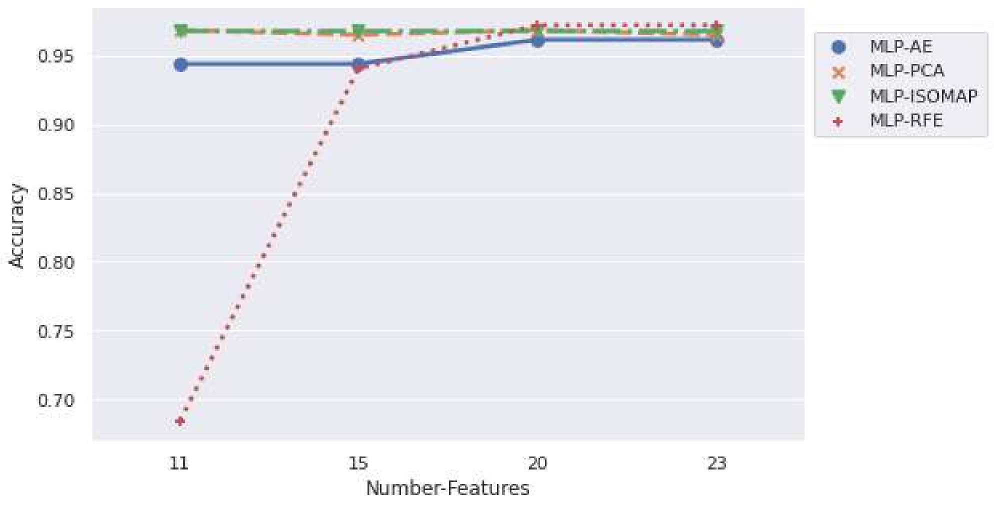

Figure 3.11: MLP's average accuracy with the four DRTs and distinct number of features.

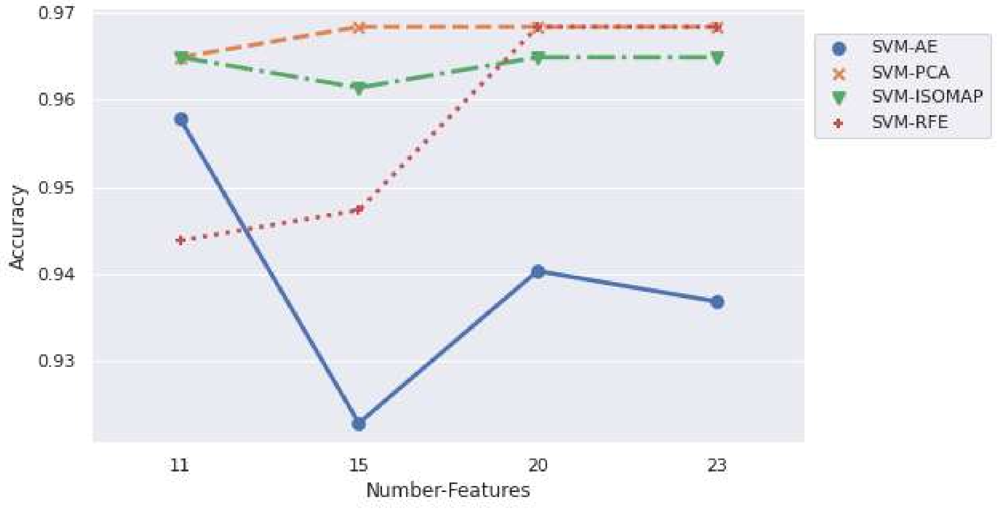

Figure 3.12: SVM's average accuracy with four DRTs and varied number features.

accuracy by 0.3%.
In general, among the other models, the combinations (MLP-PCA) and (MLP-Isomap) provided the best accuracies.

We further tested the performance of the models with varying amounts of features, as demonstrated in 3.11, 3.12, and 3.13 show.
Figure 3.11 shows a consistent correlation, indicating that increasing the number of features has no effect on average accuracy for both (MLP-Isomap) and (MLP-PCA).

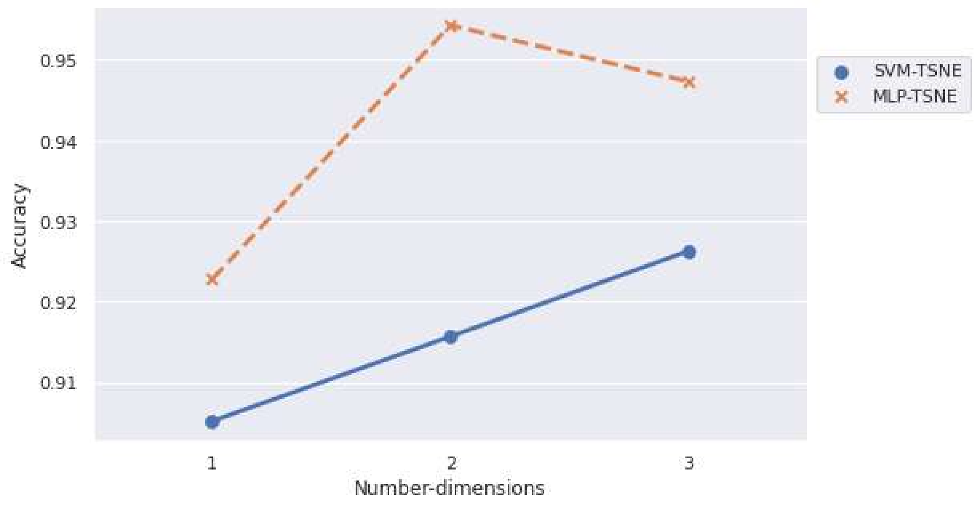

Figure 3.13: The average accuracy of SVM, and MLP classifiers using the T-SNE algorithm with various dimensions

(MLP-RFE) and (MLP-AE) revealed a positive correlation between the number of features and the average accuracy, implying that increasing the number of features enhances the average prediction accuracy.

Figure 3.12 shows a consistent correlation between the number of features and average accuracy for both (SVM-Isomap) and (SVM-PCA). It further demonstrates a positive correlation between the number of features and the average accuracy for (SVM-RFE). Nonetheless, (SVM-AE) demonstrated an inconsistent correlation with the highest average accuracy obtained using 11 as a number of features.

Figure 3.13 shows that utilizing 2 as a number of dimensions resulted in the best average accuracy for (MLP-TSNE), whereas utilizing 1 or 3 resulted in a drop. It further shows a positive correlation between the number of dimensions and (SVM-TSNE) average accuracy. (MLP-TSNE) outperformed (SVM-TSNE) in all dimensions and gave higher average accuracy.

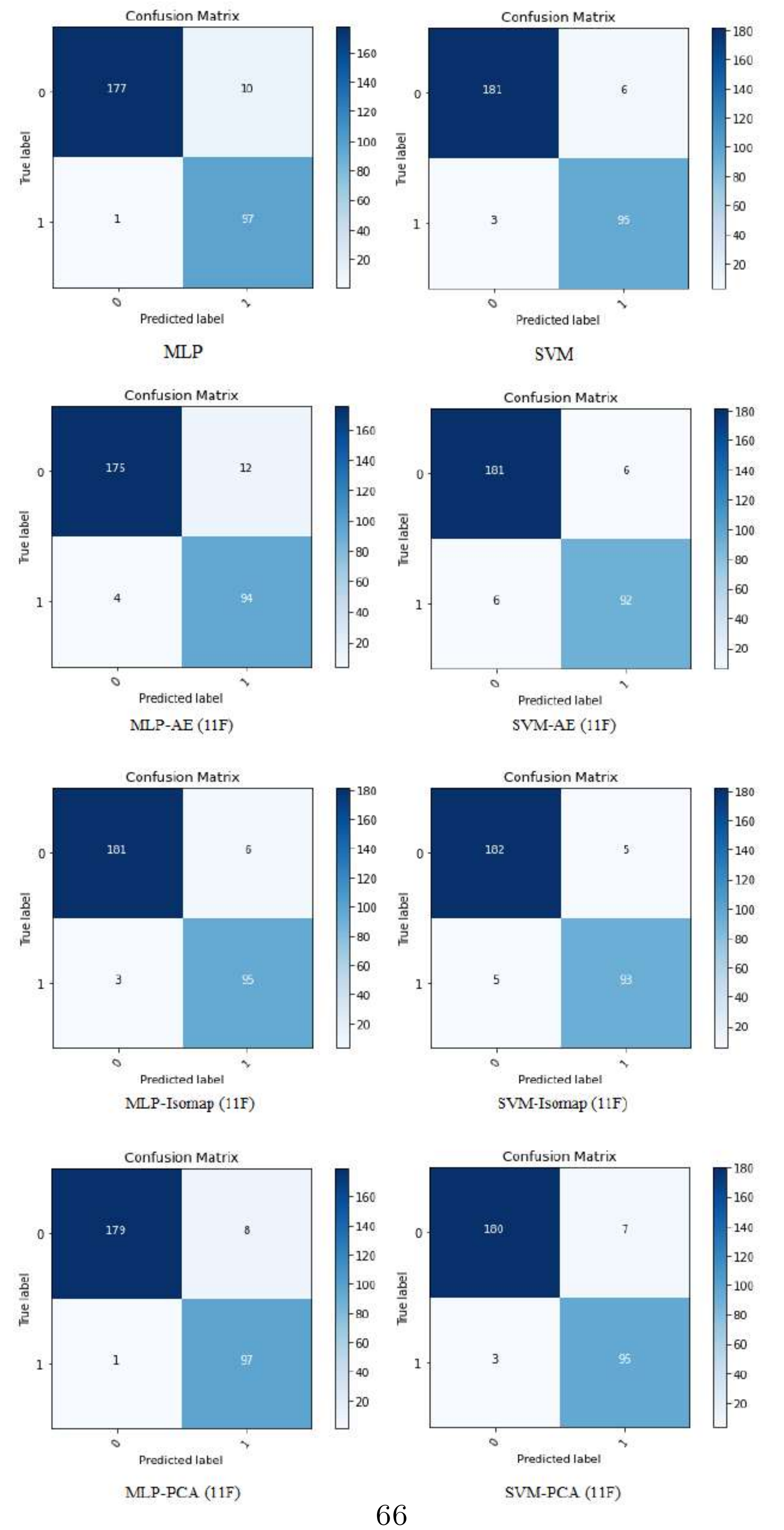

Figure 3.14: Confusion matrices for SVM, and MLP with no reducing dimensionality, and with dimensionality reduction algorithms AE, Isomap, and PCA

3.2.5 Conclusion

Machine Learning (ML) models have proven indispensable in a variety of fields, including medicine. The former is utilized to create precise models that aid in prediction and classification problems. Developing (ML) models, on the other hand, may result in high-complexity models that require a huge amount of resources to execute. Dimensionality Reduction is therefore among the most popular strategies that aid in the development of (ML) models with low-cost. In this thesis, we emphasized the performance of 5 (DRTs) integrated with 2 popular (ML) models utilized for the classification of (BC). The findings support the use of (DRTs) for high dimensionality datasets in creating more accurate (ML) models and improving classification/prediction accuracy. Furthermore, they reveal that selecting the incorrect mix of classifiers and (DRTs) might result in poor outcomes and a reduction in classification/prediction accuracy.

3.3 Next location prediction using Transformers

3.3.1 Introduction

Predicting the mobility of users has become an essential task for a variety of location-based services in various domains including travel recommender systems [64, 65], urban management, transportation recommender systems [66, 67, 68, 69], and recommender systems based on point of interest [70].

Various methods and algorithms have been proposed in predicting human mobility. These strategies are often focused on studying the history of user movements, i.e. the sequence of past locations, as well as identifying repeated patterns that will be used to forecast future locations. In this respect, markovian models are widely used in predicting human mobility [71, 72, 73, 74, 75, 76, 77]. They predict the next location of a given user based on the current context which is the sequence of the K recent visited locations.

In light of the advancements in deep learning, a number of algorithms for predicting user mobility have been proposed [78, 79, 80, 65, 81]. These algorithms overcome some of the drawbacks of traditional mobility prediction models including Markovian models. The drawbacks of the latter, for instance, are as follows: (1) the difficulty of choosing the value of k a priori, (2) the impossibility of identifying patterns from parts of the context and relying only on the whole context to predict the next location. It is important to maintain that the main advantage of using deep learning models instead of Markovian models is that they can identify and extract hidden patterns. A hidden pattern P is a sequence of locations. Each location of a hidden pattern belongs to the context. For example, a hidden pattern P of length $= 2$ with context of length $= K$ is defined as: $P = (loc_i, loc_j)$ where $1 \leq i, j \leq K$ and $i \neq j$.

Deep learning models can perfectly represent any sequence and extract hidden patterns from the users historical sequences. For instance, [82, 78], and [79] proposed algorithms based on Recurrent Neural Network (RNN) and Long Short Term Memory Network (LSTM). In all the proposed algorithms, spatial and temporal information were extracted as features and used for next location predicting. In [80], the authors considered the next location prediction as a classification task. They proposed the use of a new embedding algorithm called loc2vec. The embedded sub-sequences will be transformed to RGB images. The latter, will be fed to a pretrained (CNN) model to classify the image, thereby predicting these next location.

One of the most recent and powerful models used in Natural Language Understanding (NLU) is Bert Transformer [83]. This model can learn bidirectional representations between parts, i.e. tokens, of any sequence. Many Bert extensions were proposed with different objective functions such as ROBERTA [84], UNILM

[85], ALBERT [86] , XLNET [87], and RomeBERT [88]. Transformers are better than (LSTMs) because they can model longer dependencies in parallel manner based on attention mechanism [39].

In our work, we have got the idea of leveraging the powerfulness of Transformers. Consequently, we propose a new model called WP-BERTA that is based on a combination of Bertwordpiece tokenizer [89, 83] and ROBERTA model [84]. Our proposed model has improved significantly the prediction accuracy by 3% compared to (CNN2D) [80], which is the model that gave the higher accuracy over all the tested models in our experiments (see Table 3.7).

The present paragraph describes the structure of the rest of the sections. In section 3.3.2, we introduce the main contributions for next location prediction, describe Bert model, and specify its extensions that are used for (NLP). In section 3.3.3, and as depicted in Figures 3.15 and 3.16, we provide a general description for the Transformer in order to explain its architecture , as well as its functions. In section 3.3.4, we present the general architecture of our proposed model, along with its description. Section 3.3.5 presents and evaluates the dataset used in our work. In section 3.3.6, we evaluate the performance of our proposed model and, then, discuss the obtained results. Section 3.3.7 aims at reporting the findings, making inferences, and drawing conclusions. It also provides options for future research.

3.3.2 Related Work

a number of methods and algorithms were proposed for next location prediction. Some of them are [71, 72, 73, 74, 75, 76], and [77]. They focused on O(K) Markovian models to forecast the upcoming location. In Markovian models, the probability of a given location to be the next location depends on its current context, i.e. sequence of K recent visited locations. These models are simple and do not require a large memory to be executed or updated from one state to another. However, their main problem is the difficulty of finding the best value of K that matches a specific situation. In the mentioned works, the authors found that the smaller k was, the better accuracy they got, and the best value that provides the higher accuracy is $K = 2$.

Others concentrated on predicting the next location based on Recurrent Neural Networks (RNN). In [82], Spatial Temporal Recurrent Neural Network (ST-RNN), which follows the same architecture of (RNNs), was proposed. To get more valuable insights, the authors integrated two specific spatial-transition, time-transition matrices that are used to model, respectively, spatial-context, time-context inside each layer in the (RNN) model.

In [78], the researchers introduced Space Time Features-based Recurrent Neural Network (STF-RNN). They treated spatial and temporal historical information as features that the model should learn. They provided a look-up Table layer that is utilized to learn useful patterns from the encoded spatial and temporal information (One hot encoded) and encode them for the second time to a vector of real values that will be fed to the model. One of the drawbacks of (RNNs) is modeling problems that require long-term dependencies due to its architecture, which is composed of standard memory units. Each standard memory unit overwrites its content to compute the new content on the base of the input and the previous hidden state. (LSTMs) have solved the drawbacks of (RNNs). They can model problems that require long-term dependencies more effectively than (RNNs). (LSTM) architecture is composed of, firstly, memory unit that can keep information for long periods of time and, secondly, special units. The latter, which are called gates, control when information enters the memory cell, or when the memory cell forgets the information [90].

In [79], a Spatial-Temporal-Semantic Neural Network algorithm (STS-LSTM) were proposed. The proposed algorithm predicts the next location based on historical information on two steps. The first step is extracting spatial and temporal semantic features by applying STS algorithm. These features will be fed in the second step to an (LSTM) that will be utilized to forecast the upcoming location.

In [65], the researchers presented a new architecture for (LSTM) by adding an embedding layer after the input layer to limit the problems of high dimensionality. The embedding layer transforms sequences of discrete locations into sequences of dense vectors which will be fed to the (LSTM) cells. The latter will predict the next location. It is important to mention that the main drawback of (LSTMs) is that they treat a given sequence sequentially. In other words, the (LSTMs) treat sequence components in a predefined number of steps. Transformers have solved this problem based on attention mechanism. Transformers can capture longer term dependencies naturally between sequence components than (LSTMs) and process them simultaneously [39].

In [80], the researchers presented a new embedding method for embedding the input sequence of discrete locations called loc2vec. They converted the embedded vectors into RGB images. The latter, will be fed to a pretrained (CNN) model. The pretrained (CNN) treats the image as classification task; As a result, it forecasts the future place by classifying the image correctly. In this regard, each class represents a location where the user can reside. The main drawback of (CNNs) is that they also process sequentially the input, by applying $N * M$ filters that scan the input rows one by one horizontally and vertically. Transformers overcome (CNNs) at this point. They have the advantage of scanning any sequence parts simultaneously based on attention mechanism [39].

In [83], BERT Transformer was proposed with Masked Language Modeling that aims at learning bidirectional representations between any sequence parts (tokens) by randomly masking tokens during training and trying to forecast them utilizing Masked Language Modeling MLM objective function. Many Bert designs, such as UNILM [85], ALBERT [86], ROBERTA [84], and XLNET [87], were proposed in Natural Language Understanding (NLU) with different objective functions [91]. For example, Permutation Language Modeling (PLM) objective function is used to model sequence dependencies for XLNET model [91]. The aforementioned transformers are robust models in (NLU) [91]. They are able to model and learn useful patterns and representations for every given sequence. Substantially, in our work, we propose WP-BERTA to predict the next location. WP-BERTA is a model made up of ROBERTA and Bertwordpiece tokenizer. The state of the art methods, i.e. (LSTMs), (CNNs),...etc, which are used to predict the next location, process each part of the input locations sequence sequentially. Nevertheless, Transformers see the entire sequence of locations at once and treat each part in the sequence simultaneously based on attention mechanism [39].

3.3.3 Transformer

Transformer is made up of an encoder and decoder. Its general idea is that the encoder maps the input sequence to a sequence of continuous representations that will be used as an input to the decoder. The latter generates at each time step one output. The output will be used as an input to the decoder at the next time step along side the sequence of continuous representations in order to generate the next output, and so on [39]. Figure 3.15 shows the general architecture of the Transformer.

Encoder: The encoder is composed of N stacked identical layers in which each layer consists of two sub-layers. The first one is a multi-head-self attention layer while the second is position wise fully connected feed forward layer. Around each sub-layer, the model uses residual connections followed by normalization layer [39].
Decoder: The decoder follows the same architecture of the encoder, including the residual connections and normalization layers. Additionally, it consists of a multi-head attention layer that is utilised for the sake of performing multi-head attention over the output of the encoder. In order to prevent the decoder from knowing the future positions, i.e. prediction relying only on current and past positions, a masking mechanism is added to the first multi-head attention layer [39].
Attention: Attention is described by [39] as mapping a query with a given set of key-value pairs. The output of an attention function is computed using the dot scaled product as the weighted sum of the values, wherein the weights represent the compatibility of a query with a corresponding key.
Multi Head Attention: Multi-Head Attention function is defined as multiple self-attention functions that run in parallel and are applied to the projected queries,

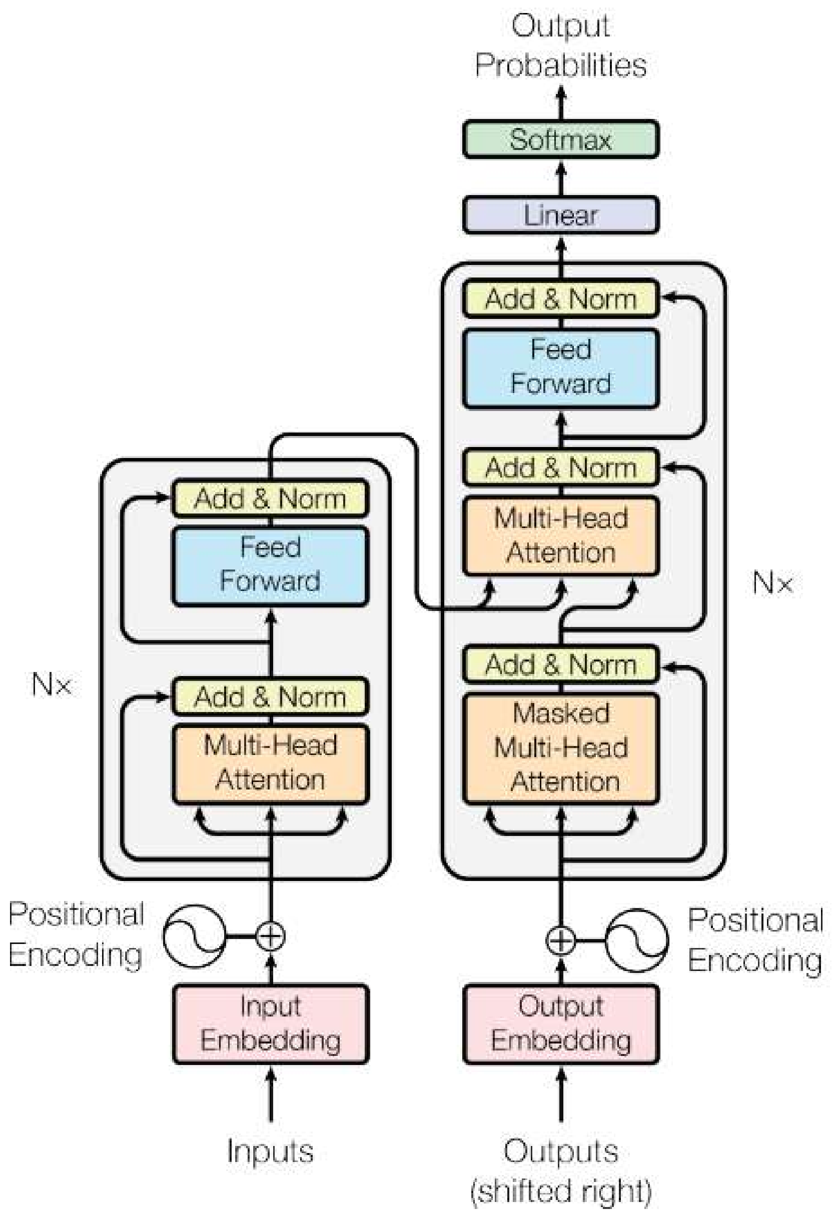

Figure 3.15: General architecture of Transformer [39]

keys, and values. The outputs of each attention head are concatenated and projected other time [39]. Multi-head attention and Self-attention mechanisms are demonstrated respectively in the right and left parts of Figure 3.16.

Position Wise Feed Forward layer: Position Wise layer is a Fully Connected Feed Forward layer. It is made up of 2 transformations using ReLU between them [39].

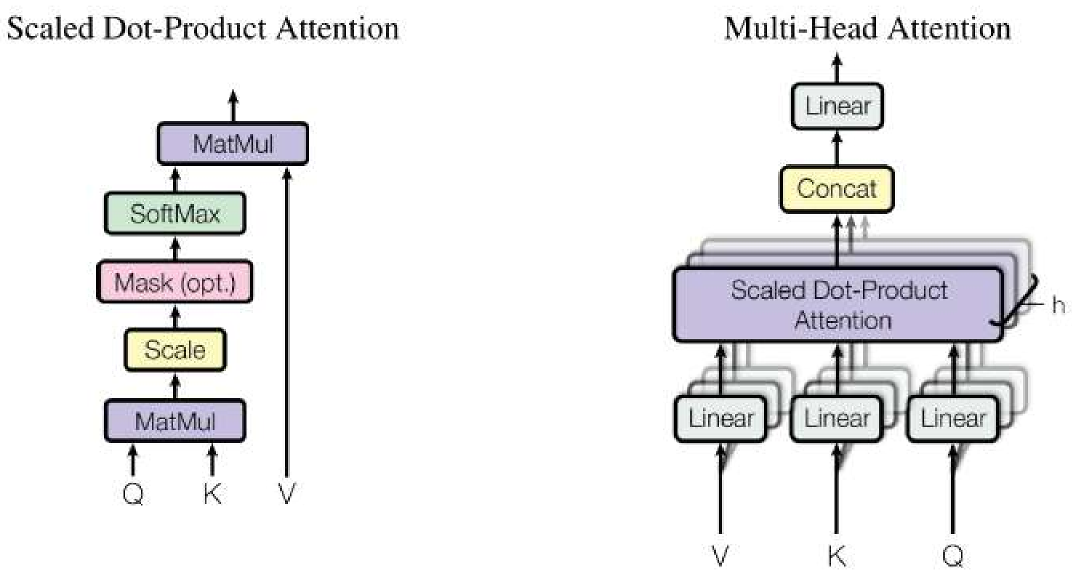

Figure 3.16: Attention function calculated using dot-product at the left, Multi-head attention at the right [39]

Embeddings and softmax: The common learned embeddings, softmax, and transformations are used in the architecture [39].
Positional Encoding: Positional Encoding is added to the input embeddings in the encoder and decoder. Cosine and sine functions of different frequencies are used to compute the positional encoding. The latter adds more information about the positions of the sequence components in order to make use of the order of the sequence [39].

3.3.4 Proposed Approach

In our work, and for the sake of predicting the next location of mobile users, we propose WP-BERTA [92]. It follows the general architecture of Transformers that is proposed by [39]. In reality, WP-BERTA is a combination of Bertwordpiece tokenizer [89, 83] and Roberta [84]. WP-BERTA is composed of L hidden layers and predefined number of self-attention heads, and hidden dimension.It should be noted that the name of the access point to which the user device is connected is defined as a location loc_i. Therefore, the aim of this work is to predict the next location a user will visit based on its historical sequences. Therefore, our proposed approach, which is illustrated in Figure 3.22, will be described. The three phases of the proposed approach are as follows:

3.3.4.1 Preprocessing Phase

3.3.4.1.1 Sub-sequences Generation There are two ways of generating sub-sequences from the sequence of locations of a given user, as [80] mentioned. The first one is called Overlapping in which sub-sequences of length k are overlapped through a specific number of locations m using a sliding window of length s. The second is referred to as Non overlapping method in which sub-sequences do not share common locations and a given sub-sequence is generated by shifting k locations from the precedent sub-sequence. In our work, we have proposed the use of the overlapping sub-sequence division. Accordingly, Table 3.5 shows an example of how the two methods work:

Locations sequence	a b c d e f g h i
Overlapping $k = 5, m = 4, s = 1$	s1= (a b c d e), s2= (b c d e f) s3= (c d e f g), s4= (d e f g h) s5= (e f g h i)

Non overlapping
$k = 3$
	s1= (a b c), s2= (d e f)
	s3=(g h i)

Table 3.5: Generating sub-sequence methods

Note that overlapping method generates more sub-sequences compared to the non-overlapping. This is due to its sliding window; consequently, more samples(sub-sequences) are fed to the model.

3.3.4.1.2 Sub-sequences Treating After the generation of $K + 1$-length sub-sequences, i.e. $(loc_1, \ldots, loc_{k+1}), (loc_{k+2}, \ldots$
$, loc_{2(k+1)}), \ldots, (loc_{N-k}, \ldots, loc_N)$ from the whole sequence of locations of the user $(loc_1, loc_2, \ldots, loc_N)$, and in order to extract our sequence tokens during training and deployment, we propose training Bertwordpiece tokenizer over the sequences of users. Note that bertwordpiece tokenizer uses special tokens to encode any sequence. [CLS] is used at the beginning of each sequence while [SEP] at the end of each sequence, and [MASK] is used to mask a given location in a sequence [83].

3.3.4.2 Training Phase

In order to train WP-BERTA, we have chosen 15% as masked language probability parameter. During training, WP-BERTA uses dynamic masking and generates masking pattern each time a sub-sequence is fed to the model.

3.3.4.3 Deployment

We divide the testing sequence of a given user into sub-sequences of length k plus the special token [MASK] at the end of each sequence, so that the length will be $k + 1$ for each sequence. The developed model WP-BERTA predicts the masked location and returns the top $k \leq 5$ locations that have the higher probability of being the next location. In our model, we have chosen the top 1 that has the highest probability as the predicted next location.

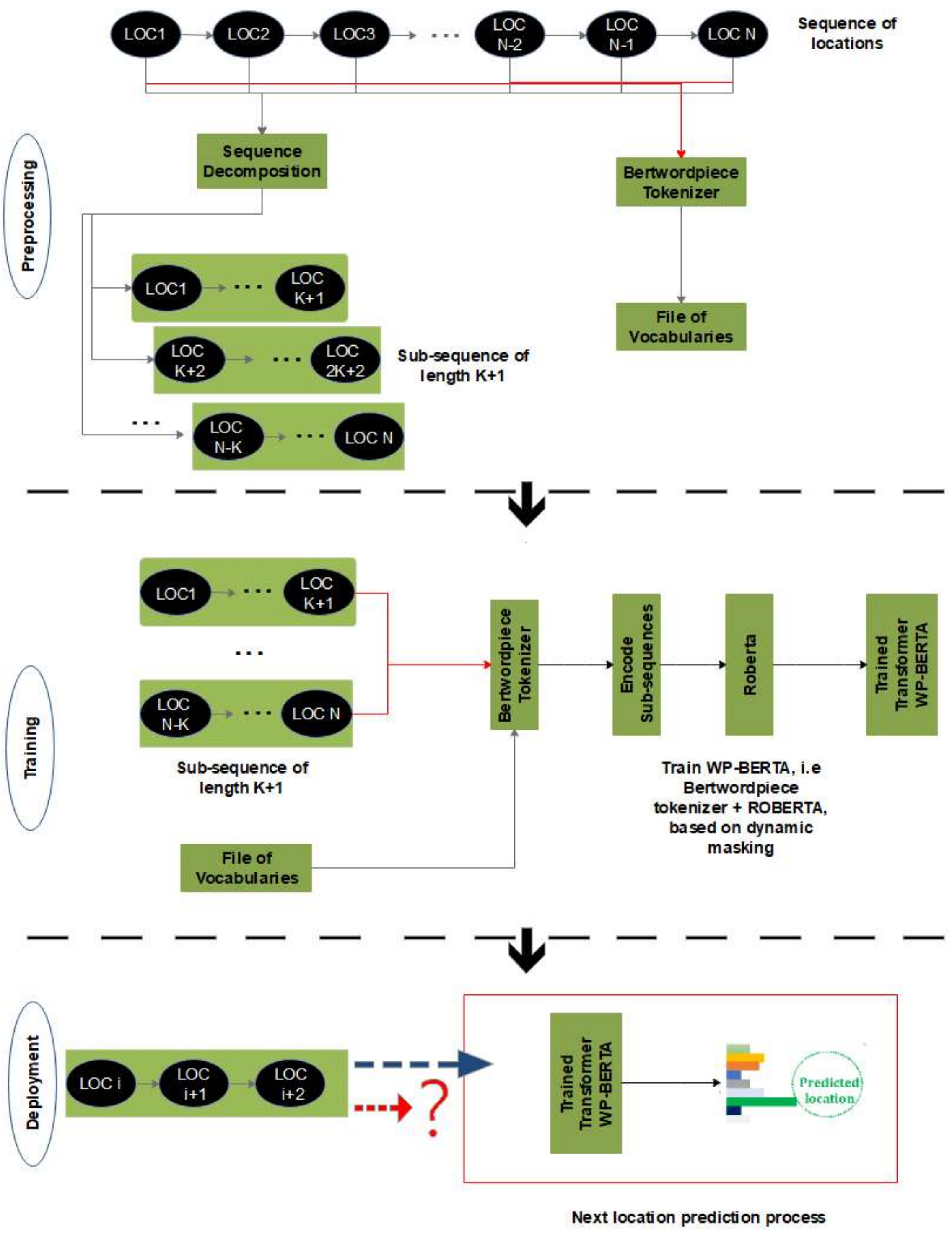

Figure 3.17: General architecture used for next location prediction based on WP-BERTA

3.3.5 Dataset Evaluation

In our work, we have used a real Dataset of WiFi traces collected by Dartmouth College [93] (For details on this dataset, consult [94, 95]). This work's dataset demonstrates that each user trace consists of a series of (time, location) pairs. The location is indicated by the name of an access point to which the user's device is currently connected, as illustrated in Table 3.9.

Timestamp	Location (AP)
1042409101	AcadBldg2AP2
1042453201	AcadBldg12AP2
1042471680	AcadBldg25AP4
1042653300	AcadBldg20AP1
1042657380	OFF

Table 3.6: A Sample of User Trace

Table 3.9 additionally shows an additional location known as OFF. The latter symbolizes the users' exit from the network. The granularity of the timestamps is one second and measured as UNIX timestamps that count the number of seconds since the epoch. It should be emphasized that a user's location does not have to match his specific physical location. It rather denotes a close match of that location to one of the access points (AP) that was servicing the user at the time. This means that the device of the user could associate and re-associate with several different nearby access points without the need of physically moving. The length of the sequence of locations that counts the number of locations in the mobility data sequence of a particular user varies considerably from one user to another. It may exceed 18000 locations for certain users. [80]. In our studies, we picked users at random who had a varied length of sequence of locations ranging from $min = 715$ to $max = 114791$ and a different number of classes ranging from $min = 4$ to $max = 332$. We divided each sequence into 67 percent for creating and training our model and 33 percent for evaluation.

3.3.6 Result and Discussion

We compared the performance of the model we proposed with popular models used to predict the next location. These models are O(K) Markov predictor [71], Embedding-LSTM [65], and the baseline model (LSTM) [96]. We treated next location prediction issue as a multiclassificaiton task. furthermore, we compared our model to (CNN2D) [80] and (SVM) model that is based on one-vs-one (OVO) strategy [97].

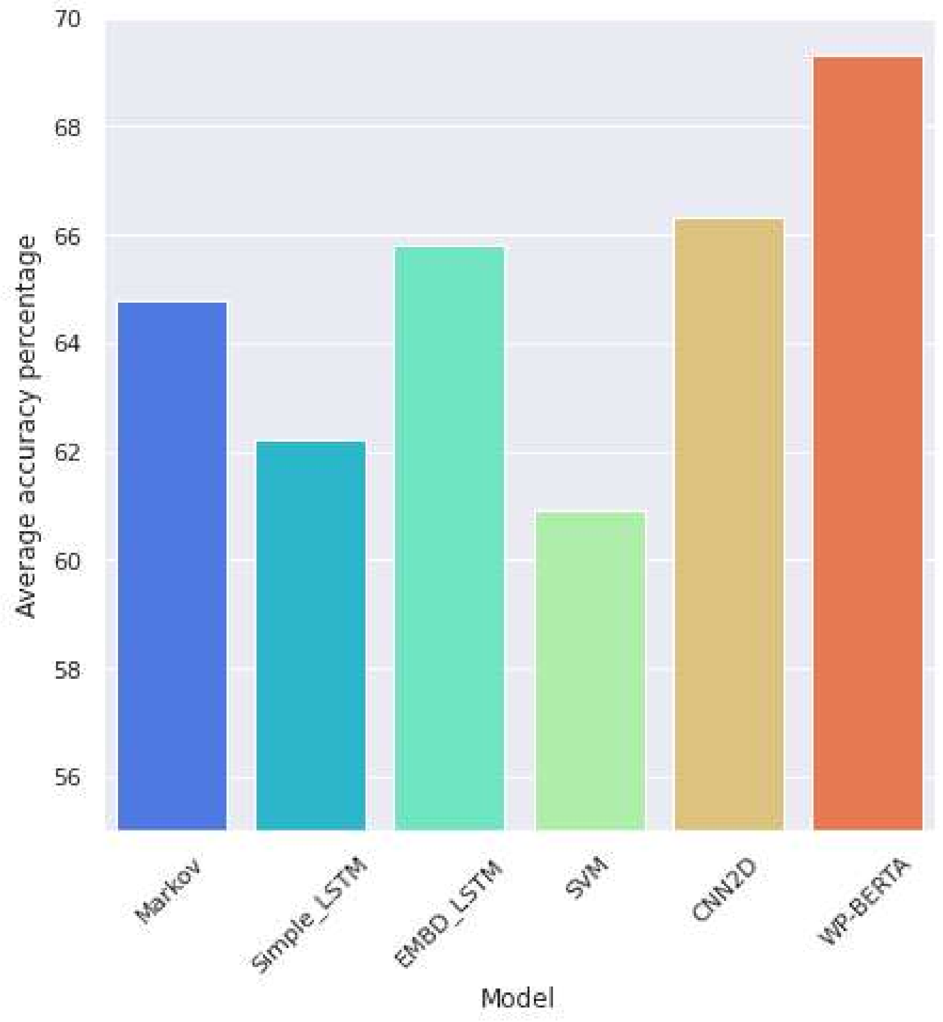

Figure 3.18: Average accuracy of the proposed model compared to state of the art models

We trained the provided models for 60 epochs, with one epoch referred to as the number of training iterations in which a specific neural network completed a full pass of the whole training set. We used $K = 2$ time steps as window for Markov which is the context that gives the higher accuracy for Markov. Moreover, we used $K = 4$ time steps as window for all the other models. In addition, we investigated the performance of the models using different time steps $K = 4, 8, 16, 32$. We used Transformers library that is provided by huggingFace to implement our model [98]. In Figure 3.18, We evaluated our proposed model's average accuracy to the average accuracy of the baseline (LSTM), Embedding-LSTM, Markovian O(K), (SVM), and (CNN2D) models. WP-BERTA improves accuracy by 3% when compared to (CNN2D), the model with the best accuracy among the reference models, 4.5% when compared to O(K) Markov, 3.5% when compared to Embedding-LSTM, 8.5% when compared to (SVM), and 7.5% when compared to baseline (LSTM). Table 3.7 shows the differences in the average accuracy among all presented algorithms. (CNN2D) and Embedding-LSTM models gave almost the same performance results. However, the former (CNN2D) exceeds the latter (Embedding-LSTM) by 0.5%.

	Markov	LSTM	EMBD-LSTM	SVM	CNN2D	**WP-BERTA**
Markov	///	+2.6	-1	+3.9	-1.5	**-4.5**
LSTM	-2.6	///	-3.6	+1.3	-4.1	**-7.1**
EMBD-LSTM	+1	+3.6	///	+4.9	-0.5	**-3.5**
SVM	-3.9	-1.3	-4.9	///	-5.4	**-8.4**
CNN2D	+1.5	+4.5	+0.5	+5.4	///	**-3**
WP-BERTA	**+4.5**	**+7.1**	**+3.5**	**+8.4**	**+3**	///

Table 3.7: Average accuracy differences.

Figure 3.19 shows the cumulative distribution of the prediction accuracy (of 100 users) obtained from the six tested prediction models. We indicate that our WP-BERTA model constantly provides a better prediction accuracy compared to the other models. The median accuracy of WP-BERTA model exceeds 72%.

Figure 3.20 represents the comparison accuracy results achieved utilizing our proposed model, as well as the reference models. Importantly, these results concern 15 users that were randomly chosen. The results further reveal that our model provides a better accuracy for almost all users, which is not the case for four of them. To clarify, and concerning these four users, Markov model gives better accuracy for only two. All the presented models give equal accuracy for the two remaining users, except for Markov which provides less accuracy.

We also observe that Embedding-LSTM and (CNN2D) models provide almost the same accuracy results for almost all users. Additionally, (SVM) and baseline (LSTM) provide weaker results for all users, except for one user. Concerning the latter, all models provide equal accuracy, except Markov that had a weaker accuracy.

We also investigated the performance of our proposed model in comparison with the other presented models with different time steps and different configurations. The results are portrayed in Figure 3.21, as well as Table 3.8.

Figure 3.21 confirms that our proposed model outperformed the other models, consequently, providing higher average accuracy with all time steps used. It also illustrates a positive correlation between time steps and average accuracy for our

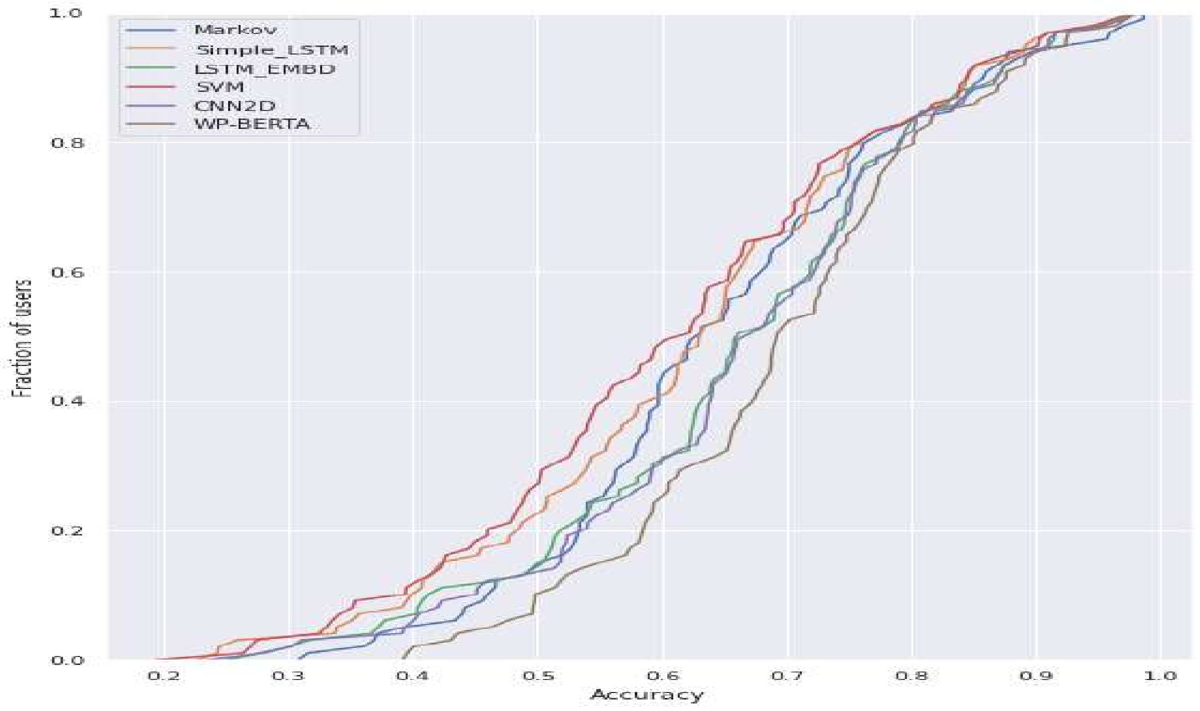

Figure 3.19: CDF of all presented models

proposed model and for Embedding-LSTM model. To clarify, the increase in time steps raises the average accuracy. Furthermore, Figure 3.21 represents a negative correlation for all the other models. That is, the increase time steps affects and reduces the average accuracy.

Table 3.8 displays the average accuracy of our proposed model with different time steps and different configurations, i.e. the number of attention heads and hidden layers inside the model. The Table shows that a positive correlation exists between time steps and average accuracy for all configurations. The best average accuracy with all different time steps was gained using 16 attention-heads and 6 hidden layers. So, we use this configuration as a reference compared to the other configurations.

Compared to the reference configuration, we gained the same average accuracy using 8 attention-heads and 3 hidden layers for 4 and 8 time steps, and a decrease by 1% for both 16 and 32 time steps.

Using 32 attention-heads and 12 hidden layers led to worse results by decreasing the average accuracy compared to the reference configuration by 12%, 3%, 1%, 1% respectively for 4, 8, 16, and 32 time steps.

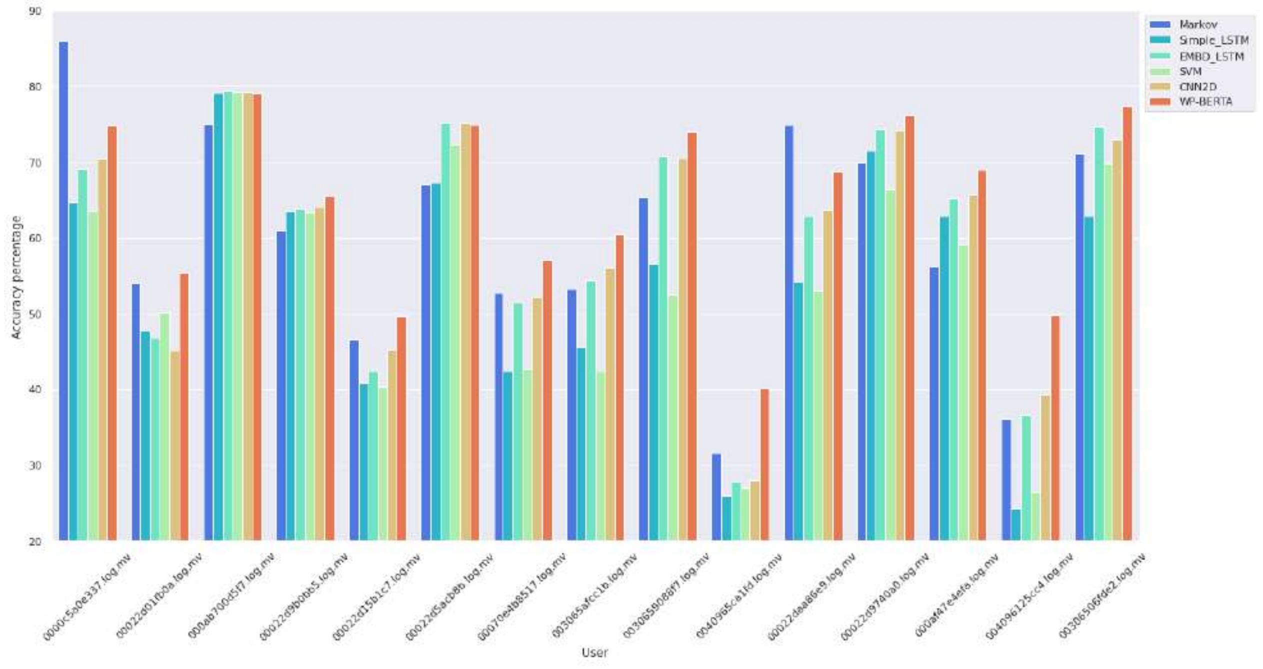

Figure 3.20: WP-BERTA, Markov, baseline LSTM, Embedding-LSTM, SVM, and CNN2D accuracy comparisons for each user

Time steps	4	8	16	32
Attention-heads = 16 Hidden-layers= 6	0.69	0.70	0.71	0.72
Attention-heads = 08 Hidden-layers = 03	0.69	0.70	0.70	0.71
Attention-heads = 32 Hidden-layers = 12	0.57	0.67	0.70	0.71

Table 3.8: Average accuracy of WP-BERTA with different time steps and different configurations.

3.3.7 conclusion

The current work introduced the importance of predicting human mobility for various domains and location-based services such as recommendation systems and urban management. Besides, it presented markovian and deep learning models, which are used for next location prediction, along with their main issue in this task. On that account, and primarily, We presented and explained Transformers that are utilised for (NLP), and then provided their advantages and some of the extensions available in the literature. Chiefly, we proposed a new combination of

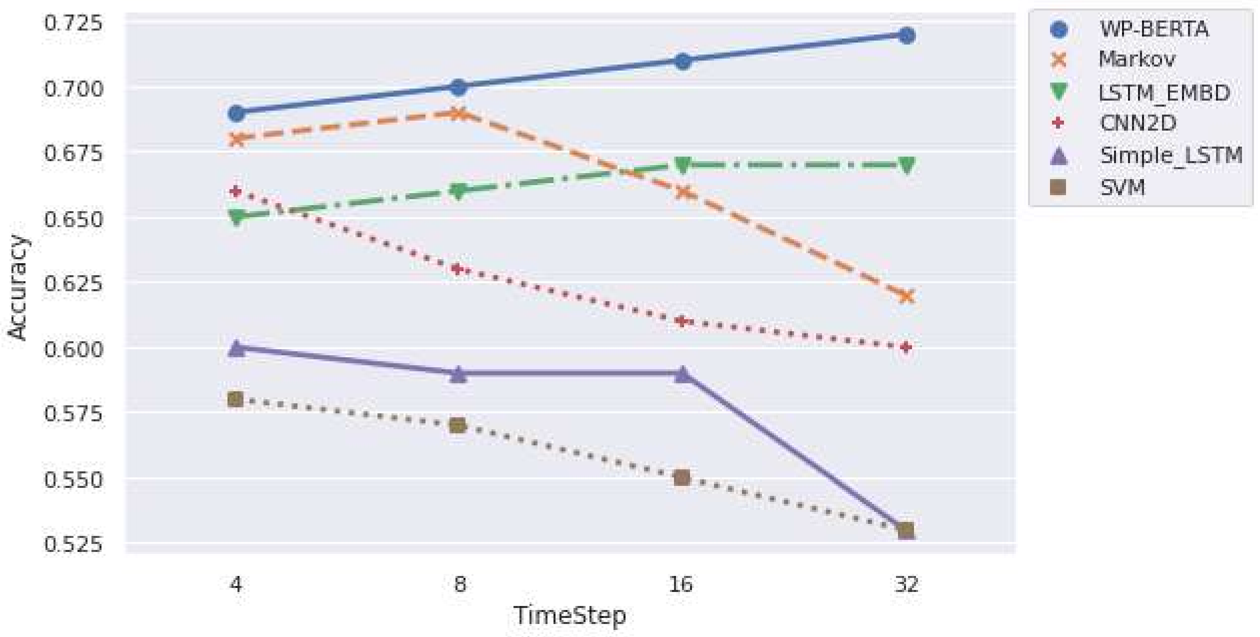

Figure 3.21: The average accuracy of all models provided utilizing different time steps

ROBERTA Transformer and Bertwordpiece tokenizer named WP-BERTA for next location forecasting. We presented and compared the performance of our proposed model to distinct and well-recognized models in the domain. These models include: Markovian, (LSTMs), (CNN2D), and (SVM). Majorly, we tested the performance of the presented models on a real subset of mobility traces. The obtained results demonstrated that our proposed model WP-BERTA outperformed all the presented models by increasing the accuracy of the predicted next location by at least 3%. Changing the hyper-parameters affected the accuracy results of predicting the next location; therefore, choosing the right configuration of hyper-parameters remains a challenge for such model. Substantially, in this work, we proposed a new model and evaluated its performance utilizing only spatial information. In future work, we seek to examine and evaluate the effectiveness of our presented model including temporal features and may propose an extension of it.

3.4 WP-CamemBERT for next location prediction

3.4.1 Introduction

In many fields and location-based services, such as travel recommendation systems [65], civic management [99], and the internet of things [100, 101], accurate prediction of the future locations of mobile users, also known as forecasting upcoming locations, has become a crucial process.

The detection of hidden patterns in users mobility data is required for the purpose of creating next location predictors. As a result, a variety of techniques and algorithms based on analyzing and scanning mobile users' history were proposed. For example, mention [76, 102], and [77] for studies that are based on markovian models. Other research focused on developing deep learning-based predictors for next location. In this regard, some of the works concentrated on LSTM architectures and recurrent neural networks (RNN), with the goal of extracting both spatial and temporal features that will be utilized to create the upcoming location predictors [78, 103, 104, 99, 65]. Further research concentrated on the CNN architecture (convolutional neural networks) and treated the upcoming location prediction issue as an issue of classification. In[80], for example, the authors presented the loc2vec embedding algorithm for the input sequence of locations. Additionally, the authors converted the embedded vectors into RGB images, which will be input into a pre-trained CNN model. To identify important features and hidden patterns that will be utilized to create the upcoming location forecasters, conventional deep learning models, such as RNNs, LSTMs, and CNNs, handle the input sequence sequentially, which is the main disadvantage of such algorithms. Unlike prior algorithm, such as RNNs, LSTMs, and CNNs, Transformers [39] perceive the entire input sequence simultaneously and process any input sequence in parallel. In this regard, numerous transformer-based designs, such as BERT [83], have been proposed for NLP. In reality, BERT transformer [83] is a sophisticated model that can learn bidirectional representations among the input sequence parts.For NLP tasks, a variety of BERT designs were proposed, including ROBERTA [84], CamemBERT [105], XLM [106], and PhoBERT [107].

This study concentrated on transformers. Particularly, we focused on BERT extensions. As a result, we presented WP-Camembert, a novel combination of Bertwordpiece tokenizers [89, 83], and CamemBERT [105], for future locations forecasting. When compared to well-known models in the domain, our presented technique improved the accuracy of predictions by at least 3.2%.

The remainder of the work is structured as follows. The key contributions for the next location prediction issue are presented in section 3.4.2 . We detail our strategy WP-Camembert and explain its three steps, namely preprocessing, training, and deployment, in section 3.4.3. The dataset used in our research is described in section 3.4.4. In section3.4.5, we analyze our model's performance and analyse the findings. The conclusion is represented by section 3.4.6 , in which we summarize our findings and outline our upcoming work.

3.4.2 Related Work

For human mobility prediction, a variety of techniques and models have been developed. Some publications, for example, [71, 76, 102, 77] , use an O(K) markovian model to forecast the future position. The next location will be predicted based on the present context (k recently visited locations). Finding the proper k for a given state remains a difficulty for Markovian models.

Other research focuses on deep learning algorithms to forecast mobile users' next location. The authors introduced Space Time Features-based Recurrent Neural Network (STF-RNN) in [78]. The proposed technique extracts both temporal and spatial features before encoding them into a one hot encoded vectors. The encoded vectors are then passed into a look-up table layer, which uses the encoded vectors to learn useful patterns and encode them again into vectors with real values.

The researchers of [104] presented a next location prediction algorithm utilizing RNN and the mechanism of self-attention. Based on the historical data, the proposed technique extracts geographical, temporal, and user ID features that will be embedded together and utilized to forecast the upcoming location.

The researchers of [99], presented an RNN-based algorithm named DeepVM. The latter aims to foresee the future location of moving vehicles by assessing the likelihood of a vehicle entering a certain place in a specific time period based on the vehicle's mobility trajectory.

The researchers of [103] presented Hierarchical Spatial-Temporal LSTM (HST-LSTM) for upcoming location prediction. The presented model, in fact, is made up of numerous hierarchical layers of Spatial-Temporal LSTM (ST-LSTM). In order to lower data sparsity, the latter incorporates spatial and temporal information into LSTM.

The researchers of [65] proposed a mix of Embedding and LSTM layers for next location forecasting. The embedding layer transforms the series of discrete locations into a sequence of dense vectors. This is subsequently given to the LSTM layers.

The researchers of [80] presented loc2vec as an embedding algorithm for the input series of discrete locations. They further suggested converting the embedded vectors into RGB images that would be input to a pretrained CNN model. The latter will forecast the future location by classifying the input images.

The primary disadvantage of traditional deep learning models, such as RNNs, LSTMs, and CNNs, is the fact that they analyze input sequence parts sequentially. In contrast to traditional deep learning designs, transformers introduced by [39] and based on attention mechanisms bypass this limitation and enable them to observe and process all sequence components simultaneously. (For further details, please see [39]).

Bert transformer was initially introduced for NLP problems by [83]. Masked Language Modeling (MLM) is employed in the introduced algorithm to learn bidirectional representations between any sequence tokens. It masks tokens at random during training and attempts to foresee them via the MLM objective function. Various Bert extensions have been proposed for NLP and with distinct objective functions [91], among them: ROBERTA [84], XLM [106], and PhoBERT [107]. for instance, The XLM [106] model can employ one of the following objective functions: causal language modeling (CLM), MLM, or Translation Language Modeling (TLM).
The extensions indicated above are effective NLP models. They can model and discover hidden patterns for any given sequence. In our study, we primarily propose WP-Camembert for next location forecasting. WP-Camembert is, in reality, a hybrid of Bertwordpiece tokenizers [89, 83], and CamemBERT model [105]. As opposed to traditional models such as LSTMs, CNNs, and so on, WP-Camembert can see the sequence locations at the same time and process them in parallel using an attention mechanism [39].

3.4.3 Proposed Approach

In our research, we propose WP-Camembert for forecasting the user's mobility, i.e. next location prediction. The general architecture of CamemBERT [105] will be used in our model. In fact, WP-Camembert is a mix of Bertwordpiece tokenizer [89, 83], and CamemBERT [105]. As a result, Figure 3.22 depicts our proposed approach, which is divided into three parts as follows:

3.4.3.1 Preprocessing

First, we build $k+1$-length sub-sequences from the entire user sequence of locations. Following that, we train Bertwordpiece tokenizer on all sub-sequences in order to learn and extract all sequence tokens that will be utilized as inputs for the model during training and deployment.

3.4.3.2 Training

WP-Camembert employs dynamic masking for training in the same manner as ROBERTA citeLiu2019. The primary distinction between ROBERTA [84] and WP-Camembert is that ROBERTA citeLiu2019 employs sub-word masking, whereas WP-Camembert employs whole-word masking. According to the above, we trained the model we proposed WP-Camembert with a masked language modeling probability of 15%.

3.4.3.3 Deployment

After creating sub-sequences of length k from the testing sequence of locations, we append the special token [MASK] to the end of each sub-sequence. The trained model will replace the special token [MASK] as the predicted next location, i.e. the location with the highest likelihood of being the next location.

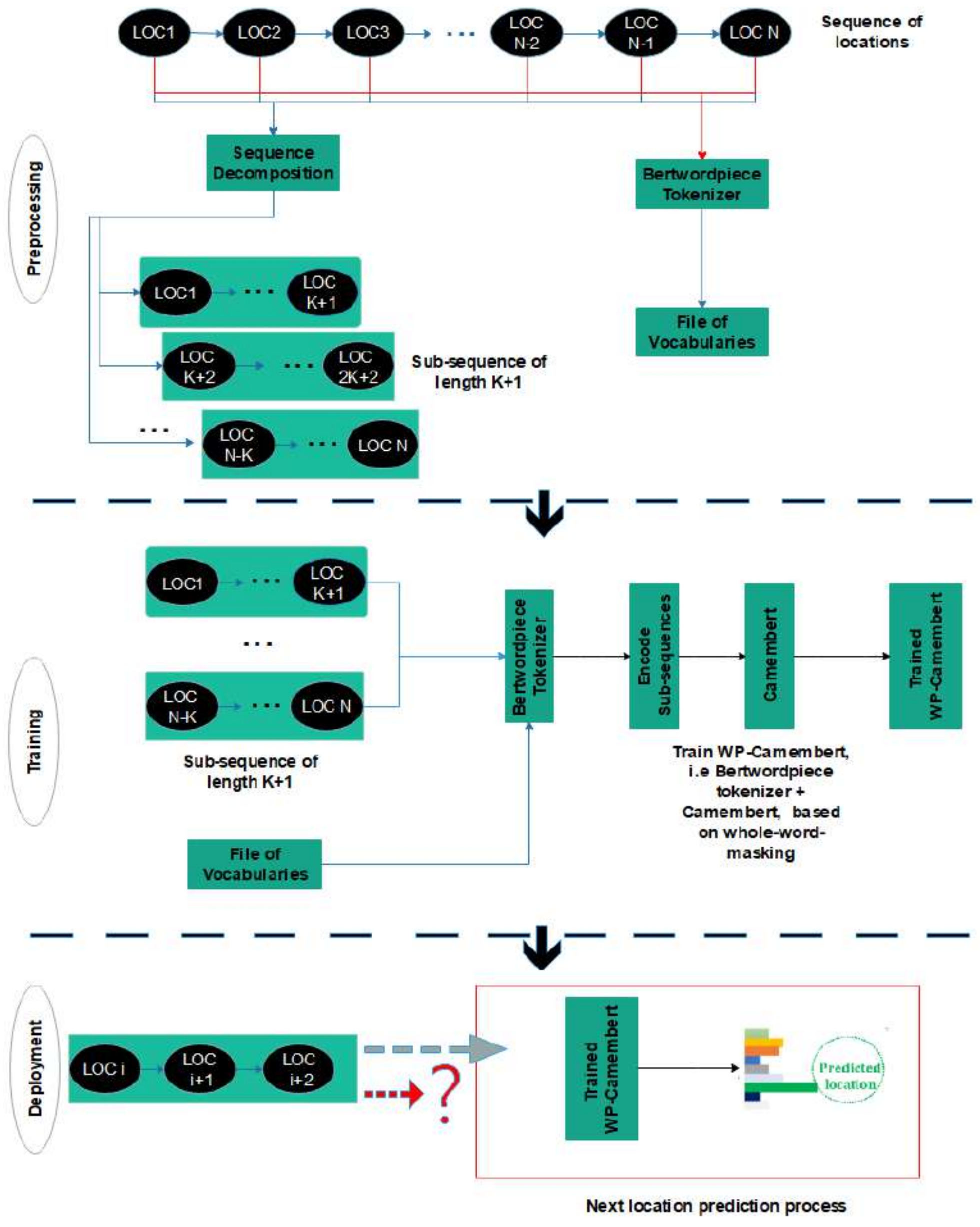

Figure 3.22: WP-Camembert's general architecture

3.4.4 Data set evaluation

We proved our approach using a real dataset of mobile users, i.e. the WIFI dataset, made accessible by Dartmouth College's through CRAWDAD project [93]. As shown in 3.9, each user's file in this dataset is a series of time and location pairs, where the location is the name of the access point (AP) to which the user's device is connected at the specified time. (For further details, check [94, 95]).

The specific location OFF indicates the departure of users from the network,

Timestamp	Location (AP)
994651234	ResBldg56AP13
994651299	OFF
1023208130	ResBldg101AP2
1023208139	ResBldg44AP3
1023314089	ResBldg91AP3

Table 3.9: A user trace example

according to Table 3.9. It should be noted that an AP to which a certain user's device is connected does not always accurately reflect his physical location. It really represents an estimate of that location to the AP that was serving the user at the time. This might result in a Ping-pong effect, in which the user's device connects and disconnects from several neighboring APs without physically moving. Each user's sequence of locations has a unique length that varies from one user to the next. It may exceed 18000 locations for certain users [80]. In our study, we chose users at random who have a different length of the sequence of locations ranging from $min = 715$ to $max = 114791$ and a distinct number of classes ranging from $min = 4$ to $max = 332$. We divided each sequence into 67 percent for creating and training our model and 33 percent for testing.

3.4.5 Result and Discussion

A comparison of the proposed model's performance with the most generally used models for forecasting the next location was made. To clarify, such models refer to the O(K) Markov predictor [71] along with the Embedding-LSTM [65] and the baseline model LSTM [96]. The subject of predicting the next location was handled as a multi-classification task. We also compared our model to the CNN2D [80] and the SVM model, which is based on the one-vs-one (OVO) strategy, according to [97]. The models presented were trained for 60 epochs, with each epoch defined as the number of training iterations in which a neural network completed a full pass of the whole training set. We used a window with K = 2 time steps for Markov. The context that gives Markov the highest accuracy is k = 2. Furthermore, for all

of the other models, we used a window of K = 4 time steps.
We compared the average accuracy of our presented model to the average accuracy of the Markovian O(K) model, Embedding-LSTM, baseline LSTM, CNN2D, and SVM models in 3.23. Our presented model improved accuracy by 3.2% when compared to CNN2D, the reference model with the highest accuracy, 3.7% when compared to Embedding-LSTM, 4.8% when compared to O(K) Markov, 7.3% when compared to baseline LSTM, and 8.7% when compared to baseline SVM. The CNN2D and Embedding-LSTM models performed nearly equally. Nonetheless, the former CNN2D surpasses the latter Embedding-LSTM by 0.5%.
Figure 3.24. depicts the cumulative distribution of prediction accuracy (of 100 users) resulting from the 6 tested prediction models. The former showed that our WP-CAMEMBERT model consistently outperforms the other models in terms of prediction accuracy. The WP-CAMEMBERT model has a median accuracy of more than 72 %.

Figure 3.26. depicts the comparison of accuracy results achieved using our presented model as well as the reference models. importantly, these results correspond to 15 randomly selected users. Except for four users, the findings indicate that our approach improves accuracy for almost all of them. To be more specific, the Markov model improves accuracy for just two of these four users. With the exception of Markov, all of the available models produce the same accuracy for the two remaining users.

We also observed that the Embedding-LSTM and CNN2D models resulted in nearly identical accuracy for almost all users. In addition, SVM and baseline LSTM yield poor results for all users with the exception of one. In terms of the latter, all models provide equal accuracy, with the exception of Markov, which has lower accuracy. We additionally evaluated the performance of our presented model to that of various alternatives with different time steps. Figure 3.25. depicts the results.

Figure 3.25 indicates that our proposed model outperformed the rest of the models, resulting in a higher average accuracy over all time steps. It reveals that there is a positive correlation between time steps and average accuracy for both our presented model and the Embedding-LSTM model. To clarify, the average accuracy is increased by raising the number of time steps. In addition, all of the other models reveal a negative correlation, as seen in Figure 3.25. That is, increasing the number of time steps reduces average accuracy.

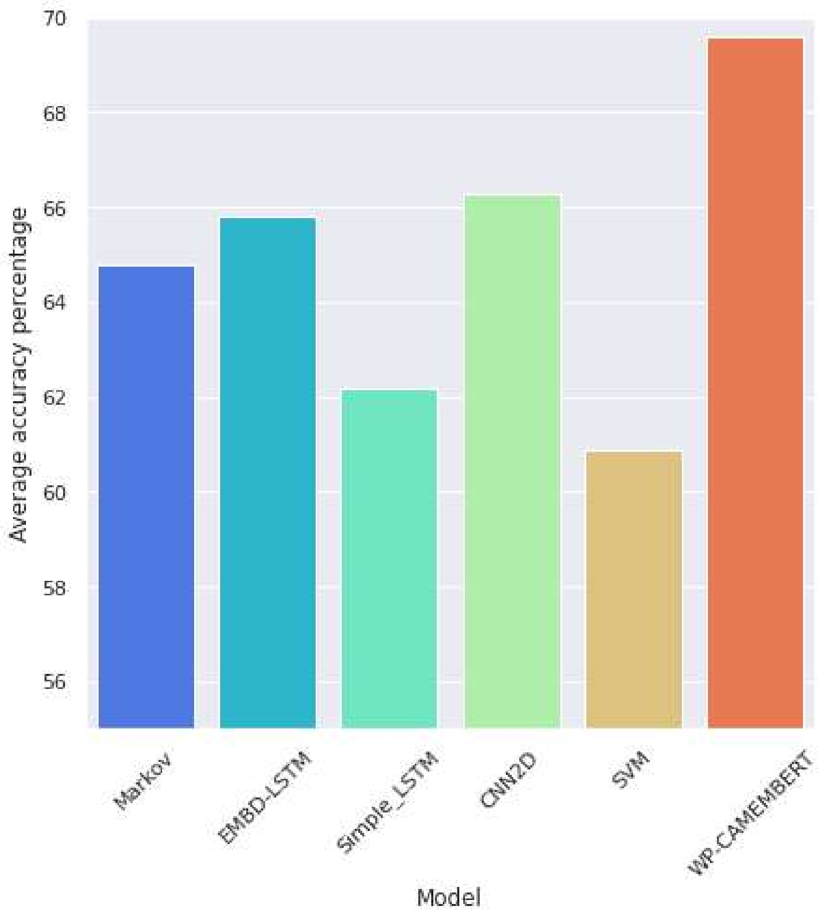

Figure 3.23: WP-CamemBERT average accuracy in compared to the reference models

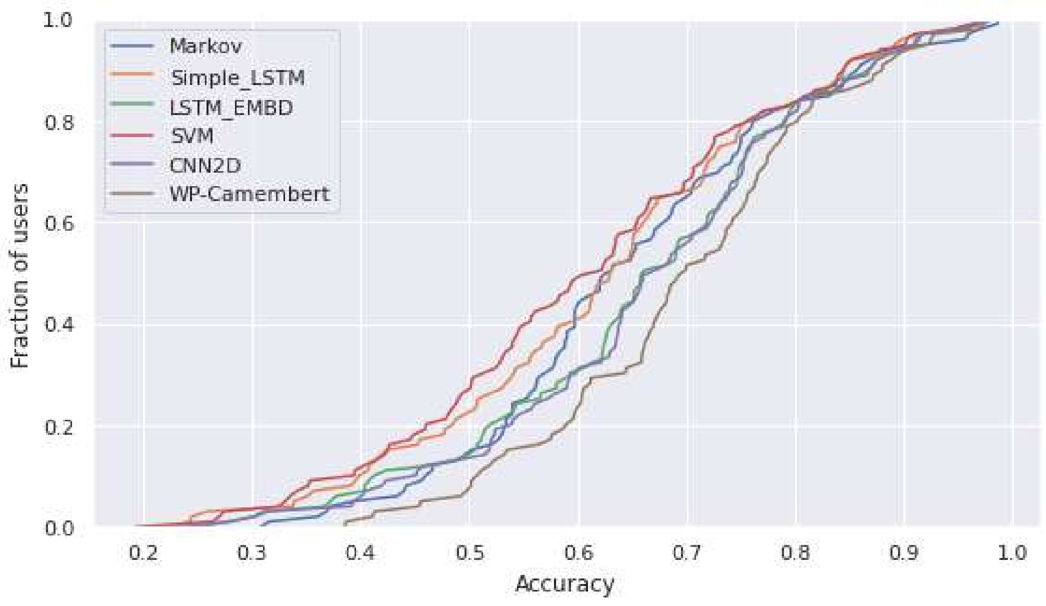

Figure 3.24: CDF of the six presented models.

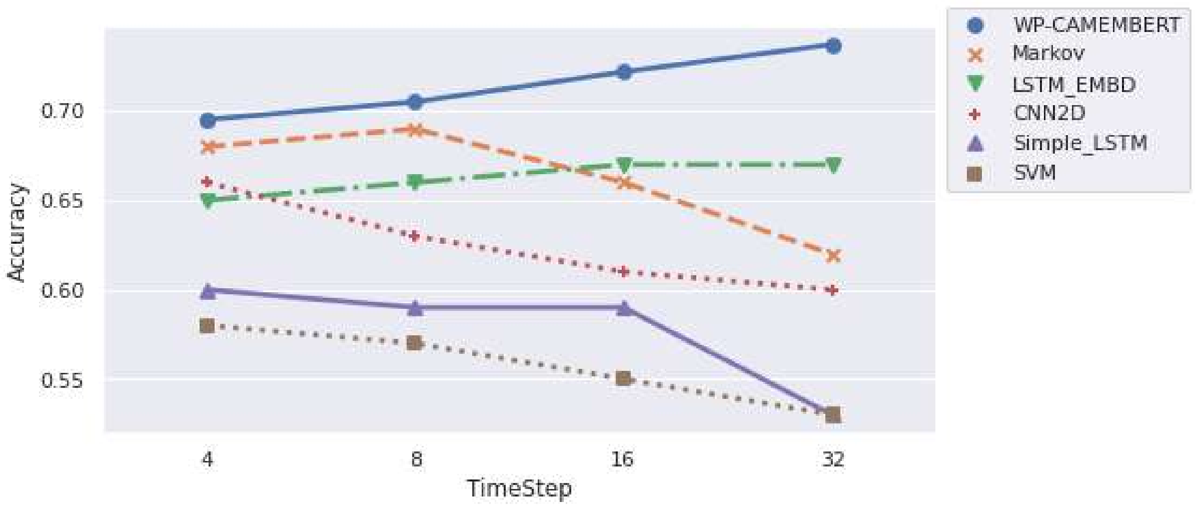

Figure 3.25: Average accuracy of the six described models employing different time steps

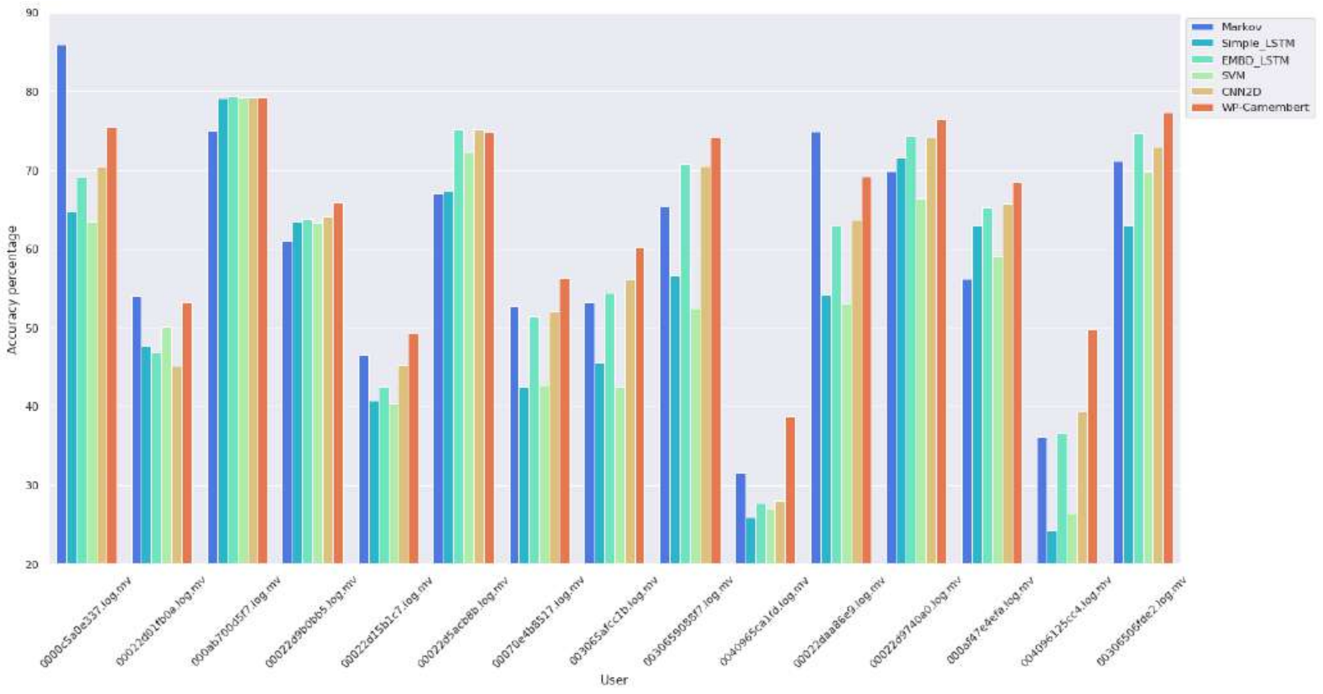

Figure 3.26: WP-CamemBERT, Markov, baseline LSTM, Embedding-LSTM, SVM, and CNN2D accuracy comparison for each user

3.4.6 Conclusion

Predicting user movement has become an important challenge in a variety of fields, including advertisement distribution, urban management, and recommendation systems. As a result, we introduced WP-Camembert, a novel strategy for the future location prediction issue. WP-Camembert, in particular, is a hybrid of bertwordpiece tokenizer and Camembert model. The latter is a popular deep learning technique utilized for natural language processing. On a real subset of mobile users made accessible through the CRAWDAD project, we investigated the efficiency of our approach WP-Camembert by comparing it to the O(k) markovian, Embedding-LSTM, Simple-LSTM, SVM, and CNN models. The findings showed that the model we proposed improved prediction accuracy by a minimum of 3.2 % when compared to CNN, the model with the best accuracy. This study primarily presented a novel combination known as WP-Camembert for upcoming location prediction using just spatial variables. We want to evaluate the efficiency of the model we propose based on temporal characteristics in future research initiatives.

Chapter 4

Conclusions and Future Work

In this thesis, four contributions were proposed in the areas of machine learning and big data. Two contributions were a comparative studies for state of the art methods that are used for DR and BC prediction problems. While the two last contributions were an original propositions named respectively WP-BERTA, and WP-Camembert for solving human mobility prediction problem. Our four contributions are summarized as follows:

- **First contribution:** comparative study for dimensionality reduction techniques for big data. We shed the light on the importance of dimensionality reduction techniques, and how high dimensionality affect negatively the analysis process in big data. Therefore, we review the main methods that are used in the literature for reducing high dimensionality in big data. Also, a detailed comparison were provided between all the presented methods.

- **Second contribution:** we focused in this contribution in studying how dimensionality reduction techniques affect breast cancer prediction. Thus, a practical comparison were proposed using the following five DRTs: AE, T-SNE, RFE, ISOMAP, and PCA, and MLP, and SVM predictors. The results showed that choosing the right combination of DRTs and predictors may increase the prediction accuracy, while choosing the wrong combination may lead to the worst results.

- **Third contribution:** A new deep learning approach was proposed in this contribution named WP-BERTA for solving the next location prediction problem. Our proposed approach was compared to five famous methods used for solving the same problem. The reference models are: Markov, simple LSTM, Embedding-LSTM, SVM, and CNN2D. WP-BERTA outperformed all the presented methods and increased the prediction accuracy by at least 3% compared to the model that resulted the highest accuracy.

- **Fourth contribution:** A new deep learning approach was proposed in this contribution named WP-CamemBERT for solving the next location prediction problem. Our proposed approach was compared to five famous methods that are: Markov, simple LSTM, Embedding-LSTM, SVM, and CNN2D. WP-CamemBERT outperformed all the presented methods and increased the prediction accuracy by at least 3.2% compared to the model that resulted the highest accuracy.

In this thesis, two datasets were used to validate our findings:

Wisconsin Diagnostic Breast Cancer dataset WDBC: from the UCI machine learning repository. The dataset contains 596 instances divided into two classes (37.3% malignant, 62.7% benign). Each raw of data is composed of thirty two features

CRAWDAD WI-FI dataset: we have presented and assessed some of machine learning models for next location prediction problem based on a real subset of actual mobility data provided by the Crawford project.

Actually, the two last contributions are both combinations of a Bertwordpiece tokenizer, and two different transformer architectures that are: ROBERTA, and CamemBERT. Bertwordpiece tokenizer was used to learn and extract all the sequence tokens of the input sequence. Then, we proposed to combine the trained Bertwordpiece tokenizer and two variations of transformer architectures that are: Roberta, and CamemBERT for predicting the next location of a specific user (predicting in which access point, the user will be connected to).

Our proposed models are based only on spatial features for predicting the next location of a given user. Our future work, will focus on including temporal features and investigate the performance of our models. Also, we seek to explore a variety of large-scale datasets such as GPS-based datasets to validate our findings. In addition, in order to find more useful features that can help us predict human movements more accurately, We are interested in exploiting different types of datasets, such as Transportation and Transit Data, and social networks data like Twitter and Facebook.

Bibliography

[1] Jinchuan CHEN, Yueguo CHEN, Xiaoyong DU, Cuiping LI, Jiaheng LU, Suyun ZHAO, and Xuan ZHOU. “Big data challenge: a data management perspective.” In: *Frontiers of computer Science* 7.2 (2013), pp. 157–164 (cit. on pp. 19, 45, 46).

[2] Aisha SIDDIQA, Ibrahim Abaker Targio HASHEM, Ibrar YAQOOB, Mohsen MARJANI, Shahabuddin SHAMSHIRBAND, Abdullah GANI, and Fariza NASARUDDIN. “A survey of big data management: Taxonomy and state-of-the-art.” In: *Journal of Network and Computer Applications* 71 (2016), pp. 151–166 (cit. on pp. 19, 45).

[3] Khadija EL BOUCHEFRY and Rafael S de SOUZA. “Learning in big data: Introduction to machine learning.” In: *Knowledge Discovery in Big Data from Astronomy and Earth Observation*. Elsevier, 2020, pp. 225–249 (cit. on p. 19).

[4] Idris GOKSU. “Bibliometric mapping of mobile learning.” In: *Telematics and Informatics* 56 (2021), p. 101491 (cit. on p. 20).

[5] Hongbo SI, Yue WANG, Jian YUAN, and Xiuming SHAN. “Mobility prediction in cellular network using hidden markov model.” In: *2010 7th IEEE consumer communications and networking conference*. IEEE. 2010, pp. 1–5 (cit. on p. 20).

[6] Trinh Minh Tri DO, Olivier DOUSSE, Markus MIETTINEN, and Daniel GATICA-PEREZ. “A probabilistic kernel method for human mobility prediction with smartphones.” In: *Pervasive and Mobile Computing* 20 (2015), pp. 13–28 (cit. on p. 20).

[7] Baichuan MO, Zhan ZHAO, Haris N KOUTSOPOULOS, and Jinhua ZHAO. “Individual mobility prediction: an interpretable activity-based hidden markov approach.” In: *arXiv preprint arXiv:2101.03996* (2021) (cit. on p. 20).

[8] WORLD HEALTH ORGANIZATION. 2021 (cit. on pp. 20, 57).

[9] Djihane Houfani, Sihem Slatnia, Okba Kazar, Hamza Saouli, and Abdelhak Merizig. “Artificial intelligence in healthcare: a review on predicting clinical needs.” In: *International Journal of Healthcare Management* (2021), pp. 1–9 (cit. on pp. 21, 57).

[10] Ashraf Osman Ibrahim and Siti Mariyam Shamsuddin. “Intelligent breast cancer diagnosis based on enhanced Pareto optimal and multilayer perceptron neural network.” In: *International Journal of Computer Aided Engineering and Technology* 10.5 (2018), pp. 543–556. issn: 17572665. doi: `10.1504/ijcaet.2018.094327` (cit. on pp. 21, 57, 58).

[11] Seral Şahan, Kemal Polat, Halife Kodaz, and Salih Güneş. “A new hybrid method based on fuzzy-artificial immune system and k-nn algorithm for breast cancer diagnosis.” In: *Computers in Biology and Medicine* 37.3 (2007), pp. 415–423 (cit. on pp. 21, 57).

[12] Arpit Bhardwaj and Aruna Tiwari. “Breast cancer diagnosis using Genetically Optimized Neural Network model.” In: *Expert Systems with Applications* 42.10 (2015), pp. 4611–4620. issn: 09574174. doi: `10.1016/j.eswa.2015.01.065`. url: `http://dx.doi.org/10.1016/j.eswa.2015.01.065` (cit. on pp. 21, 57, 58).

[13] Haifeng Wang, Bichen Zheng, Sang Won Yoon, and Hoo Sang Ko. “A support vector machine-based ensemble algorithm for breast cancer diagnosis.” In: *European Journal of Operational Research* 267.2 (2018), pp. 687–699. issn: 03772217. doi: `10.1016/j.ejor.2017.12.001`. url: `https://doi.org/10.1016/j.ejor.2017.12.001` (cit. on pp. 21, 57, 58).

[14] Shaeela Ayesha, Muhammad Kashif Hanif, and Ramzan Talib. “Overview and comparative study of dimensionality reduction techniques for high dimensional data.” In: *Information Fusion* 59 (2020), pp. 44–58 (cit. on p. 21).

[15] Xian-Fang Song, Yong Zhang, Yi-Nan Guo, Xiao-Yan Sun, and Yong-Li Wang. “Variable-size cooperative coevolutionary particle swarm optimization for feature selection on high-dimensional data.” In: *IEEE Transactions on Evolutionary Computation* 24.5 (2020), pp. 882–895 (cit. on p. 21).

[16] Barbara Pes. “Ensemble feature selection for high-dimensional data: a stability analysis across multiple domains.” In: *Neural Computing and Applications* 32.10 (2020), pp. 5951–5973 (cit. on p. 21).

[17] Maria Cristina Mariani, Osei Kofi Tweneboah, and Maria Pia Beccar-Varela. *Data Science in Theory and Practice: Techniques for Big Data Analytics and Complex Data Sets*. John Wiley & Sons, 2021 (cit. on pp. 23, 26, 27).

[18] Thomas Erl, Wajid Khattak, and Paul Buhler. Vol. 1. Prentice Hall Boston, 2016 (cit. on pp. 23–28).

[19] Amir Gandomi and Murtaza Haider. “Beyond the hype: Big data concepts, methods, and analytics.” In: *International journal of information management* 35.2 (2015), pp. 137–144 (cit. on pp. 24, 45, 46).

[20] Junfei Qiu, Qihui Wu, Guoru Ding, Yuhua Xu, and Shuo Feng. “A survey of machine learning for big data processing.” In: *EURASIP Journal on Advances in Signal Processing* 2016.1 (2016), pp. 1–16 (cit. on pp. 24, 45, 46).

[21] Yuri Demchenko, Cees De Laat, and Peter Membrey. “Defining architecture components of the Big Data Ecosystem.” In: *2014 International conference on collaboration technologies and systems (CTS)*. IEEE. 2014, pp. 104–112 (cit. on p. 26).

[22] Saeed Shahrivari. “Beyond batch processing: towards real-time and streaming big data.” In: *Computers* 3.4 (2014), pp. 117–129 (cit. on p. 27).

[23] Poonam Vashisht and Vishal Gupta. “Big data analytics techniques: A survey.” In: *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*. IEEE. 2015, pp. 264–269 (cit. on p. 27).

[24] Jasmine Zakir, Tom Seymour, and Kristi Berg. “Big data analytics.” In: *Issues in Information Systems* 16.2 (2015) (cit. on p. 27).

[25] Peter Ghavami. *Big Data Management: Data Governance Principles for Big Data Analytics*. Walter de Gruyter GmbH & Co KG, 2020 (cit. on p. 27).

[26] Shui Yu and Song Guo. *Big data concepts, theories, and applications*. Springer, 2016 (cit. on pp. 29–31).

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016 (cit. on pp. 32, 39).

[28] Yaser S. Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning From Data*. AMLBook, 2012. isbn: 1600490069 (cit. on pp. 33, 34).

[29] Corinna Cortes Yann LeCun and Chris Burges. *MNIST Database of Handwritten digits*. url: `http://yann.lecun.com/exdb/mnist/` (cit. on pp. 33, 54).

[30] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. url: `http://archive.ics.uci.edu/ml` (cit. on pp. 33, 60).

[31] T.M. Mitchell. *Machine Learning*. McGraw-Hill international editions - computer science series. McGraw-Hill Education, 1997. isbn: 9780070428072. url: `https://books.google.dz/books?id=xOGAngEACAAJ` (cit. on p. 34).

[32] Aston Zhang, Zachary C Lipton, Mu Li, and Alexander J Smola. "Dive into deep learning." In: *arXiv preprint arXiv:2106.11342* (2021) (cit. on pp. 35, 40).

[33] Jafar Alzubi, Anand Nayyar, and Akshi Kumar. "Machine learning from theory to algorithms: an overview." In: *Journal of physics: conference series*. Vol. 1142. 1. IOP Publishing. 2018, p. 012012 (cit. on pp. 35, 36, 38).

[34] Batta Mahesh. "Machine Learning Algorithms-A Review." In: *International Journal of Science and Research (IJSR).[Internet]* 9 (2020), pp. 381–386 (cit. on pp. 35–37).

[35] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on pp. 37, 38).

[36] Viet-Hung Dang, Nhat-Duc Hoang, Le-Mai-Duyen Nguyen, Dieu Tien Bui, Pijush Samui, et al. "A novel GIS-based random forest machine algorithm for the spatial prediction of shallow landslide susceptibility." In: *Forests* 11.1 (2020), p. 118 (cit. on p. 39).

[37] Li Deng and Dong Yu. "Deep learning: methods and applications." In: *Foundations and trends in signal processing* 7.3–4 (2014), pp. 197–387 (cit. on p. 39).

[38] Witold Pedrycz and Shyi-Ming Chen. *Deep Learning: Concepts and Architectures*. Springer, 2020 (cit. on pp. 40–43).

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In: *Advances in Neural Information Processing Systems* 2017-Decem.Nips (2017), pp. 5999–6009. issn: 10495258. arXiv: `arXiv:1706.03762v5` (cit. on pp. 43, 69–73, 83, 85).

[40] Krishna Karthik Gadiraju, Manik Verma, Karen C Davis, and Paul G Talaga. "Benchmarking performance for migrating a relational application to a parallel implementation." In: *Future Generation Computer Systems* 63 (2016), pp. 148–156 (cit. on p. 45).

[41] John Gantz and David Reinsel. "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east." In: *IDC iView: IDC Analyze the future* 2007.2012 (2012), pp. 1–16 (cit. on pp. 45, 46).

[42] *Big Data.(n.d)*. `https://www.gartner.com/en/information-technology/glossary/big-data`. Retrieved May 13, 2020 (cit. on p. 45).

[43] Min Chen, Shiwen Mao, and Yunhao Liu. "Big data: A survey." In: *Mobile networks and applications* 19.2 (2014), pp. 171–209 (cit. on p. 46).

[44] Tonglin Zhang and Baijian Yang. "Dimension reduction for big data." In: *Statistics and Its Interface* 11.2 (2018), pp. 295–306 (cit. on pp. 47, 54).

[45] Tu-Bao Ho. *Dimensionality Reduction in Big Data with Nonnegative Matrix Factorization.* Tech. rep. JAPAN ADVANCED INSTITUTE OF SCIENCE and TECHNOLOGY ISHIKAWA, 2017 (cit. on pp. 48, 49, 52, 54).

[46] Muneeba Sirshar, Sajid Saleem, Muhammad U Ilyas, Muhammad Murtaza Khan, Mohammed Saeed Alkatheiri, and Jalal S Alowibdi. "Big Data Dimensionality Reduction for Wireless Sensor Networks Using Stacked Autoencoders." In: *The International Research & Innovation Forum.* Springer. 2019, pp. 391–400 (cit. on pp. 49, 52–54, 59).

[47] Ahmad Taher Azar and Aboul Ella Hassanien. "Dimensionality reduction of medical big data using neural-fuzzy classifier." In: *Soft computing* 19.4 (2015), pp. 1115–1127 (cit. on pp. 49–51, 54).

[48] Dewa Made Sri Arsa, Grafika Jati, Aprinaldi Jasa Mantau, and Ito Wasito. "Dimensionality reduction using deep belief network in big data case study: Hyperspectral image classification." In: *2016 International Workshop on Big Data and Information Security (IWBIS).* IEEE. 2016, pp. 71–76 (cit. on pp. 51, 52, 54).

[49] *CBCL FACE DATABASE.* URL: `http://www.ai.mit.edu/projects/cbcl.old/software-datasets/FaceData.html` (cit. on p. 54).

[50] Rob Fergus, Antonio Torralba, and William T. Freeman. "Tiny Images Dataset." In: (). URL: `http://horatio.cs.nyu.edu/mit/tiny/data/index.html` (cit. on p. 54).

[51] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository.* 2017. URL: `http://archive.ics.uci.edu/ml` (cit. on p. 54).

[52] Amit Kumar and Bikash Kanti Sarkar. "A case study on machine learning classification." In: *International Journal of Information and Decision Sciences* 9.2 (2017), pp. 179–208. ISSN: 17567025. DOI: `10.1504/IJIDS.2017.084885` (cit. on pp. 57, 58).

[53] Djihane Houfani, Sihem Slatnia, Okba Kazar, Noureddine Zerhouni, Hamza Saouli, and Ikram Remadna. "Breast cancer classification using machine learning techniques: a comparative study." In: *Medical Technologies Journal* 4.2 (2020), pp. 535–544 (cit. on pp. 57, 58, 61).

[54] Bichen Zheng, Sang Won Yoon, and Sarah S. Lam. "Breast cancer diagnosis based on feature extraction using a hybrid of K-means and support vector machine algorithms." In: *Expert Systems with Applications* 41.4 PART 1 (2014), pp. 1476–1482. ISSN: 09574174. DOI: `10.1016/j.eswa.2013.08.044`. URL: `http://dx.doi.org/10.1016/j.eswa.2013.08.044` (cit. on p. 58).

[55] U Karthik Kumar, M B Sai Nikhil, and K Sumangali. "1-vote-Naïve Bayes SMV J48.pdf." In: August (2017), pp. 108–114 (cit. on p. 58).

[56] Emina ALIČKOVIĆ and Abdulhamit SUBASI. "Breast cancer diagnosis using GA feature selection and Rotation Forest." In: *Neural Computing and Applications* 28.4 (2017), pp. 753–763. ISSN: 09410643. DOI: `10.1007/s00521-015-2103-9` (cit. on p. 58).

[57] Htet Thazin TIKE THEIN and Khin Mo MO TUN. "An Approach for Breast Cancer Diagnosis Classification Using Neural Network." In: *Advanced Computing: An International Journal* 6.1 (2015), pp. 1–11. ISSN: 2229726X. DOI: `10.5121/acij.2015.6101` (cit. on p. 58).

[58] Ahmed M. ABDEL-ZAHER and Ayman M. ELDEIB. "Breast cancer classification using deep belief networks." In: *Expert Systems with Applications* 46 (2016), pp. 139–144. ISSN: 09574174. DOI: `10.1016/j.eswa.2015.10.015`. URL: `http://dx.doi.org/10.1016/j.eswa.2015.10.015` (cit. on p. 58).

[59] Jian Sheng GUAN, Lo Yi LIN, Guo Li JI, Chih Min LIN, Tien Loc LE, and Imre J. RUDAS. "Breast tumor computer-aided diagnosis using self-validating cerebellar model neural networks." In: *Acta Polytechnica Hungarica* 13.4 (2016), pp. 39–52. ISSN: 17858860. DOI: `10.12700/APH.13.4.2016.4.3` (cit. on p. 58).

[60] Laurens VAN DER MAATEN and Geoffrey HINTON. "Visualizing data using t-SNE." In: *Journal of machine learning research* 9.11 (2008) (cit. on p. 59).

[61] Isabelle GUYON, Jason WESTON, Stephen BARNHILL, and Vladimir VAPNIK. "Gene selection for cancer classification using support vector machines." In: *Machine learning* 46.1 (2002), pp. 389–422 (cit. on p. 59).

[62] Joshua B TENENBAUM, Vin DE SILVA, and John C LANGFORD. "A global geometric framework for nonlinear dimensionality reduction." In: *science* 290.5500 (2000), pp. 2319–2323 (cit. on p. 59).

[63] Harold HOTELLING. "Analysis of a complex of statistical variables into principal components." In: *Journal of educational psychology* 24.6 (1933), p. 417 (cit. on p. 59).

[64] Damanjit KAUR, Shweta KHANNA, and Darpandeep AULAKH. "A Collaborative Location Based Travel Recommendation System through Enhanced Rating Prediction for the Group of Users." In: *Structural Chemistry* 24.1 (2013), pp. 357–367. ISSN: 10400400. DOI: `10.1007/s11224-012-0084-1` (cit. on p. 68).

[65] Alessandro CRIVELLARI and Euro BEINAT. "LSTM-based deep learning model for predicting individual mobility traces of short-term foreign tourists." In: *Sustainability (Switzerland)* 12.1 (2020), p. 349. ISSN: 20711050. DOI: `10.3390/su12010349` (cit. on pp. 68, 70, 77, 83, 84, 88).

[66] Alicia Rodriguez-Carrion, Carlos Garcia-Rubio, Celeste Campo, Alberto Cortés-Martín, Estrella Garcia-Lozano, and Patricia Noriega-Vivas. “Study of LZ-based location prediction and its application to transportation recommender systems.” In: *Sensors (Switzerland)* 12.6 (2012), pp. 7496–7517. ISSN: 14248220. DOI: 10.3390/s120607496 (cit. on p. 68).

[67] Jianming Lv, Qing Li, Qinghui Sun, and Xintong Wang. “T-CONV: A Convolutional Neural Network for Multi-scale Taxi Trajectory Prediction.” In: *Proceedings - 2018 IEEE International Conference on Big Data and Smart Computing, BigComp 2018* (2018), pp. 82–89. DOI: 10.1109/BigComp.2018.00021. arXiv: 1611.07635 (cit. on p. 68).

[68] Renhe Jiang, Xuan Song, Zipei Fan, Tianqi Xia, Quanjun Chen, Satoshi Miyazawa, and Ryosuke Shibasaki. “DeepUrbanMomentum: An online deep-learning system for short-term urban mobility prediction.” In: *32nd AAAI Conference on Artificial Intelligence, AAAI 2018* (2018), pp. 784–791 (cit. on p. 68).

[69] Wei Liu and Yozo Shoji. “DeepVM: RNN-Based Vehicle Mobility Prediction to Support Intelligent Vehicle Applications.” In: *IEEE Transactions on Industrial Informatics* 16.6 (2020), pp. 3997–4006. ISSN: 19410050. DOI: 10.1109/TII.2019.2936507 (cit. on p. 68).

[70] Chunyang Liu, Jiping Liu, Jian Wang, Shenghua Xu, Houzeng Han, and Yang Chen. “An Attention-Based Spatiotemporal Gated Recurrent Unit Network for Point-of-Interest Recommendation.” In: *ISPRS International Journal of Geo-Information* 8.8 (2019), p. 355. ISSN: 2220-9964. DOI: 10.3390/ijgi8080355 (cit. on p. 68).

[71] Libo Song, David Kotz, Ravi Jain, and Xiaoning He. “Evaluating location predictors with extensive Wi-Fi mobility data.” In: *Proceedings - IEEE INFOCOM* 2 (2004), pp. 1414–1424. ISSN: 0743166X. DOI: 10.1145/965732.965747 (cit. on pp. 68, 69, 77, 84, 88).

[72] Libo Song, David Kotz, Ravi Jain, and Xiaoning He. “Evaluating next-cell predictors with extensive Wi-Fi mobility data.” In: *Proceedings - IEEE INFOCOM* 2.12 (2006), pp. 1414–1424. ISSN: 0743166X. DOI: 10.1145/965732.965747 (cit. on pp. 68, 69).

[73] Sébastien Gambs, Marc Olivier Killijian, and Miguel Núñez Del Prado Cortez. “Show me how you move and i will tell you who you are.” In: *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS, SPRINGL 2010* (2010), pp. 34–41. ISSN: 2013-1631. DOI: 10.1145/1868470.1868479 (cit. on pp. 68, 69).

[74] Akinori Asahara, Kishiko Maruyama, Akiko Sato, and Kouichi Seto. "Pedestrian-movement prediction based on mixed Markov-chain model." In: *GIS: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems* (2011), pp. 25–33. doi: 10.1145/2093973.2093979 (cit. on pp. 68, 69).

[75] Sébastien Gambs, Marc Olivier Killijian, and Miguel Núñez Del Prado Cortez. "Next place prediction using mobility Markov chains." In: *Proceedings of the 1st Workshop on Measurement, Privacy, and Mobility, MPM'12* (2012), pp. 0–5. doi: 10.1145/2181196.2181199 (cit. on pp. 68, 69).

[76] Paul Baumann, Wilhelm Kleiminger, and Silvia Santini. "The Influence of temporal and spatial features on the performance of next-place prediction algorithms." In: *UbiComp 2013 - Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (2013), pp. 449–458. doi: 10.1145/2493432.2493467 (cit. on pp. 68, 69, 83, 84).

[77] Baichuan Mo, Zhan Zhao, Haris N. Koutsopoulos, and Jinhua Zhao. "Individual Mobility Prediction: An Interpretable Activity-based Hidden Markov Approach." In: (2021), pp. 1–11. arXiv: 2101.03996. url: http://arxiv.org/abs/2101.03996 (cit. on pp. 68, 69, 83, 84).

[78] Abdulrahman Al-Molegi, Mohammed Jabreel, and Baraq Ghaleb. "STF-RNN: Space Time Features-based Recurrent Neural Network for predicting people next location." In: *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016* (2017). doi: 10.1109/SSCI.2016.7849919 (cit. on pp. 68, 69, 83, 84).

[79] Fan Wu, Kun Fu, Yang Wang, Zhibin Xiao, and Xingyu Fu. "A spatial-temporal-semantic neural network algorithm for location prediction on moving objects." In: *Algorithms* 10.2 (2017), p. 37. issn: 19994893. doi: 10.3390/a10020037 (cit. on pp. 68, 70).

[80] Abdessamed Sassi, Mohammed Brahimi, Walid Bechkit, and Abdelmalik Bachir. "Location Embedding and Deep Convolutional Neural Networks for Next Location Prediction." In: *Proceedings - 2019 IEEE 44th Local Computer Networks Symposium on Emerging Topics in Networking, LCN Symposium 2019* (2019), pp. 149–157. doi: 10.1109/LCNSymposium47956.2019.9000680 (cit. on pp. 68–70, 74, 77, 83, 85, 88).

[81] Renhe Jiang, Xuan Song, Zipei Fan, Tianqi Xia, Zhaonan Wang, Quanjun Chen, Zekun Cai, and Ryosuke Shibasaki. "Transfer Urban Human Mobility via POI Embedding over Multiple Cities." In: *ACM/IMS Transactions on Data Science* 2.1 (2021), pp. 1–26. issn: 2691-1922. doi: 10.1145/3416914 (cit. on p. 68).

[82] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. "Predicting the next location: A recurrent model with spatial and temporal contexts." In: *30th AAAI Conference on Artificial Intelligence, AAAI 2016* (2016), pp. 194–200 (cit. on pp. 68, 69).

[83] Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of deep bidirectional transformers for language understanding." In: *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference* 1 (2019), pp. 4171–4186. arXiv: 1810.04805 (cit. on pp. 68–70, 73, 74, 83, 85).

[84] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach." In: 1 (2019). arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692 (cit. on pp. 68, 69, 71, 73, 83, 85, 86).

[85] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao Wuen Hon. "Unified language model pre-training for natural language understanding and generation." In: *Advances in Neural Information Processing Systems* 32.NeurIPS (2019). ISSN: 10495258. arXiv: 1905.03197 (cit. on pp. 69, 71).

[86] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations." In: (2019), pp. 1–17. arXiv: 1909.11942. URL: http://arxiv.org/abs/1909.11942 (cit. on pp. 69, 71).

[87] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. "XLNet: Generalized autoregressive pretraining for language understanding." In: *Advances in Neural Information Processing Systems* 32.NeurIPS (2019), pp. 1–11. ISSN: 10495258. arXiv: 1906.08237 (cit. on pp. 69, 71).

[88] Shijie Geng, Peng Gao, Zuohui Fu, and Yongfeng Zhang. "RomeBERT: Robust Training of Multi-Exit BERT." In: (2021). arXiv: 2101.09755. URL: http://arxiv.org/abs/2101.09755 (cit. on p. 69).

[89] Mike Schuster and Kaisuke Nakajima. "Japanese and Korean voice search." In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2012, pp. 5149–5152. DOI: 10.1109/ICASSP.2012.6289079 (cit. on pp. 69, 73, 83, 85).

[90] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling." In: (2014), pp. 1–9. arXiv: 1412.3555. URL: http://arxiv.org/abs/1412.3555 (cit. on p. 70).

[91] Dongling Xiao, Yu-Kun Li, Han Zhang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. "ERNIE-Gram: Pre-Training with Explicitly N-Gram Masked Language Modeling for Natural Language Understanding." In: (2020). arXiv: 2010.12148. URL: http://arxiv.org/abs/2010.12148 (cit. on pp. 71, 85).

[92] Salah Eddine Henouda, Fatima Zohra Laallam, Okba Kazar, and Abdessamed Sassi. "Next location prediction using transformers." In: *International Journal of Business Intelligence and Data Mining* 21.2 (2022), pp. 247–263 (cit. on p. 73).

[93] David Kotz, Tristan Henderson, Ilya Abyzov, and Jihwang Yeo. *CRAWDAD dataset dartmouth/campus (v. 2009-09-09)*. Sept. 2009. DOI: 10.15783/C7F59T (cit. on pp. 77, 88).

[94] David Kotz and Kobby Essien. "Analysis of a campus-wide wireless network." In: *Wireless Networks* 11.1-2 (2005), pp. 115–133 (cit. on pp. 77, 88).

[95] Tristan Henderson, David Kotz, and Ilya Abyzov. "The changing usage of a mature campus-wide wireless network." In: *Computer Networks* 52.14 (2008), pp. 2690–2712 (cit. on pp. 77, 88).

[96] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. "TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS 1 LSTM: A Search Space Odyssey." In: *arXiv:1503.04069* (2015), pp. 1–11. URL: https://arxiv.org/pdf/1503.04069.pdf (cit. on pp. 77, 88).

[97] Yang Liu, Jian-Wu Bi, and Zhi-Ping Fan. "A method for multi-class sentiment classification based on an improved one-vs-one (OVO) strategy and the support vector machine (SVM) algorithm." In: *Information Sciences* 394-395 (2017), pp. 38–52. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2017.02.016. URL: https://www.sciencedirect.com/science/article/pii/S002002551631043X (cit. on pp. 77, 88).

[98] Thomas Wolf et al. "Transformers: State-of-the-Art Natural Language Processing." In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6 (cit. on p. 78).

[99] W. Liu and Y. Shoji. "DeepVM: RNN-Based Vehicle Mobility Prediction to Support Intelligent Vehicle Applications." In: *IEEE Transactions on Industrial Informatics* 16.6 (2020), pp. 3997–4006. DOI: 10.1109/TII.2019.2936507 (cit. on pp. 83, 84).

[100] Francesco Piccialli, Fabio Giampaolo, Giampaolo Casolla, Vincenzo Schiano Di Cola, and Kenli Li. “A Deep Learning approach for Path Prediction in a Location-based IoT system.” In: *Pervasive and Mobile Computing* 66 (2020), p. 101210 (cit. on p. 83).

[101] Hao Lin, Guannan Liu, Fengzhi Li, and Yuan Zuo. “Where to go? Predicting next location in IoT environment.” In: *Frontiers of Computer Science* 15.1 (2021), pp. 1–13 (cit. on p. 83).

[102] Yuanyuan Qiao, Zhongwei Si, Yanting Zhang, Fehmi Ben Abdesslem, Xinyu Zhang, and Jie Yang. “A hybrid Markov-based model for human mobility prediction.” In: *Neurocomputing* 278 (2018). Recent Advances in Machine Learning for Non-Gaussian Data Processing, pp. 99–109. issn: 0925-2312. doi: `https://doi.org/10.1016/j.neucom.2017.05.101`. url: `https://www.sciencedirect.com/science/article/pii/S0925231217314455` (cit. on pp. 83, 84).

[103] Dejiang Kong and Fei Wu. “HST-LSTM: A Hierarchical Spatial-Temporal Long-Short Term Memory Network for Location Prediction.” In: 18.7 (2018), pp. 2341–2347 (cit. on pp. 83, 84).

[104] Jun Zeng, Xin He, Haoran Tang, and Junhao Wen. “A next location predicting approach based on a recurrent neural network and self-attention.” In: *International Conference on Collaborative Computing: Networking, Applications and Worksharing*. Springer. 2019, pp. 309–322 (cit. on pp. 83, 84).

[105] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benot Sagot. “Camembert: a tasty french language model.” In: *arXiv preprint arXiv:1911.03894* (2019) (cit. on pp. 83, 85).

[106] Guillaume Lample and Alexis Conneau. “Cross-lingual language model pretraining.” In: *arXiv preprint arXiv:1901.07291* (2019) (cit. on pp. 83, 85).

[107] Dat Quoc Nguyen and Anh Tuan Nguyen. “PhoBERT: Pre-trained language models for Vietnamese.” In: *arXiv preprint arXiv:2003.00744* (2020) (cit. on pp. 83, 85).