People's Democratic Republic of Algeria
الجمهورية الجزائرية الديمقراطية الشعبية

Ministry of Higher Education and Scientific Research
وزارة التعليم العالي والبحث العلمي

National Polytechnic School of Oran -Maurice Audin-
المدرسة الوطنية متعددة التقنيات بوهران - موريس اودان -

Department of Computer Systems Engineering

End of Studies Project Thesis For Obtaining Master's Degree in Information Systems

Option: Engineering and Management of Information Systems

Audio Summarizing system based on Transformers and Transfer Learning

Presented By :
Mr. MAHDJOUR Oussama

Supervided By :
Mme. KABLI Fatima (ENPO)
Mme. BOUMEDJOUT Amal (ENPO)

Presented on 20/09/2023 in Front Of The Jury Compose Of :

Mme. BENDIMERAD NAWEL : ENPO - President
Mr. BELBACHIR REDOUANE : ENPO - Examiner

College year: 2023/2022

DEDICATION OUSSAMA

With the expression of my gratitude, I dedicate this work to :

To My Loving Parents, Whose Unwavering Support, Encouragement, and Sacrifices have been my constant source of inspiration throughout my academic journey.

To My Dear Sisters, who have always been there for me, offering their love, guidance, and encouragement.

To My Brother, whose unwavering belief in my abilities has been a driving force behind my successes.

To My Uncles and Aunts, whose guidance and wisdom have shaped my perspective on life and provided valuable advice.

To all my Friends, both old and new, who have walked with me on this path, offering their support, friendship, and laughter. Your presence has made this journey all the more memorable and enjoyable.

I am eternally grateful for your love, support, and encouragement. You have played a key part in my successes, and I am honored to have you in my life.

- Oussama

ACKNOWLEDGEMENT

it seems opportune to start this report with thank God the Merciful, who gave us the strength, to send all my gratitude to my mentor ***Mme. KABLI Fatima*** *for her patience, her availability, and especially her precious advice .*

I would like to take this opportunity to express our sincere appreciation to the faculty and administrative staff of ***the National Polytechnic School of Oran - Maurice Audin*** *as well as all the teachers who have contributed to our education throughout our academic journey, from near or far. My thanks also to all the members of the jury for agreeing to examine our work with patience. My warm thanks go to my dear parents who have always been there for us and who have given me a magnificent model of hard work and perseverance.*

I hope they will find my gratitude, appreciation, and love in this work. We would like to express my gratitude to my friends and colleagues who brought me their moral and intellectual support throughout my process Finally, my most sincere thanks go to all those who have contributed, from near and far, to the improvement of this memory.

Abstract

Abstract :

The proliferation of audio data in various domains necessitates effective methods for distilling meaningful insights from this content. In this thesis, we present an innovative audio summarization system that harnesses the capabilities of transformer-based models and transfer learning. Our approach focuses on simplifying the complex task of audio summarization, making it accessible to a broader audience.

We delve into the world of deep learning, leveraging state-of-the-art transformer models to extract salient information from audio recordings. By fine-tuning these models on domain-specific data, we optimize their performance for the audio summarization task. Our system streamlines the process by providing a user-friendly web application where users can effortlessly upload audio files and obtain both transcriptions and concise summaries with a single click.

Through a comprehensive exploration of the dataset collection, we ensure the relevance and quality of the data used for training and fine-tuning. The resulting model demonstrates remarkable performance in generating coherent and informative audio summaries.

Our work showcases the significance of audio summarization in various applications, from content indexing to enhancing accessibility for individuals with hearing impairments. By providing an end-to-end solution, we contribute to simplifying the utilization of audio data, making it an invaluable resource in today's data-driven world.

Keywords : Audio Summarization, Transformers, Transfer Learning, Deep Learning, Natural language processing,

Résumé :

La prolifération des données audio dans divers domaines nécessite des méthodes efficaces pour extraire des informations significatives de ce contenu. Dans cette thèse, nous présentons un système innovant de résumé audio qui exploite les capacités des modèles basés sur des transformateurs et de l'apprentissage par transfert. Notre approche se concentre sur la simplification de la tâche complexe de résumé audio, la rendant accessible à un public plus large.

Nous plongeons dans le monde de l'apprentissage en profondeur, en tirant parti des modèles de transformateurs de pointe pour extraire des informations saillantes à partir d'enregistrements audio. En affinant ces modèles sur des données spécifiques au domaine, nous optimisons leurs performances pour la tâche de résumé audio. Notre système rationalise le processus en fournissant une application web conviviale où les utilisateurs peuvent télécharger facilement des fichiers audio et obtenir à la fois des transcriptions et des résumés concis en un seul clic.

Grâce à une exploration complète de la collecte de données, nous nous assurons de la pertinence et de la qualité des données utilisées pour la formation et l'affinage. Le modèle résultant présente des performances remarquables dans la génération de résumés audio cohérents et informatifs.

Notre travail met en évidence l'importance du résumé audio dans diverses applications, de l'indexation de contenu à l'amélioration de l'accessibilité pour les personnes malentendantes. En fournissant une solution de bout en bout, nous contribuons à simplifier l'utilisation des données audio, en faisant une ressource inestimable dans le monde axé sur les données d'aujourd'hui.

Mots-clés : Résumé Audio, Transformateurs, Apprentissage par Transfert, Apprentissage Profond, Process des language natural.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

GENERAL INTRODUCTION AND OBJECTIVE

Introduction

The advent of modern technology has witnessed an unprecedented proliferation of digital audio content across various domains, from education and entertainment to information dissemination. The ubiquity of audio data, however, has brought forth a compelling challenge : how to extract meaningful insights and summaries from this vast auditory landscape. Audio summarization, the process of condensing audio content into concise and informative representations, has emerged as a vital endeavor in enhancing the accessibility, comprehension, and utility of audio data.

The burgeoning interest in audio summarization intersects with the transformative advancements in deep learning and machine learning, particularly the transformative potential of transformer-based models and transfer learning. These innovations have unlocked new horizons in natural language processing and understanding, making them ideal candidates for tackling the complexities of audio summarization. In this era of data-driven decision-making, the ability to harness the latent knowledge within audio data can have profound implications across industries and applications.

This thesis embarks on a journey into the realm of audio summarization, guided by the principles of transformer-based models and transfer learning. Our objective is to develop an intelligent system capable of extracting salient information and generating coherent summaries from audio content, with a particular focus on efficiency and accessibility. Through a synergistic fusion of cutting-edge technologies, we aim to empower users with the ability to effortlessly transform audio data into actionable insights.

Objective

In this research, our primary goal is to pioneer an innovative approach to audio summarization by leveraging the power of transformer-based models, transfer learning, and deep learning. Specifically, our objectives are as follows :

- Develop an Advanced Audio Summarization System : Design and implement a sophisticated audio summarization system that leverages the capabilities of transformer-based models, enabling users to efficiently extract key information and generate coherent summaries from audio data.
- Harness Transfer Learning for Audio Understanding : Exploit the advantages of transfer learning to build upon pre-trained models, adapting them to the nuances of audio summarization. This approach allows us to make efficient use of limited data resources.

- Ensure User-Friendly Accessibility : Prioritize user-friendliness and accessibility in system design, creating a web-based application that simplifies the audio summarization process, making it accessible to a broad audience.
- Optimize Data Handling and Workflow : Implement efficient data handling techniques to minimize redundancy and enhance the overall workflow. This ensures that the system operates seamlessly and delivers results swiftly.
- Showcase the Significance of Audio Summarization : Highlight the importance of audio summarization in various domains, from education and journalism to content creation and beyond. Demonstrate the real-world impact of our research.

By accomplishing these objectives, we aim to contribute to the field of audio summarization, bridging the gap between audio data's richness and the need for concise, actionable information.

Contents of Chapters

The subsequent chapters of this thesis are organized to provide a comprehensive exploration of our research :

- **Chapter 1 : Machine and Deep Learning** offers an overview of the foundations of machine learning, deep learning, and their relevance to audio summarization.
- **Chapter 2 : State-of-the-Art** provides a survey of the current state-of-the-art techniques and methodologies in audio summarization, setting the stage for our innovative approach.
- **Chapter 3 : Conception and Implementation** delves into the conception and development of our advanced audio summarization system, detailing its design, workflow, and user-friendly accessibility.

Through this comprehensive journey, we aspire to demonstrate the profound implications of audio summarization and the capabilities of transformer-based models, ultimately advancing the accessibility and utility of audio data in our data-driven world.

CHAPITRE 1

MACHINE AND DEEP LEARNING

1.1 Introduction

In today's fast-paced technological landscape, we are encountering a rapid growth of data-driven applications, primarily due to the increasing amount of data available in various domains and the rapid evolution of machine and deep learning that resulted in the transformation in complex data analysis and decision-making. This chapter covers the basic concepts of the machine and deep learning fields and explains the data mining process and how knowledge extraction from data. Finally presents the most common algorithms and the domains of applications. Overall, this chapter aims to provide an overview of the machine and deep learning and their applications and lay a foundation for the subsequent chapters of this thesis.

1.2 KDD Process

The term KDD (Knowledge Discovery in Databases), has been used since 1989. It is the result of the convergence of research in automatic learning, recognition of forms, databases, statistics, artificial intelligence, and data visualization. There are so many definitions of the KDD process, Han and Kamer (Han and Kamer,2006) explain the KDD process as the process of analysis of the database, often having a large size, to discover unsuspected relationships and summarize the data in an understandable and useful way. According to Fayyad in 1996 KDD is the process of extracting knowledge based on identifying unknown patterns that can be exploited in the database, this process aims to transform a large data, in multiform, stored in different formats and gathered from different places into usable knowledge that can be used in decision making. KDD process model consists of a set of steps to follow ; He takes data assets and provides new knowledge as final outings. The stages of the KDD process, are illustrated in the following figure :

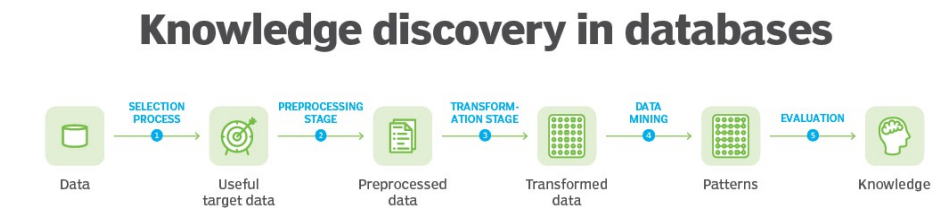


FIGURE 1.1 – KDD-Process
[2]

1.2.1 Data Selection and Extraction

This step consists of two steps :

1.2.1.1 Understanding the domain of application

This step includes understanding the domain of application, the discovery of relevant previous knowledge, and the definition of the objective of the KDD process.

1.2.1.2 The creation of the target data set

This step includes selecting a data set or concentration on a subset of variables or data samples. In addition, it includes data integration if the data is of different heterogeneous resources. The final output of this step is the set of data on which the discovery must be made.

1.2.2 Data Preprocessing and Cleaning

This step includes deleting noise or aberrant values, collecting the information necessary to model or process noise, use of previous knowledge to delete inconsistencies and duplicates from data, choosing or use of strategies to manage missing data fields, etc. Most of the data collected is incomplete and inconsistent, which leads to confusing the rest of the process and leads to unreliable and non-valid results. For this, this step must be established to improve the quality of the data and arrive at a good result as a final outing. Here are some of the basic steps in data pre-processing :

1.2.2.1 Handling missing values

This can cause a really big problem in data manipulation and model training because the NaN values occupy a space in memory that have no value which can't be processed by the computer, this problem also accrues while handling infinite values. To solve this problem, we have two approaches either by dropping the rows that contain missing values if the dataset is large enough or we can generate new data to fill the missing value using the statistical approach

1.2.2.2 Handling categorical data

There are two types of data : qualitative data which can represent categories and have groups or classes and quantitative data which represents numerical values Most machine learning algorithms can't process categorical data so in this step we need to encode categorical data into numerical.

1.2.2.3 Handling outliers

Outliers are the values that fall off the extreme of the value distribution that represents rare cases and that can cause errors in the algorithm to overfit on rare cases therefore to ensure the best results we need to handle those outliers

1.2.2.4 Normalization of the dataset

The value of the dataset can be large numbers and this can take a lot of processing power and time, to solve this problem we need to normalize the data this can be done by statistical tools

1.2.2.5 Handle Dataset imbalances

Often the dataset has an imbalance in the number of instances in target classes that can cause to model to have a bias toward the majority classes while neglecting minority classes to solve this problem we have two methods :

Under sampling : Using different algorithms, we can reduce the size of the majority classes to have a balanced ratio with the minority classes, this approach is optimal with large datasets but can reduce the results of the models when applied to small datasets.

Data Augmentation : Using different algorithms, we can generate new data in the minority classes to augment the ratio with the majority classes

1.2.2.6 Correlation analyses

The machine learning algorithm can struggle with large sets of features that can drain an unnecessary number of resources so it is optimal to analyze the relationship between variables and combine strongly related variables

1.2.3 Data Transformation and Reduction

This step includes the search for useful features to represent the data, depending on the objective of the process. More specifically, processing techniques are used in this step to make data more appropriate for exploitation such as aggregation and normalization ... and reduction techniques are applied to produce a reduced representation of data such as aggregation and dimension reduction.

1.2.4 Data Mining and Pattern Extraction

It is the application of a group of different methods on ready-to-use data. Due to the great diversity of the data used, the result is a large number of data excavation methods. The latter come from various fields such as statistics, data analysis, automatic learning, etc. In addition, some of these methods can be combined to reduce the disadvantages of one or the other. The choice of data excavation methods depends on the one hand, the needs expressed by the user and on the other hand, the used data.

1.2.5 Pattern Evaluation and Interpretation

This phase consists of the evaluation of the model, which measures the performance of the model on many different levels and then presents the results to the user using different visualization techniques.

1.3 Machine learning

Machine learning is a sub-field of artificial intelligence that focuses on extracting knowledge from data. It is a field based on the fusion of statistics and computer science to create systems and models that learn and improve based on the data the process

1.3.1 Definition

The term “Machine learning” was defined in 1959 by IBM employee “Arthur Samuel” [3] as the following : “Machine learning is the discipline giving computers the ability to learn without being explicitly programmed”. Machine learning algorithms create models by training the algorithms on large amounts of data to identify patterns and relationships and then use them to make predictions or decisions on new data. Those models enable computers to automatically improve their performance on a task by learning from experience.

1.3.2 Types of Machine Learning

Machine learning algorithms can be divided into three main types : supervised learning, unsupervised learning, and reinforcement learning

1.3.2.1 Supervised Learning :

Supervised learning is a type of machine learning where the algorithm trains on labeled data where the desired output or target is known for each input data point, and then it makes predictions on new, unseen data. the algorithms of supervised learning are the most widely used, it exists two main categories in supervised learning : regression and classification.[4]

Classification : In classification, the algorithm predicts a categorical label or class, such as "spam" or "not spam". The output variable is qualitative.

Regression : In regression, the algorithm predicts a continuous value, such as a price or a temperature. The output variable is quantitative.

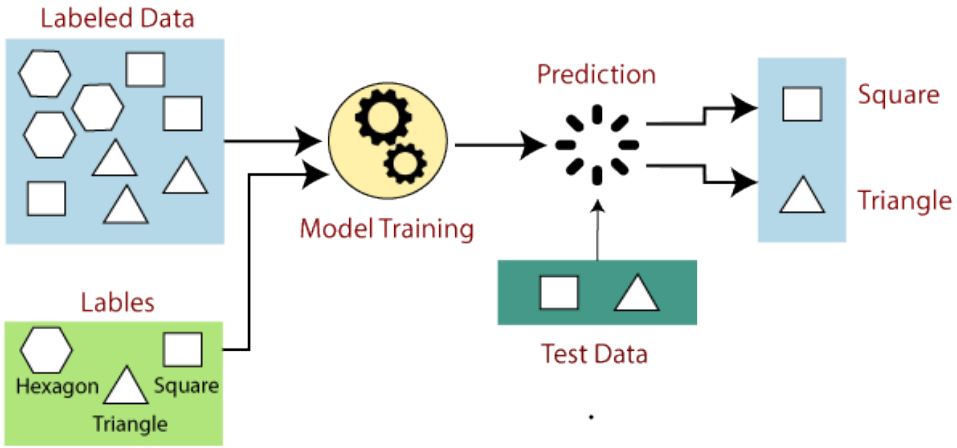


Figure 1.2 – illustration of supervised machine learning
[5]

A. Linear Regression Linear regression is a regression-type of supervised learning algorithm capable of establishing a linear relationship between a dependent variable, and

one or more independent variables. The main goal is to fit a better line that comes as close as possible to a set of points, represented by a linear equation.[6]

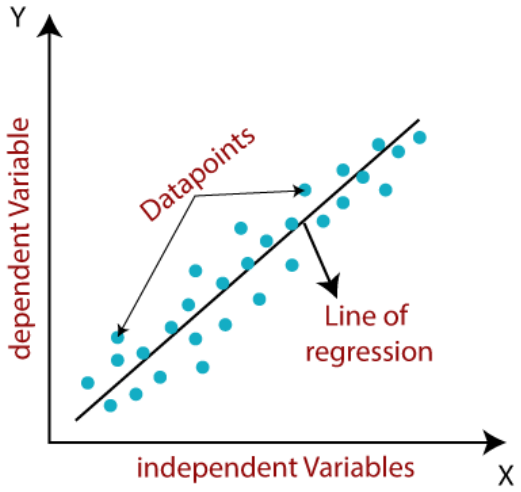


FIGURE 1.3 – illustration of Linear Regression Architecture
[7]

B. Polynomial Regression Polynomial Regression is a regression algorithm that models the relationship between a dependent(y) and independent variable(x) as an nth-degree polynomial, this considers an upgrade to the linear regression algorithm so it can have better results on complex data.

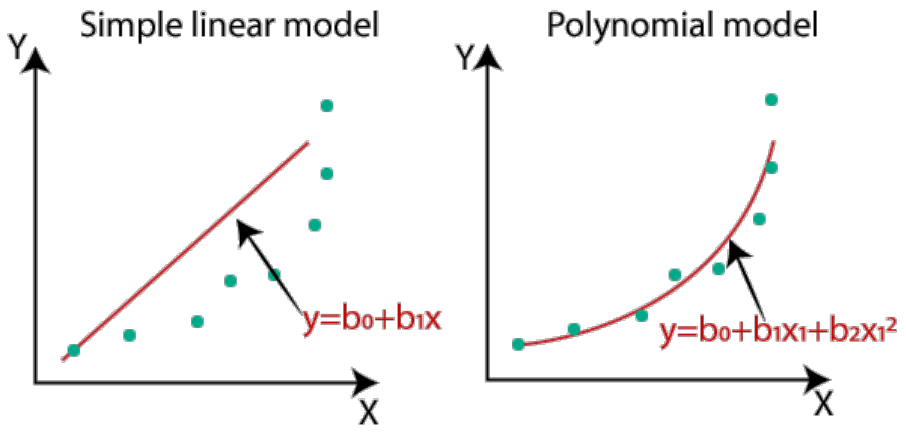


FIGURE 1.4 – illustration of Poly-nominal Regression Architecture
[8]

C. Logistic Regression : Logistic regression is one of the most known supervised learning methods ; this type of algorithm uses the logistic function on the independent variables to predict the dependent categorical variable. logistic regression is a form of regression that is used for classification where the output is the probability of the class.

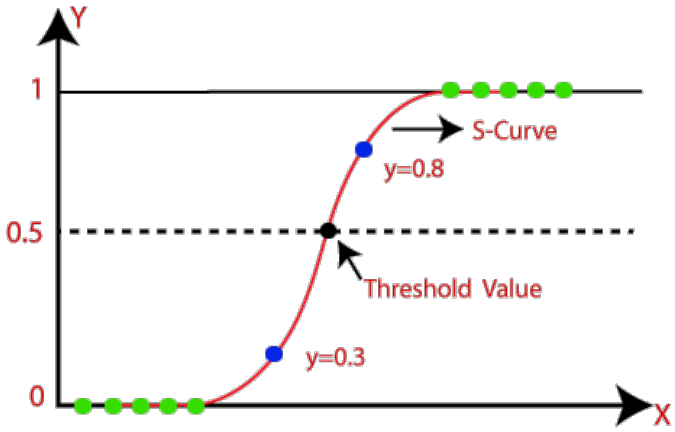


FIGURE 1.5 – illustration of logistic Regression Architecture
[7]

D. K-Nearest Neighbours (KNN) The KNN algorithm is one of the simplest supervised learning algorithms. It aims to classify each new data with the use of a distance metric such as Euclidean distance.

The simple founding idea is to start from a labeled database, we can estimate the class of new data by looking at what is the majority class of the k nearest neighboring data (hence the name of the 'algorithm). The only parameter to fix is k, the number of neighbors to consider.

This algorithm can be used for Regression as well as for Classification but mostly it is used for Classification problems.

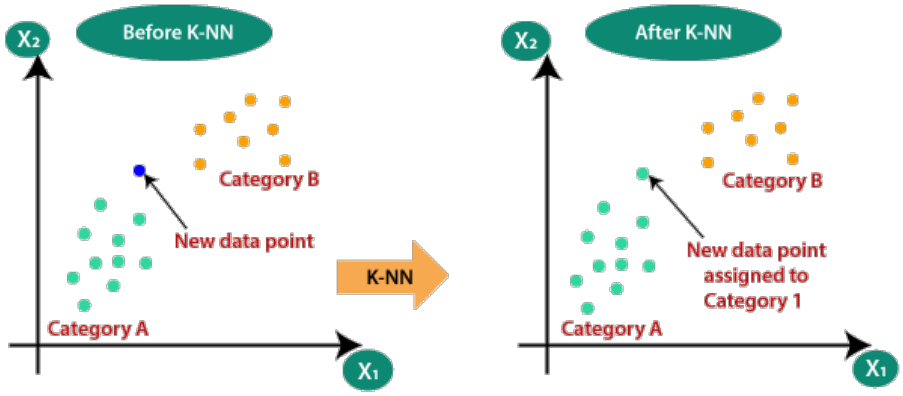


FIGURE 1.6 – illustration of KNN Architecture
[9]

Algorithm 1: K-Nearest Neighbor Algorithm

Input: i : number of instances in the training dataset ($i > 0$), C : Class of i sample, $k > 0$, D : Training Dataset, $Dist$: Function of calculation distance, X : example to classify

Output: Y : the label of the instance X

$KNN(X) \leftarrow \{\}$
for $(i, C) \in D$ **do**
    `// Calculate the distance`
    $KNN(X) \leftarrow Dist(i, X)$
end
$Z \leftarrow \{\}$
for $i \in KNN(X)$ **do**
    `// Calculate the number of occurrences of each class` C `in` Z
    $Z \leftarrow$ Occurrences of class C in i
end
$y \leftarrow \max(Z)$
Result: $Y \leftarrow y$

E. Naive Bayes A probabilistic model that calculates the probability of each class given the input features using Bayes' theorem, it is one of the simple and most effective Classification algorithms which helps in building fast machine learning models that can make quick predictions. This algorithm is a probabilistic classifier, which means it predicts on the basis of the probability of an object, as the name suggests this algorithm uses the Bayes rule which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability according to the following formula

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1.1}$$

P(A|B) represents the Probability of hypothesis A on the observed event B.
P(B|A) represents the Probability of the evidence given that the probability of a hypothesis is true.
P(A) represents the Probability of the hypothesis before observing the evidence.
P(B) represents the Probability of Evidence.

F. Support Vector Machine (SVM) Support vector machine is a popular algorithm of machine learning that was developed in the 1990s, it is capable of performing linear and non-linear classification and regression. the principle of this algorithm is to seek a hyperplane or border which best separates two classes of data by guaranteeing that the margin between the two classes is maximum. Multiple planes can separate the two classes, but a single plane can maximize the headroom or distance between classes.

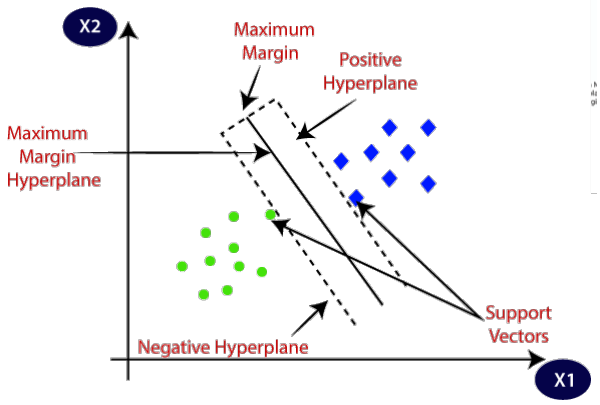


FIGURE 1.7 – illustration of SVM Architecture
[10]

G. Decision Tree Decision trees are supervised learning models, which can be used for classification as well as regression, they have several algorithms known as ID3 (Quinlan 1986) and C4.5 (Quinlan 1993), which generate a tree structure where each node internally refers to a test on an attribute. Each branch represents the test result and each leaf node contains a decision which is a unique value. Inputs and outputs can be discrete or continuous.

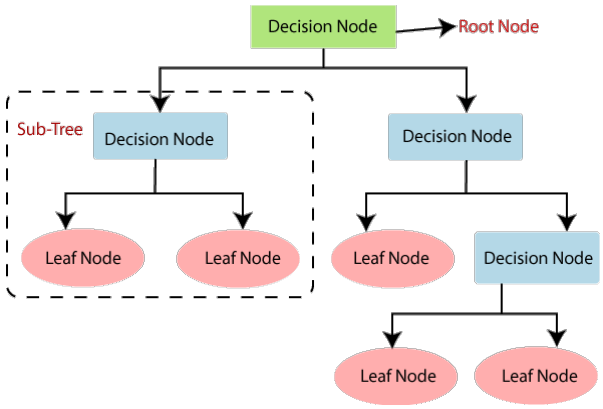


FIGURE 1.8 – illustration of Decision tree Architecture
[11]

H. Ensemble learning Ensemble learning is a group of techniques of machine learning called meta-algorithms that consists of combining a group of unique models to optimize the performance and have better generalization. ensemble learning methods are divided into two main categories based on the nature of the combination : bagging (parallel training) and Boosting (sequential training).

Bagging The Bagging ensemble technique is the acronym for "bootstrap aggregating" and is one of the earliest ensemble methods proposed. It consists of creating subsamples from a dataset called "bootstrap sampling". Then train several Machine Learning models on those subsets to create independent models called weak learners. During test time, the predictions from all weak learners are accounted for.

Boosting The boosting ensemble mechanism works in a way slightly different from the bagging mechanism. where, instead of parallel processing of data, sequential processing of the dataset occurs. The first weak learner is fed with the entire dataset, and the predictions are analyzed. The instances where the previous weak learner fails to produce correct predictions will be fed to the next weak learner. This is done so that the next weak learner can specifically focus on the problematic areas and correct the errors made by the previous weak learner. Similarly, further steps of the same idea are employed, and then the ensemble of all these previous classifiers is computed to make the final prediction on the test data.

I. Random Forest Random Forest is a popular ensemble-based supervised machine learning algorithm used for both Classification and Regression problems in machine learning. Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.[12]

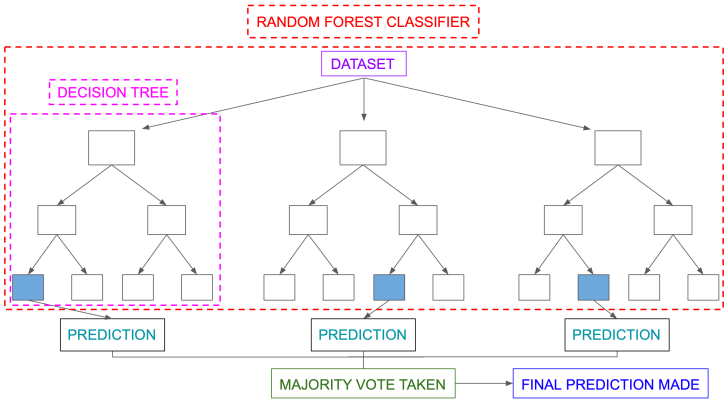


FIGURE 1.9 – illustration of Random Forest Architecture [13]

J. Adaptive boosting (Adaboost) AdaBoost (Adaptive boosting), proposed by "Freund and Schapire" in 1996, was the first boosting algorithm to combine various weak classifiers into a single strong classifier in the history of machine learning. Each time a basic estimator is trained, the previously misclassified instances are weighted with a higher weight, with the aim that during the next iterations, the new models correct the errors of the previous models, which should improve the overall performance.[14]

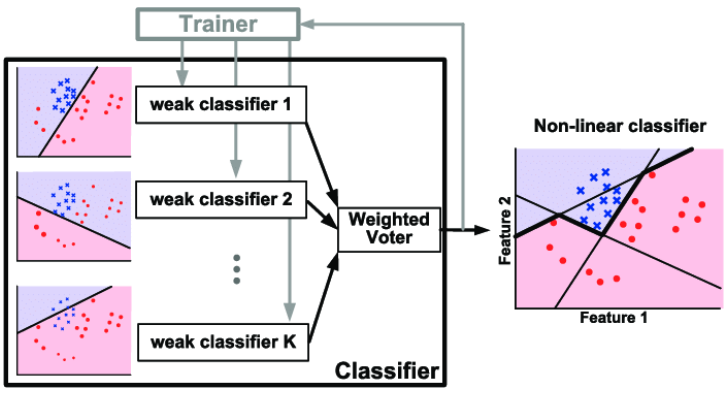


FIGURE 1.10 – illustration of ADA Boost Architecture [15]

K. gradient boosting machine (GBM) Gradient boosting machine (GBM) is one of the most popular sequential ensemble learning algorithms, GBM can be used for creating predictive models for classification or regression problems. The GBM algorithm is similar to the Adaboost algorithm with one key difference in how the model is being updated after incorrect prediction, It works on the principle that many weak learners can train a more accurate model, where each model fits the residual from the previous step to improve the model using the Gradient Descent algorithm, the residual is the gap between prediction and reality and can be calculated using different loss functions for different problems, for example, we can use mean absolute error for regression problems or we can use log loss function for classification problems

L. Extreme Gradient Boost (XGBoost) XGBoost is an evolutionary ensemble-based supervised learning system that is used to reinforce decision tree-based models such as random forest, it was proposed by Chen and Guestrin in 2016. Their principal is to combine the results of weak simple models to produce butter prediction, contrary to the random forest algorithm this model uses the boosting method for their ensemble learning. their principle of sequential auto-optimization includes the use of a large number of hyper-parameters to minimize the loss function and it consists of three elements :

- optimized loss function
- weak learner model
- An additive model to combine our weak learners to minimize the loss function

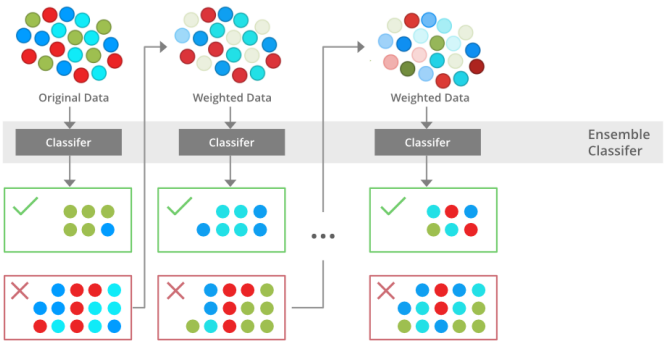


FIGURE 1.11 – illustration of XGBoost Architecture [16]

M. LigtGBM Algorithm Introduced by Microsoft in January 2017, the Light Gradient Boosting Machine (LightGBM) represents an enhanced version of the Gradient Boosting Machine (GBM). This model improves on the conventional decision tree algorithm by using a gradient-based one-side sampling (GOSS) approach and an exclusive feature bundling (EFB) technique. LightGBM was designed to be more efficient, consuming less memory and delivering improved performance, especially for large-scale data processing.[16]

Contrary to the traditional level-wise growth of decision trees, LightGBM implements a leaf-wise growth strategy. After the initial split, subsequent divisions are carried out only on the leaf node that incurs the highest loss. This unique approach aids in reducing loss more rapidly, contributing to the algorithm's efficiency.

N. CatBoost Algorithm Developed by Yandex in 2017, the CatBoost algorithm—derived from the term 'Categorical Boosting'—is particularly adept at handling categorical va-

riables in data. This model transforms categorical variables into numerical ones by applying various statistical techniques on feature combinations.

Many machine learning algorithms struggle to directly process data represented in strings or categories. Hence, the conversion of categorical variables into numerical values becomes an essential preprocessing step. The CatBoost algorithm excels in this conversion, thus eliminating the need for explicit preprocessing. [16]

The distinguishing feature of CatBoost is its capacity to handle categorical data directly, enabling it to maintain a high level of accuracy even when dealing with data that has numerous categorical features.

1.3.2.2 Unsupervised Learning

unsupervised learning consists of building a model from unlabeled input data. To do this, the system will cross-reference the information submitted to it, to be able to group in the same class the elements with certain similarities.

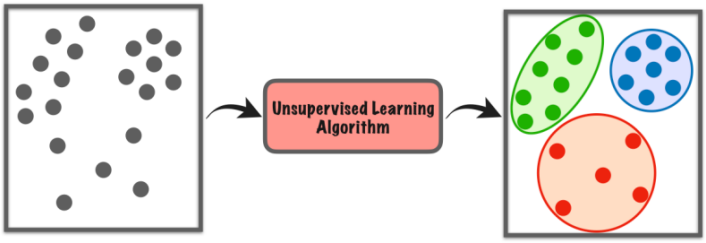


FIGURE 1.12 – illustration of unsupervised learning
[17]

there are three main types of unsupervised learning :

A. Clustering Clustering is the method of statistical analyses used to group the data points from unlabelled datasets into homogeneous Clusters with each cluster consisting of similar data points, with the object with similarities in the same group, most of the clustering algorithms work by finding similar patterns in the data points like shape or colors. the clustering algorithms can be divided into two main groups :
Hard Clustering : were each data point can only exist in a single cluster, and **Soft Clustering :** where each data point can exist in multiple clusters

B. Dimensionality reduction Dimensionality reduction is a technique used to reduce the complexity of data by reducing the number of features or variables while preserving the important information contained in the data. Its main goal is to simplify the data without sacrificing crucial information. This is usually achieved by grouping similar features together or by transforming the original features into a new set of features that represent the original data more efficiently.

Dimensionality reduction is widely used in various fields that handle high-dimensional data, such as speech recognition, signal processing, image processing, and machine learning. The reduction in dimensionality makes the data more manageable, easier to analyze, and often leads to improved performance of machine learning algorithms. Additionally, dimensionality reduction can help to eliminate noise, redundancy, and irrelevant information, which can improve the accuracy and efficiency of data analysis.

C. Association rules The algorithms used for association rule mining belong to the family of unsupervised learning methods. These methods enable the identification of patterns within a group of transaction data, which can then be used to generate rules representing possible associations between different items. The resulting association rules can help to identify correlations and co-occurrences between datasets and are particularly useful for explaining patterns in seemingly independent information repositories, such as relational and transactional databases. Association rule mining is a powerful tool for discovering hidden patterns and relationships within large datasets and can provide valuable insights for decision-making in a variety of fields.

1.3.2.3 Reinforcement learning

Reinforcement learning (RL) is a distinct type of machine learning that differs from other approaches in that it involves a multi-agent system. In RL, the learning process is based on successive experiences, where an agent interacts with an environment by taking actions in order to find an optimal solution based on a reward system, with the ultimate goal of finding the best strategy.

RL is particularly useful in complex, dynamic environments where it may be difficult to model all possible scenarios. By learning from experience, RL algorithms can adapt to changing environments and find optimal solutions. Examples of RL applications include robotics, game-playing, and autonomous vehicle control.

1.4 Deep learning

Deep learning is the newfound sub-field of machine learning, it is the technological key component of today's revolutionary AI application that makes it possible to perform tasks that were still unthinkable a few years back such as chat-bots that can generate human-like written text like ChatGPT or a generative AI that can generate an image starting with a descriptive prompt. The success of deep learning is based both on the technical evolution of computing capability, and the huge advancement in information modeling and representation in statistical learning methods.

1.4.1 Definition of deep learning

It is the newest field of research in machine learning, and it was created to surpass the limitations of machine learning with data pre-processing and feature extraction. deep learning algorithm inspired by the power of the human brain and how it can process large data in a short time, this resulted in deep learning algorithms having better performance on large data than classical machine learning algorithms Deep learning algorithms are based on one key competent called artificial neural networks which are curated to handle large data without any problems Deep learning models surpass machine learning models with their ability to extract important features from raw data without human intervention, this is due to the nature of the neural network and its deep layers that compose a group of linear or nonlinear equations those feature than the process through deep hidden layers to then finally generate the final output of the model [18][19][20][21]

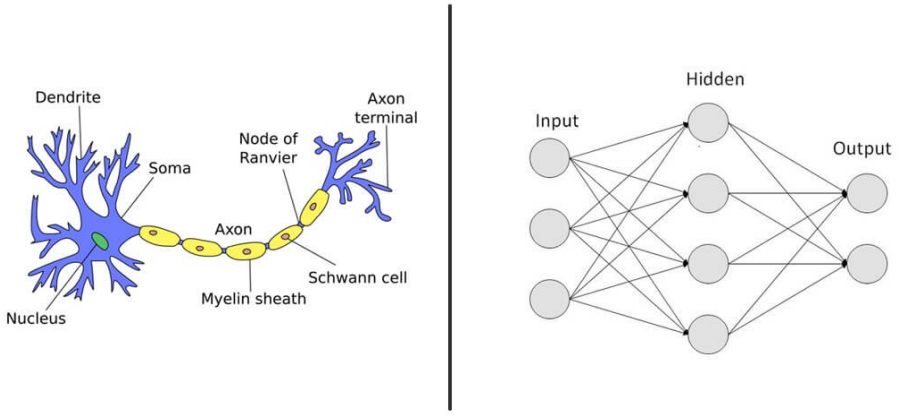


FIGURE 1.13 – illustration of Neural Network and Biological Neural [22]

1.4.2 Why Deep Learning ?

with the rise of data-driven applications and the technical advancement in electronics, more data have been stored and processed than enabled machine learning algorithms to solve imaginable problems but due to the nature of machine learning algorithms they need human intervention to help process the data and eliminate unnecessary features to have good performance, this intervention limited the advancement of AI application to a certain domain of application where the data can be pre-processed by humans efficiency. The deep learning algorithm, on the other hand, came to solve this problem and help surpass the machine learning limitation by mimicking the human brain structure with artificial neural networks this structure can extract important features from given information without any human intervention needed, and this opened up a lot more domain of application for AI where it was impossible to achieve with traditional machine learning algorithms like computer vision and AI chatbots. Due to the nature of deep learning algorithms, it needs a large set of data to extract feature and outperform machine learning, studies show that machine learning algorithms outperform deep learning algorithm in small data but when the data is increased the machine learning algorithm stops improving while deep learning algorithms outperform them, with amount data large enough machine learning algorithms can outperform humans in problems like image processing and pattern recognition [23]

1.4.3 Neural networks

Deep learning algorithms consist of artificial neural networks called ANN, which are models that treat information inspired by the structure of biological neural systems. It is similar to the way the human brain processes information. all neural networks consist of an ensemble of interconnected nodes called layers.

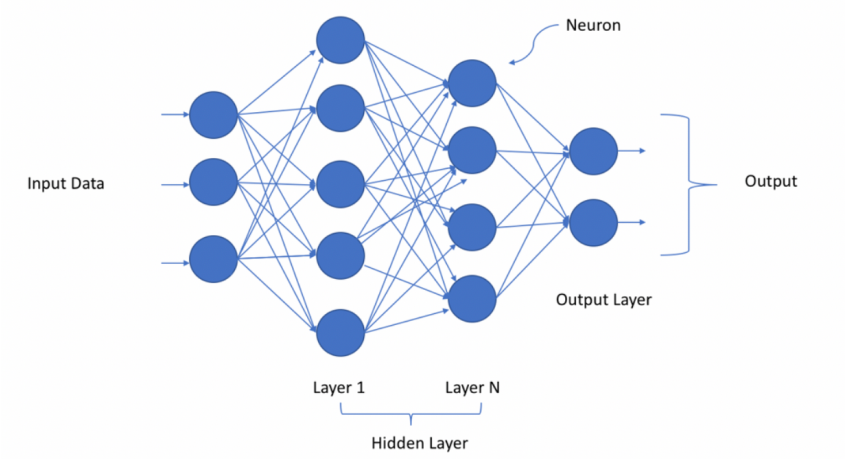


FIGURE 1.14 – illustration of Artificial Neural Network
[24]

1.4.3.1 Activation functions

They are key components of the neural network, they are mathematical functions that are applied in the calculation process of the value of each node in the layers, this value is then inputted into the next or previous layer as input depending on the complexity of the network, this function helps to optimize the learning process by assist in the weight updating process of each node. activation functions are usually nonlinear functions necessary for solving complex problems within the neural network. Therefore a lot of research has been done to create optimal activation functions here are some examples of widely used activation functions out there :[25] [26]

A. Sigmoid This function is one of the most commonly used activation functions, also known as the logistic function, or logistic sigmoid, this function's output is ranged between 0 and 1 where this output represents the stochastic probability of node activation. Due to the nature of this function, it is primarily used for binary classification problems.

B. Rectified Linear Function (ReLu) This is one of the fastest activation functions used in building deep learning models. This function is one of the most popular functions to be used in hidden layers because of its optimized performance and the speed of calculating the desired output, this function is a combination of two functions.

C. Softmax softmax function is a type of regression function that is a generalization of logistic regression primarily used for multi-classification problems, unlike other activation functions softmax function the output of node in this layer dependent on the output of all the other nodes exist in this function which assures that the sum of all the nodes outputs equals to 1.

D. Hyperbolic Tangent Functions (tanH) tanH function is a superior function compared to other widely used activation functions like the sigmoid function, However, it takes less account of the relationships and it is slower to converge.

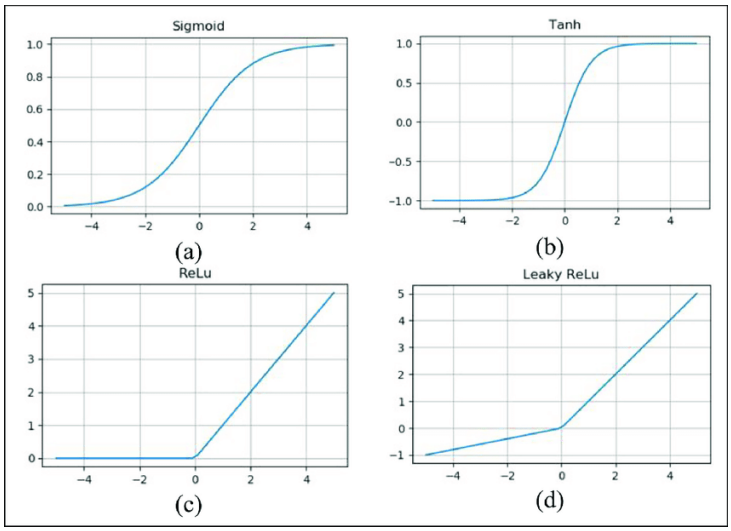


FIGURE 1.15 – illustration of Activation Functions [27]

1.4.3.2 Back-propagation

Is the principle of updating the weight of each node to minimize the loss ; this process is the key component in the power of deep learning algorithms. This process is dependent largely on the choice of the loss function that is optimal for the problem, this process is controlled with a parameter called learning rate which measures the intensity of the changes in the node's weights.

1.4.3.3 Types of neural networks

The structure of neural networks depends on the problems we want to solve. Still, we can define three main classes of neural networks :

A. Single-layer perceptron (feed-forward) This form is the simplest example of a neural network, in this form, we have only two layers : an input layer and an output layer. where the propagation of the information is linear from the input where the information enters the network to pass by the activation function then it outputs via the output layer as a system response. This structure is simple as there is no returning connection or cross-connection in the output layer.[28]

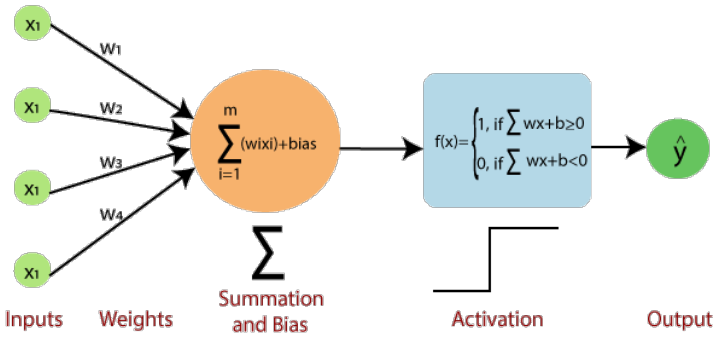


FIGURE 1.16 – illustration of Single-Layer perceptron [29]

B. Multi-layer perceptrons (feed-forward) Multi-layer perceptrons, also called MLP (Multi-layer Perceptron), are more general neural networks than single-layer perceptrons. This type of network consists of one or more hidden layers, whose computational nodes are called hidden neurons or hidden units. The function of hidden neurons is to interact in a useful way between the external input and the network output and to extract higher-order statistics. The source nodes of the input layer of the network provide the input signal to the neurons of the second layer (1st hidden layer). The output signals from the second layer are used as inputs to the third layer and so on. The set of output signals from the neurons in the output layer of the network constitutes the global response of the network to the activation model provided by the source nodes of the first input layer.[30]

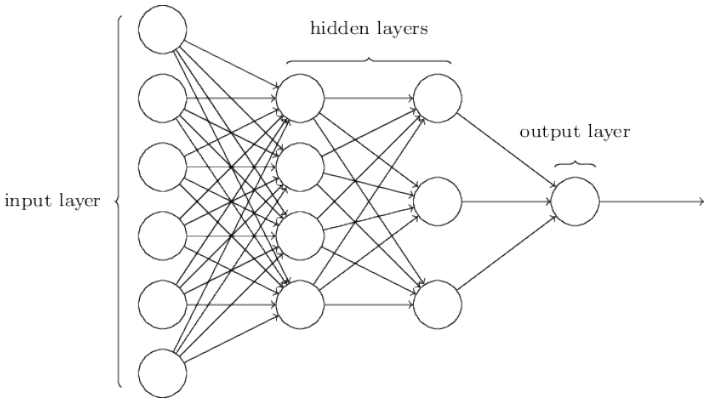


FIGURE 1.17 – illustration of Multi-Layer perceptrons
[31]

C. Recurrent deep neural network (feed-backward) A recurrent network differs from the previous ones in that it is cyclic : comprising one or more hidden layers with at least one feedback loop. The presence of cycles has a profound impact on the learning capacities of the network and on its performance, in particular making the system dynamic, which leads to a nonlinear dynamic behavior.[30][32]

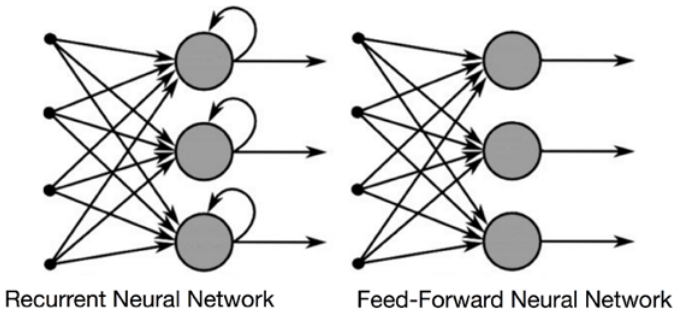


FIGURE 1.18 – illustration of Recurrent Neural Network
[33]

1.4.4 Architectures

Deep learning is based on multi-layered artificial neural networks. This technology has allowed scientists to solve complex problems and advance the field of data science and AI to a whole new level. These multi-layer neural networks can comprise millions of neurons, distributed in several tens of layers. They are used in deep learning to design supervised and unsupervised learning mechanisms. the modification in the different parameters of

the neural network such as the type of each layer and the way those layers are stacked up against each other results in different network architectures that can be used to solve various problems. here are some of the most widely used network architectures in the field of deep learning

1.4.4.1 Convolutional Neural Networks (CNN)

Convolutional neural networks are a type of specialized neural network for data processing having a grid-like topology. CNNs have had considerable success in practical applications. We can then define a CNN as a neural network that has one or more convolution layers in the network architecture. A nonlinear activation function is then applied to the output of these convolutions and the convolution process. The name "convolutional neural network" indicates that the network employs a mathematical operation called convolution. Convolution is a special linear operation. Convolutional networks are simply neural networks that use convolution instead of matrix multiplication in at least one of their layers. They have wide applications in image and video recognition, recommendation systems and natural language processing, etc.

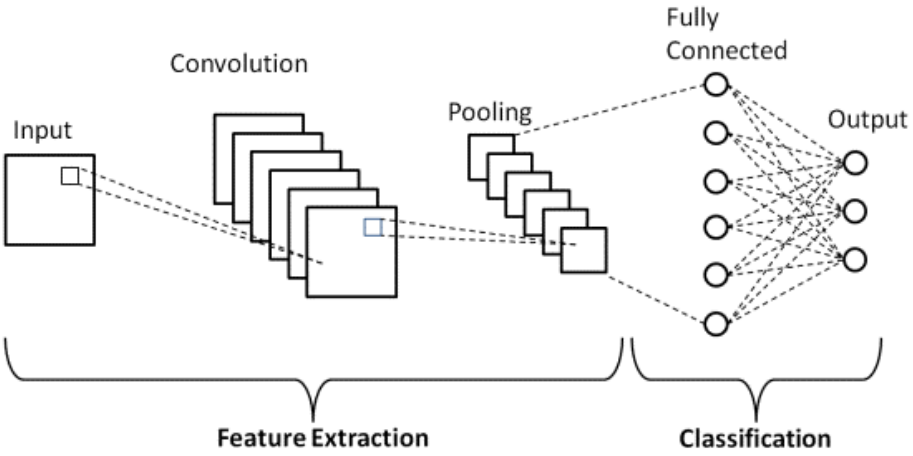


FIGURE 1.19 – illustration of Convolutional Neural Network Architecture [34]

1.4.4.2 Recurrent Neural Networks (RNN)

The recurrent neural network is based on the principle of saving the output of a layer and feeding it back to the input to help predict the outcome of the layer. The first layer is trained similarly to the feed-forward neural network with the product of the sum of weights and features. The recurrent neural network process starts once this is calculated, each neuron will remember some information that it had at the previous time step. This causes each neuron to act as a memory cell during calculations. In this process, we have to let the neural network work on forward propagation and remember the information it needs for later use. RNNs are called recurrent because they perform the same task for each element of a sequence, with the output dependent on previous computations.

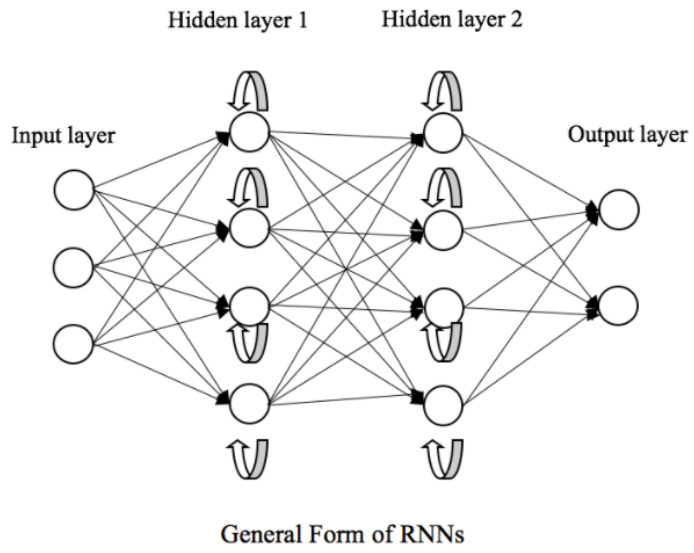


FIGURE 1.20 – illustration of Recurrent Neural Networks architecture [35]

1.4.4.3 Generative Adversarial Networks (GAN)

GANs for Generative Adversarial Networks. GANs are an example of a network that uses unsupervised learning to train two models in parallel. The first model is the generator. It generates a sample and tricks the discriminator when it generates data, on the other hand, the second network (adversary, discriminator), tries to detect if a sample is real or if it is the result of the generator. Its purpose is to learn to differentiate between these two types of data.

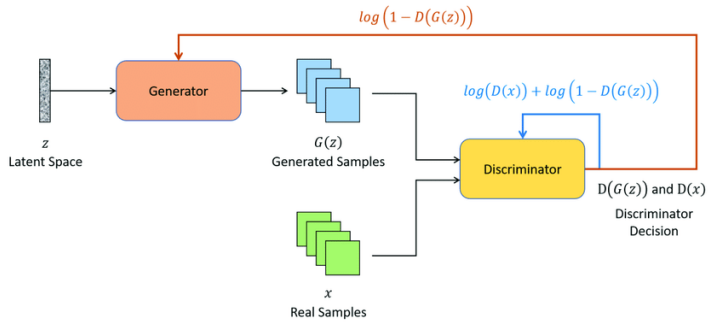


FIGURE 1.21 – illustration of GAN Architecture [36]

1.4.4.4 Automatic encoders (Auto Encoder) :

Auto-encoders (AEs) are a type of artificial neural network that can learn to compress data and then reconstruct it with minimal loss of information. The AE structure consists of two parts : the encoder and the decoder.

The encoder compresses the input data into a lower-dimensional representation, while the decoder reconstructs the original input from the compressed representation. The encoder and decoder are trained together to minimize the difference between the input and the output.

AEs have various use cases, including data compression, image denoising, image colorization, anomaly detection, and dimensionality reduction. For example, in data compression, the encoder compresses the data into a smaller representation, reducing the

storage requirements. In anomaly detection, the AE is trained on normal data, and when an anomaly is detected, the reconstruction error is higher, indicating that the input is not normal. In dimensionality reduction, the AE learns a low-dimensional representation of high-dimensional data, which can be useful for visualization or for reducing the complexity of a problem.

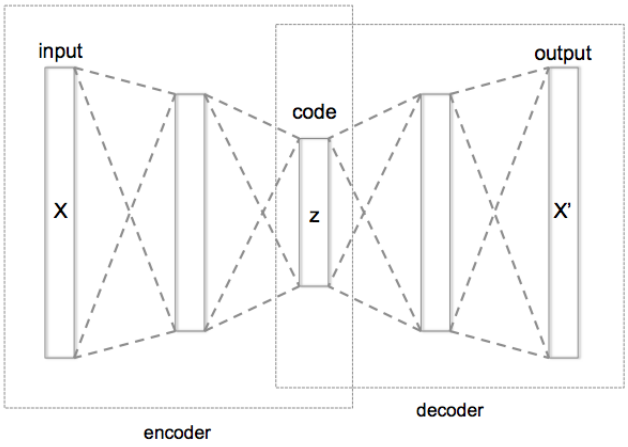


FIGURE 1.22 – illustration of AutoEncoder architecture
[37]

1.4.5 Transformers

The Transformer architecture, introduced in the seminal paper "Attention Is All You Need" by Vaswani et al. [38], has revolutionized the field of deep learning, particularly in the context of sequence-to-sequence tasks. Transformers are characterized by their ability to capture long-range dependencies in sequences efficiently, making them well-suited for various natural language processing (NLP) tasks and beyond.

1.4.5.1 Model Architecture

The Transformer architecture follows the encoder-decoder paradigm commonly used in sequence transduction models. In this setup, the encoder maps an input sequence of symbol representations $(x_1, \ldots, x_n)$ to a sequence of continuous representations $(z_1, \ldots, z_n)$. Given z, the decoder generates an output sequence $(y_1, \ldots, y_m)$ autoregressively, meaning it generates each symbol based on previously generated ones. The Transformer architecture comprises stacked self-attention and point-wise, fully connected layers in both the encoder and decoder, as illustrated in this Figure.

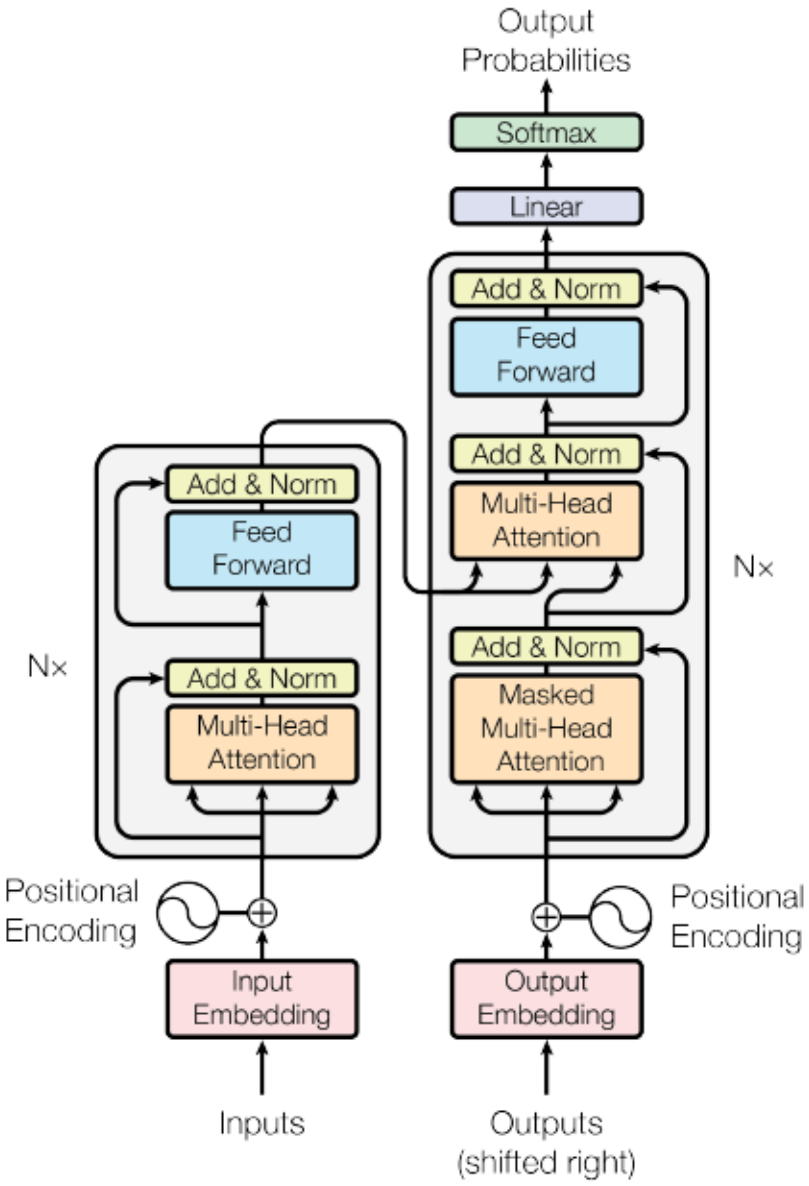


FIGURE 1.23 – The Transformer model architecture.
[38]

1.4.5.2 Encoder and Decoder Stacks

The encoder consists of a stack of $N = 6$ identical layers, each with two sub-layers. The first sub-layer implements multi-head self-attention, while the second sub-layer is a position-wise fully connected feed-forward network. Residual connections [39] and layer normalization [40] are applied around each sub-layer. Each sub-layer's output has a dimension of $d_{\text{model}} = 512$.

The decoder also comprises a stack of $N = 6$ identical layers. In addition to the two sub-layers found in the encoder, the decoder includes a third sub-layer, performing multi-head attention over the encoder stack's output. Similar to the encoder, residual connections and layer normalization are employed. Importantly, the self-attention sub-layer in the decoder prevents positions from attending to subsequent positions, ensuring the auto-regressive property of the model.

1.4.5.3 Attention

The Transformer uses scaled dot-product attention, a critical component for capturing dependencies in the input sequence. The input consists of queries, keys, and values, each with dimensions d_k, d_k, and d_v, respectively. The attention mechanism computes the dot products of queries and keys, scales them by $\sqrt{1/d_k}$, applies a softmax function to obtain weights, and then computes a weighted sum of the values.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1.2}$$

The Transformer employs multi-head attention, which allows the model to attend to different parts of the input sequence simultaneously. Instead of using a single attention head with d_{model}-dimensional keys, values, and queries, the model linearly projects them h times into d_k, d_k, and d_v dimensions, respectively, and runs attention in parallel.

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \tag{1.3}$$

Each attention head head_i operates on the projected queries, keys, and values and produces d_v-dimensional output values. The final values are obtained by concatenating the outputs of all heads and projecting them using W^O. In this work, $h = 8$ parallel attention heads are used, with $d_k = d_v = d_{\text{model}}/h = 64$.

1.4.5.4 Positional Encoding

Since the Transformer lacks recurrence or convolutional layers, it relies on positional encodings to incorporate sequence order information. Positional encodings are added to the input embeddings to indicate the position of each token in the sequence.

The chosen positional encoding method involves using sine and cosine functions of different frequencies to create positional embeddings. These embeddings are summed with the input embeddings to combine positional information with token representations. Specifically, for position *pos* and dimension i, the positional encoding is given by :

$$\text{PE}(pos, 2i) = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \tag{1.4}$$

$$\text{PE}(pos, 2i+1) = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \tag{1.5}$$

This approach ensures that the model can capture positional information effectively without the need for recurrence or convolution.

The Transformer architecture, with its self-attention mechanism and multi-head attention, has significantly improved the state-of-the-art in various NLP tasks. Its success has extended to other domains beyond NLP, making it a versatile and powerful tool for sequence transduction tasks.

1.4.6 Transfer Learning

Transfer Learning refers to the set of methods that enable the transfer of knowledge gained from solving one problem to another. This has become particularly relevant in the era of Deep Learning, as models in this domain often require significant computational resources and time. By using pre-trained models as a starting point, Transfer Learning allows for the rapid development of high-performing models and the efficient resolution of complex problems in areas such as Computer Vision or Natural Language Processing (NLP).

Transfer Learning has become a powerful technique in machine learning, allowing the efficient use of resources and improving performance. It has been applied in various fields such as image classification, object detection, sentiment analysis, and speech recognition.

Transfer Learning draws heavily on the way humans learn, with the idea of reusing knowledge gained in other settings (source) to solve a particular problem (target). There are several approaches to Transfer Learning, depending on what is being transferred, when, and how the transfer is done. Generally, there are three types of Transfer Learning :

A. Inductive Transfer Learning In this configuration, the source and target domains are the same (same data), but the source and target tasks are different but closely related. The idea is to use existing models to advantageously reduce the scope of possible models (model bias), as illustrated in the figure below.

B. Unsupervised Transfer Learning Similar to inductive Transfer Learning, the source and target domains are similar, but the tasks are different. However, the data in both domains are not labeled. Obtaining large amounts of unlabeled data from databases and sources on the web, for example, is often easier than obtaining labeled data. Hence, the idea of using unsupervised learning in combination with Transfer Learning is gaining great interest.

C. Transductive Transfer Learning In this configuration, the source and target tasks are similar, but the corresponding domains differ either in terms of data or marginal probability distributions.

1.5 Machine and Deep learning applications

Artificial intelligence have various applications and that is thanks to the availability of data and the reliance of modern society on technology, this cause a lot of data scientist to work hard to research and apply ai to every aspect of the digital world and every economic sector. This made the application of AI dependent on the field expertise and the types of data handled in those fields, many data scientists like to differentiate between AI applications and the way those data translate in the field of application based on the type of data here are some of the major machine and deep learning applications :

1.5.1 Computer Vision

Image processing has a huge impact on a lot of critical economic sectors such as the medical, industrial, and security sectors this made the domain of computer vision a major field of application for Artificial intelligence, due to the nature of images and the advancement in processing power, a lot of research has been done to create optimal algorithms for image processing such as conventional neural networks that have revolutionized this field and enabled a wide range of application to existing such as facial recognition and object detection.

1.5.2 Natural language processing

Human communication is a huge challenge in the face of the creation of a global society and international affairs, languages are a key component of human communication but due to the nature of language and its difficulty the application of AI in this field is a must to optimize the human interaction and facilitate the globalization of the world. Natural language processing or NLP is a field of AI that consist of teaching computers

how to interpret human language in text Due to its nature, researchers had faced a lot of problems applying machine and deep learning techniques to it, but thanks to openAI researcher teams that lay the foundation of natural language processing with their paper 'attention is all you need where they introduce a new approach into text embedding called the 'Attention mechanism' and created a new architecture for deep learning called transformers, which was a revolution in the domain of AI and resulted in a lot of advancement in the field, such as the creation of the BERT model that is used for semantic analyses and GPT model that have taken the world by a storm with its capability to generate texts natural language processing has a lot of applications such as translation that enables access to all the information across all languages, semantic analyses like comment and review analyses that enable companies to understand and optimize their interactions with customers, and lastly chatbots like OpenAI's chatGPT and Google's Bard that can fully automate human interaction.

1.5.3 Speech and audio processing

The gap between humans and machines is based on the availability and simplicity of the interactions between them, and the advancement made in NLP that made it so that the machine can understand human languages can make those interactions easier but is still dependent on text. speech and audio processing in machine learning include a wide range of technologies that handle data in the form of audio such as automatic speech recognition some of those technologies are often used to differentiate between sound and speaker others are used in the segmentation of clips into classes.

1.5.4 Time series analyses

Time series analysis consists of using Machine and deep learning methods to analyze specific sequences of data points collected throughout time, those data points are usually time-sensitive, which makes time series data represent the evolution of the variable through time best example of time series problems is the stock market where the change in time can directly affect the output.

1.5.5 Anomaly detection

consists of applying machine and deep learning algorithms to examine specific data points and detecting rare occurrences that seem suspicious because they are different from the design default patterns of behavior changes. the techniques used in anomaly detection problems are mainly focused on learning the default patterns and later any changes in the patterns as an anomaly

1.6 Conclusion

In this chapter we have presented an overview of artificial intelligence and its relation with machine and deep learning, we started this chapter with a definition of machine learning and its different existing architectures, and after that, we presented the knowledge discovery in the database process in detail and described each step of the process, next we presented a definition and basic concept of deep learning and most used architectures, in the end, we presented machine and deep learning domains of applications. in the next chapter, we will present the state-of-the-art machine and deep learning techniques in both voice and natural language processing.

CHAPITRE 2

STATE-OF-THE-ART

2.1 Introduction

In the rapidly evolving landscape of machine learning and artificial intelligence, the emergence of Transformers and large language models has heralded a transformative era. These breakthroughs have unlocked new dimensions of natural language understanding, speech recognition, and text summarization. In this chapter, we delve into the state of the art in these fields, exploring their significance and the wealth of research efforts dedicated to overcoming the challenges within them.

In the realm of speech recognition, these technological strides have ushered in a new era of human-computer interaction. Through the integration of powerful speech recognition models, machines can now comprehend and respond to spoken language with unprecedented accuracy. This chapter sheds light on the remarkable progress in speech recognition, delving into the intricacies of acoustic modeling, language modeling, and the fusion of these components.

Similarly, text summarization has witnessed a paradigm shift, thanks to the advent of Transformers. These models have the unique capability to distill vast volumes of textual information into concise and coherent summaries. From extractive to abstractive summarization, this chapter explores the various approaches and techniques that have emerged, emphasizing the role of Transformers in elevating the quality and efficiency of text summarization.

As we navigate through the intricate web of advancements, we will witness how Transformers, bolstered by pre-trained models, have redefined the boundaries of language processing. From their application in text summarization, where they distill vast volumes of information into concise insights, to their pivotal role in speech recognition, where they bridge the gap between human communication and technology, Transformers have become the linchpin of modern AI. This chapter serves as a gateway to the expansive realm of contemporary research, revealing the solutions and innovations that have emerged to tackle the complexities of audio and text. Through an exploration of existing studies and their implications, we embark on a journey to comprehend the extraordinary potential of these technologies and the challenges that lie ahead.

This chapter serves as a gateway to the expansive realm of contemporary research, revealing the solutions and innovations that have emerged to tackle the complexities of audio and text. Through an exploration of existing studies and their implications, we embark on a journey to comprehend the extraordinary potential of these technologies and the challenges that lie ahead.

2.2 Summarization Technologie

In this section, we delve into the core concepts of summarization technologies. Summarization plays a pivotal role in our approach to audio summarization based on transformers and transfer learning. By understanding the foundations of summarization, we can appreciate the significance of our work in the broader context of natural language processing and data analysis.

2.2.1 Introduction to Summarization

Summarization technology, also known as text summarization, is a natural language processing (NLP) technique used to generate concise and coherent summaries of longer pieces of text. It aims to distill the most essential information from the source text while preserving its core meaning. Summarization technology can be broadly categorized into two main types : extractive and abstractive summarization.

2.2.2 Extractive Summarization

Extractive summarization involves selecting and extracting sentences or phrases directly from the source text to create a summary. These selected segments are typically the most representative and informative parts of the original text. Extractive summarization techniques rely on statistical and machine learning algorithms to determine the relevance of sentences or phrases.

2.2.3 Abstractive Summarization

Abstractive summarization, on the other hand, goes beyond mere extraction. It involves generating new sentences that capture the essential information of the source text in a more concise and coherent manner. Abstractive summarization techniques often employ natural language generation (NLG) models, such as transformers, to rewrite and rephrase the content while preserving its meaning.

2.2.4 Challenges in Summarization

Summarization technology faces several challenges, including content selection, coherence, and fluency of generated summaries. Ensuring that the summary retains the crucial details of the source text while avoiding redundancy and ambiguity is a complex task. Moreover, abstractive summarization introduces the challenge of generating human-like, contextually appropriate language.

2.2.5 Applications of Summarization

Summarization technology finds applications in various fields, such as journalism, content curation, information retrieval, and data analysis. It enables the automatic creation of news digests, the extraction of key insights from lengthy documents, and the generation of concise reviews, among others.

2.2.6 Importance in Audio Summarization

In the context of our thesis, understanding summarization technology is crucial because it forms the basis for our approach to audio summarization. We adapt and apply principles from text summarization to the domain of audio data, allowing us to create meaningful and coherent summaries of audio content using deep learning techniques.

2.3 Existence study

Many researchers suggest the application of machine learning and deep learning techniques to enhance Natural language processing and spoken language analyses. In this context, we present the state-of-the-art of recent works based on machine learning and deep learning.

2.3.1 Literature Review for the Article Towards End-to-end Speech-to-text Summarization

Objective :

- Raul Monteiro and Al, In This research paper aims to make significant contributions to the field of Speech-to-Text (S2T) abstractive summarization, particularly focusing on the French language. The objective is to explore the capabilities of pre-trained models and fine-tuning techniques to achieve high-quality, coherent, and fluent summaries that are similar to those written by human experts.[41]

Dataset :

- The dataset is a specialized corpus constructed from articles sourced from the Euro-News website. This dataset is unique in its focus on S2T abstractive summarization for the French language. The authors are in contact with EuroNews to potentially make this dataset publicly available, which could be a valuable resource for the research community.

Approach :

- The paper employs a Wav2Vec2ForCTC model, which is part of the Transformers library by Huggingface, and has been pre-trained on French speech data. The architecture includes a cross-modal adapter that serves as an intermediary to convert sequences of audio features into sequences of textual features. This adapter undergoes pre-training before being integrated into the S2T abstractive summarizer. The entire model is then fine-tuned on the BNews corpus. The model's performance is evaluated based on the ROUGE-2 score, and the checkpoint with the highest score on the development split is selected for further evaluation.

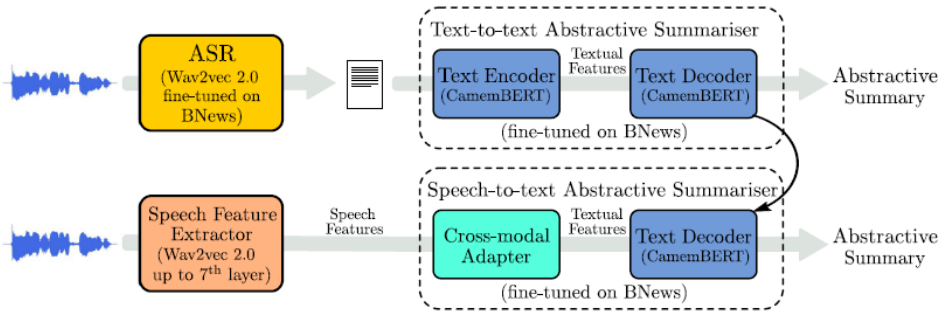


Fig. 2: Architectures of the cascade and E2E abstractive summarizers.

FIGURE 2.1 – ilustration of architecture in the article Towards End-to-end Speech-to-text Summarization

Results :

- The paper reports that their approach outperforms strong cascade baselines, which is a significant achievement in the field. The generated summaries are not only more natural but also coherent and fluent. They closely resemble summaries that a

human specialist in the field would produce. This is indicative of the model's ability to generate high-quality abstractive summaries, meeting the research objectives set forth in the paper.

2.3.2 Literature Review for the Article : Deep reinforcement and transfer learning for abstractive text summarization : A review

Objective :

- Ayham Alomaria and AL, in this paper aims to provide a comprehensive review of Automatic Text Summarization (ATS) with a focus on abstractive methods. It explores the state-of-the-art technologies, including deep neural sequence-to-sequence models, Reinforcement Learning (RL), and Transfer Learning (TL) approaches. [42]

Dataset :

- The paper does not focus on a specific dataset but rather reviews various datasets and their applications in ATS.

Approach :

- The paper categorizes ATS into extractive and abstractive types, emphasizing the challenges and requirements of abstractive ATS. It discusses the limitations of RNN-based models and how RL and TL can address these issues. The paper also reviews various Pre-Trained Language Models (PTLMs) like BERT, PEGASUS, T5, and their applications in ATS.

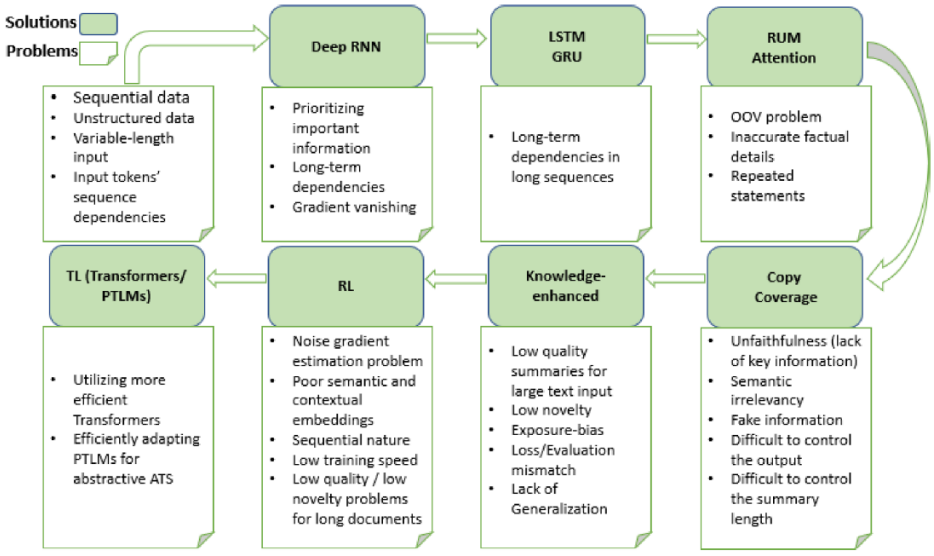


Fig. 19. Evolution of problems and solutions in sequence-sequence-based abstractive ATS models.

FIGURE 2.2 – challenges presented in the article Deep reinforcement and transfer learning for abstractive text summarization : A review

Results :

- The paper concludes that abstractive ATS has evolved significantly, benefiting from the advances in deep learning, RL, and TL. It highlights that abstractive summaries are more readable, consistent, and concise but also point out the challenges such as computational costs and the risk of generating trivial or incorrect summaries.

2.3.3 Literature Review for the Article : Attention Is All You Need

Objective :

- Ashish Vaswani and Al, in this paper introduce a novel network architecture called the Transformer, which is based solely on attention mechanisms. It aims to replace complex recurrent or convolutional neural networks in sequence transduction models, focusing on machine translation tasks. [38]

Dataset :

- The paper evaluates the Transformer on the WMT 2014 English-to-German and English-to-French translation tasks.

Approach :

- The Transformer architecture dispenses with recurrence and convolutions entirely, relying solely on attention mechanisms. It employs multi-head attention and scaled dot-product attention. The paper also introduces a unique learning rate schedule and employs the Adam optimizer with specific hyperparameters for training.

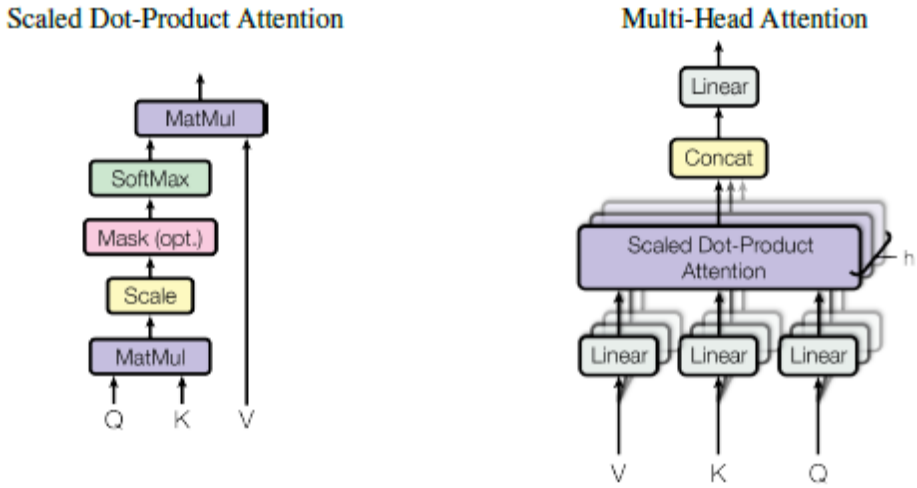


FIGURE 2.3 – ilustration of attention in the article Attention is all you need

Results :

- The Transformer achieves a BLEU score of 28.4 on the WMT 2014 English-to-German translation task and 41.8 on the English-to-French translation task. These scores outperform existing best results, including ensembles, and the model requires significantly less time to train. The paper also shows that the Transformer generalizes well to other tasks like English constituency parsing.

2.3.4 Literature Review for the Article : Universal Speech Models for Automatic Speech Recognition and Translation

Objective :

- Yu Zhang and Al, in this paper aims to develop Universal Speech Models (USM) that can be applied to both Automatic Speech Recognition (ASR) and Automatic Speech Translation (AST). It introduces a scalable speech representation learner called BEST-RQ and a method called MOST for utilizing large-scale text data to improve speech tasks.[43]

Dataset :

- The paper uses YouTube data for training USMs for YouTube closed captions. It evaluates the models on public benchmarks like SpeechStew, FLEURS, and CO-RAAL for ASR, and CoV oST 2 for AST.

Approach :

- The paper employs a three-stage training pipeline. The first stage involves unsupervised pre-training using BEST-RQ on a large unlabeled speech dataset. The second stage, called MOST, involves multi-objective supervised pre-training that utilizes various kinds of datasets. The third stage fine-tunes the pre-trained encoder on ASR or AST tasks. The paper also explores the use of adapter units for both generic and pre-trained ASR models.

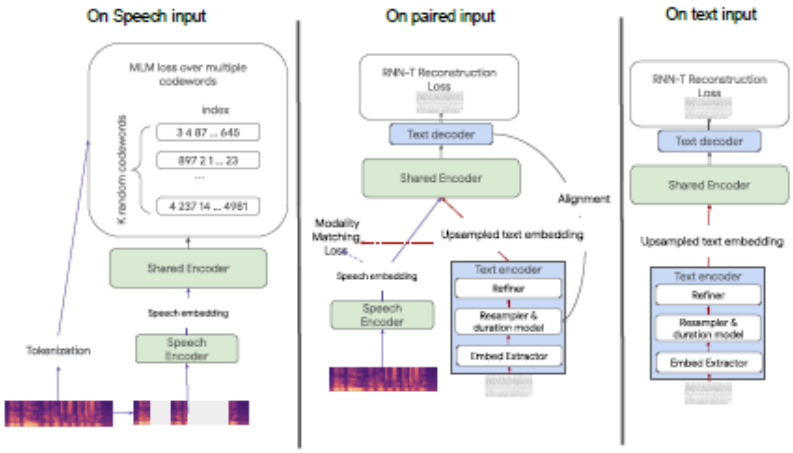


Figure 5: Overview of MOST text injection. The left-most panel depicts MOST training on unlabeled speech input; the center panel depicts training on paired speech and text input; the right-most panel depicts training on unlabeled text data.

FIGURE 2.4 – illustration of approach in the article Universal Speech Models for Automatic Speech Recognition and Translation

Results :

- The paper establishes a new state-of-the-art performance on public speech translation tasks. It demonstrates that the USM models can be effectively fine-tuned on downstream tasks, achieving quality gains for the FLEURS and CoV oST 2 tasks. The paper also presents results on long-form test sets like CORAAL.

2.3.5 Literature Review for the Article : Evaluating ChatGPT's Performance on Summary Inconsistency Detection

Objective :

- Zheheng Luo and Al, this paper aims to evaluate the performance of ChatGPT in detecting inconsistencies in generated summaries. It explores two different zero-shot prompts in the Natural Language Inference (NLI) setting to assess the model's ability to identify factual inconsistencies.[44]

Dataset :

- The paper evaluates ChatGPT's performance on the SUMMAC benchmark, which includes six of the largest summary inconsistency detection datasets like FactCC, CoGenSumm, XSumFaith, SummEval, FRANK, and Polytope.

Approach :

- The paper employs two different zero-shot prompts. The first prompt directly asks ChatGPT to answer 'yes' or 'no' to infer the consistency between the source document and the corresponding generated summary. The second prompt is inspired by a zero-shot Chain-of-Thought approach, which encourages the model to think step-by-step before making a judgment.

Results :

- The paper reports that ChatGPT performs reasonably well in detecting inconsistencies in summaries. It also highlights that the model's performance can be unstable when changing the label, order, and amount of examples in the prompt. The paper suggests that further exploration is needed to stabilize performance.

Methods	SUMMAC Benchmark Datasets					
	CoGenSum	XsumFaith	Polytope	FactCC	SummEval	FRANK
NER Overlap	53.0	63.3	52.0	55.0	56.8	60.9
MNLI-doc	57.6	57.5	61.0	61.3	66.6	63.6
FactCC-CLS	63.1	57.6	61.0	75.9	60.1	59.4
DAE	63.4	50.8	62.8	75.9	70.3	61.7
FEQA	61.0	56.0	57.8	53.6	53.8	69.9
QuestEval	62.6	62.1	**70.3**	66.6	72.5	82.1
SummaC_{ZS}	70.4	58.4	62.0	83.8	78.7	79.0
SummaC_{Conv}	64.7	**66.4**	62.7	**89.5**	81.7	81.6
ChatGPT_{ZS}	63.3	64.7	56.9	74.7	76.5	80.9
$\text{ChatGPT}_{ZS\text{-}COT}$	**74.3**	63.1	61.4	79.5	**83.3**	**82.6**

Table 2: Balanced accuracy results of inconsistency detect models on the test set of SummaC. Results of baselines are referenced from the paper (Laban et al., 2022).

FIGURE 2.5 – Results of The review in the article Evaluating ChatGPT's Performance on Summary Inconsistency Detection

2.3.6 Literature Review for the Article : Robust Speech Recognition via Large-Scale Weak Supervision

Objective :

- Alec Radford and Al, in this paper aims to improve the robustness of Automatic Speech Recognition (ASR) systems through large-scale weak supervision. It introduces a new training approach that leverages a massive amount of weakly labeled data to enhance the model's performance.[1]

Dataset :

- The paper uses a large-scale dataset consisting of 680k hours of multitask training data, including English transcription, any-to-English speech translation, and non-English transcription.

Approach :

- The paper employs a sequence-to-sequence learning approach with Transformer encoder and decoder blocks. It uses a three-stage training pipeline involving unsupervised pre-training, multi-objective supervised pre-training, and fine-tuning. The paper also introduces a custom vocabulary and time-aligned transcription to improve ASR performance.

Results :

- The paper reports a strong correlation between the amount of training data and the resulting zero-shot performance for different languages. It also discusses the challenges faced by languages that are more distantly related to the Indo-European languages, such as Hebrew, Telugu, Chinese, and Korean.

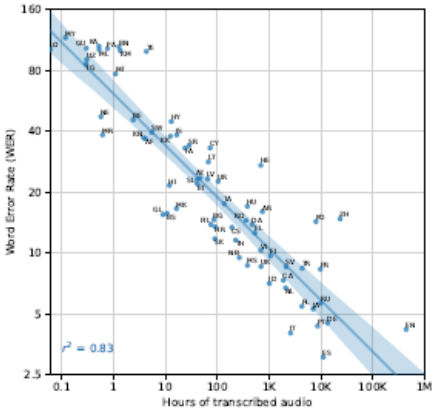


Figure 3. **Correlation of pre-training supervision amount with downstream speech recognition performance.** The amount of pre-training speech recognition data for a given language is very predictive of zero-shot performance on that language in Fleurs.

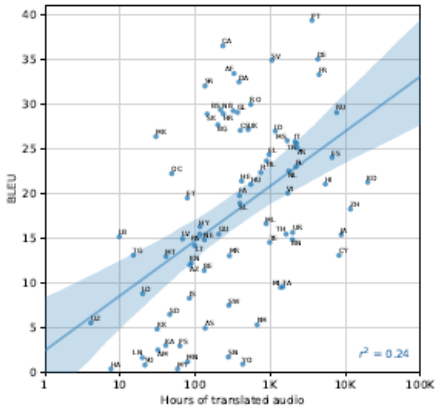


Figure 4. **Correlation of pre-training supervision amount with downstream translation performance.** The amount of pre-training translation data for a given language is only moderately predictive of Whisper's zero-shot performance on that language in Fleurs.

FIGURE 2.6 – Results of whisper model depending on amount of training data

2.3.7 Literature Review for the Article : BART : Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension

Objective :

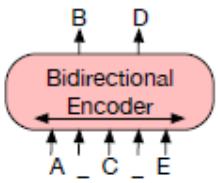
- Mike Lewis and AL, in this paper introduce BART, a denoising autoencoder for pretraining sequence-to-sequence models. It aims to generalize various pretraining schemes like BERT and GPT and is designed for a wide range of NLP tasks including generation, translation, and comprehension. [45]

Dataset :

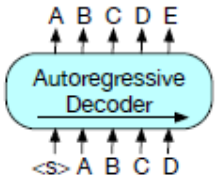
- The paper evaluates BART on multiple benchmarks including GLUE, SQuAD, and XSum, among others.

Approach :

- BART employs a sequence-to-sequence architecture with a bidirectional encoder and an autoregressive decoder. It uses various noising functions to corrupt the input text and then trains the model to reconstruct the original text. The paper also explores different types of noising schemes like token masking, token deletion, and text infilling.



(a) BERT: Random tokens are replaced with masks, and the document is encoded bidirectionally. Missing tokens are predicted independently, so BERT cannot easily be used for generation.



(b) GPT: Tokens are predicted auto-regressively, meaning GPT can be used for generation. However words can only condition on leftward context, so it cannot learn bidirectional interactions.

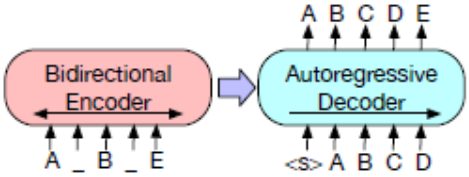


FIGURE 2.7 – ilustration of architecture in the article BART

Results :

- BART achieves state-of-the-art performance on a variety of tasks. It matches the performance of RoBERTa on GLUE and SQuAD benchmarks and sets new records on abstractive dialogue, question answering, and summarization tasks. It improves performance by up to 6 ROUGE points on the XSum dataset.

	CNN/DailyMail			XSum		
	R1	R2	RL	R1	R2	RL
Lead-3	40.42	17.62	36.67	16.30	1.60	11.95
PTGEN (See et al., 2017)	36.44	15.66	33.42	29.70	9.21	23.24
PTGEN+COV (See et al., 2017)	39.53	17.28	36.38	28.10	8.02	21.72
UniLM	43.33	20.21	40.51	-	-	-
BERTSUMABS (Liu & Lapata, 2019)	41.72	19.39	38.76	38.76	16.33	31.15
BERTSUMEXTABS (Liu & Lapata, 2019)	42.13	19.60	39.18	38.81	16.50	31.27
BART	**44.16**	**21.28**	**40.90**	**45.14**	**22.27**	**37.25**

FIGURE 2.8 – Results in the article BART

2.4 Conclusion

In conclusion, the state of the art in Transformers and large language models represents a pivotal juncture in the fields of natural language processing, speech recognition, and text summarization. The advent of these powerful models has not only expanded the horizons of AI but has also fundamentally altered the way we interact with and understand both textual and audio data.

Throughout this chapter, we have witnessed the remarkable strides made in leveraging Transformers and pre-trained models to unravel the complexities of language understanding, enabling applications such as text summarization to distill knowledge from vast corpora and speech recognition to enhance human-computer communication.

In the chapters that follow, we will delve deeper into the intricacies of implementation and experimentation, where we will witness firsthand the impact of these state-of-the-art techniques in the context of audio summarization using Transformers and pre-trained models. As we embark on this journey, we remain cognizant of the vast potential these technologies hold and the ongoing pursuit of excellence in the realm of AI research.

CHAPITRE 3

CONCEPTION AND IMPLEMENTATION OF AUDIO SUMMARIZING SYSTEM

3.1 introduction

In this chapter, we present our comprehensive approach for audio summarization, harnessing the power of two pre-trained transformer-based architectures through the application of transfer learning techniques. This chapter provides a meticulous overview of the intricate design underpinning our approach, facilitating a deeper comprehension of its multifaceted functionalities.

Our approach is meticulously structured into distinct phases encompassing data preprocessing, audio file handling, and data preparation for fine-tuning. We elucidate the various techniques and tools seamlessly integrated into each phase. The primary aim of our research is to establish a dependable and efficient audio summarization system, capitalizing on cutting-edge advancements in deep learning methodologies. This approach is meticulously engineered to tackle the constraints of conventional spoken language processing methods and grapple with the intricacies posed by vast and intricate audio data.

Our overarching goal is to cultivate a resilient and adaptable system, capable of learning from data and swiftly adapting to novel scenarios in real time. To attain this objective, our approach is bifurcated into discrete phases, each serving as a pivotal cog in the machinery of our system's triumph. This chapter meticulously delineates each phase, elucidating their symbiotic contributions to the ultimate success of our system. Furthermore, we delve into the challenges and opportunities intrinsic to our approach and present preliminary results that underscore its efficacy.

3.2 General Approach

Our approach seeks to address the evolving challenges in audiobook summarization, primarily focusing on the need to reduce the time required to comprehend audio content while enhancing accuracy. This endeavor is motivated by the growing demand for efficient and precise methods to extract meaningful insights from audio data. Our approach is structured around five key phases, each meticulously designed to contribute to the development of our audio summarization system. In the subsequent sections, we will delve into each phase in detail. Here, we provide an overview to highlight the critical components of our approach.

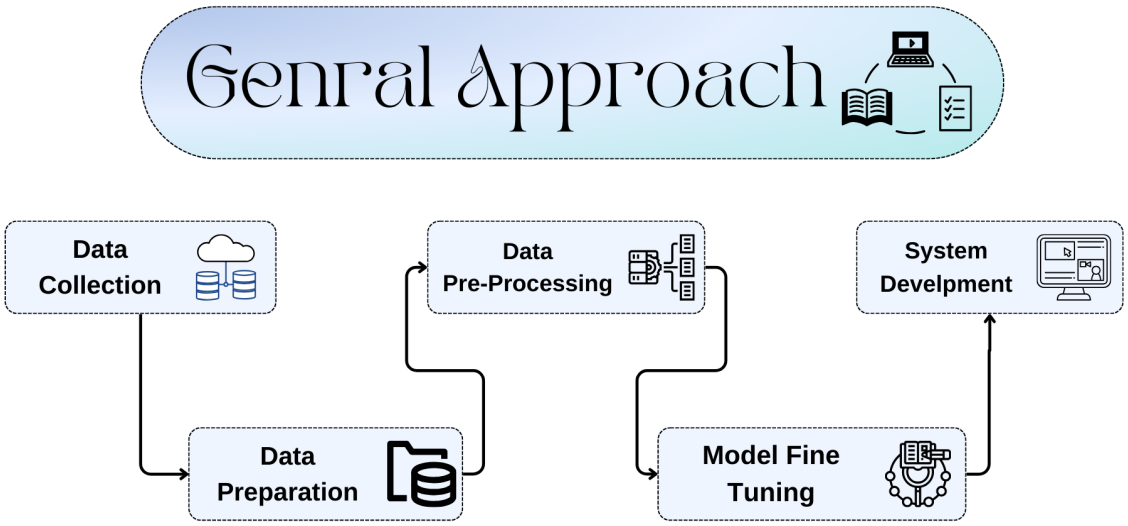


FIGURE 3.1 – illustration of the Approach Architecture

Data Collection

The first phase of our approach is data collection. In this phase, we meticulously curate and select our data sources to ensure that our model aligns seamlessly with the specific requirements of audiobook summarization. We will delve into the details of our data sources, extraction processes, and the rationale behind our choices in the upcoming sections.

Data Preparation

Following data collection, we proceed to data preparation. This phase involves the careful adjustment of the dataset to suit our particular needs. We consider factors such as data format, structure, and organization, making necessary modifications to optimize the dataset for subsequent processing steps. Detailed discussions on our data preparation techniques will be presented in the forthcoming sections.

Data Preprocessing

The data preprocessing phase is dedicated to preparing our dataset for model fine-tuning. It involves various transformations and enhancements aimed at improving the quality and usability of our data. We will elaborate on the specific preprocessing steps, techniques, and tools employed to condition our data for further model development.

Model Fine-Tuning

Model fine-tuning is a critical phase in our approach. Here, we engage in the training and specialization of our audio model to perform effectively in the context of audiobook summarization. We will provide comprehensive insights into the architectural details of our chosen models, the fine-tuning procedures, and the parameters that enable our models to excel in their respective tasks.

System Design and Application

In the final phase of our approach, we present our proposed system design and application. This culmination represents the synthesis of our efforts across the previous phases, resulting in a functional audio summarization system. We will offer an in-depth look into the development process, the integration of models, and the features of our system.

Our approach centers around the utilization of transformer-based large language models, harnessing their capabilities for spoken language detection, transcription, and text summarization. Through extensive research and state-of-the-art reviews, we have identified the optimal models for our tasks. Specifically, we have chosen the OpenAI Whisper

model for audio transcription and the Facebook BART-Large-CNN model for text summarization.

Our commitment to advancing audiobook summarization stems from the recognition of the need for efficient and accurate methods to extract insights from audio content. The transformative potential of our approach lies in its ability to bridge the gap between time-consuming audio comprehension and streamlined access to meaningful content. We have undertaken extensive research to identify the most suitable models, bearing in mind the ever-evolving landscape of AI and natural language processing.

The following sections will delve into the specifics of each phase of our approach, providing a comprehensive understanding of our journey towards an enhanced audiobook summarization system.

3.3 Environment and Tools

3.3.1 Libraries used

3.3.1.1 Python

Python is a popular high-level programming language that is widely used for general-purpose programming. It was created in the late 1980s by Guido van Rossum, and its design philosophy emphasizes code readability and ease of use. Python is known for its simplicity, clarity, and elegance, and it has a large and active community of developers who contribute to its growth and development. It is used for a wide range of applications, including web development, scientific computing, data analysis, artificial intelligence, and automation, among others. One of the main advantages of Python is its ease of learning and use, which makes it an ideal choice for beginners and professionals alike. Its syntax is straightforward and readable, and it offers a vast array of libraries and frameworks that simplify complex tasks, making it one of the most popular programming languages in the world.

FIGURE 3.2 – python
[46]

3.3.1.2 PyTorch

PyTorch is an open-source machine learning library for Python that provides a flexible and dynamic framework for building deep learning models. It is widely used by researchers and developers for tasks like natural language processing, computer vision, and reinforcement learning.

One of PyTorch's key features is its dynamic computation graph, which allows for dynamic, on-the-fly graph generation during model training. This dynamic approach is particularly beneficial for research and experimentation.

PyTorch provides extensive support for neural network architectures, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformers. Its user-friendly interface simplifies the process of designing and training complex models.

The library offers GPU acceleration for faster model training and supports seamless integration with popular Python libraries like NumPy, SciPy, and Matplotlib. PyTorch's active community and extensive documentation make it an excellent choice for machine learning practitioners.

In summary, PyTorch is a versatile and powerful library that caters to both beginners and experts in deep learning. Its dynamic computation graph and extensive model support make it a preferred choice for various machine-learning tasks.

FIGURE 3.3 – pytorch
[47]

3.3.1.3 Streamlit

Streamlit is a Python library that simplifies the process of creating web applications for data science and machine learning. It allows users to build interactive and data-driven apps with minimal code and effort.

With Streamlit, developers can turn data scripts into shareable web apps quickly. It provides a straightforward and intuitive API for creating widgets, charts, and visualizations, making it accessible to both beginners and experienced developers.

Streamlit's real-time preview feature enables instant updates to the app as code changes are made, streamlining the development process. It also supports integration with popular data science libraries like Pandas, Matplotlib, and Plotly.

In summary, Streamlit is a valuable tool for data scientists and developers looking to create interactive and user-friendly web applications for showcasing data and machine learning projects.

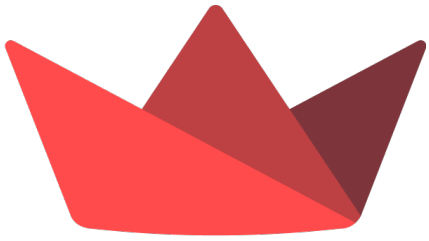


FIGURE 3.4 – Streamlit
[48]

3.3.1.4 Hugging Face Transformers

The Hugging Face Transformers library is a comprehensive and versatile open-source library for natural language processing (NLP) tasks. It is specifically designed for working with transformer-based models, which have revolutionized NLP in recent years.

Transformers, as implemented in this library, enable users to perform a wide range of NLP tasks, including text classification, named entity recognition, translation, summarization, and more. The library offers a user-friendly API for accessing pre-trained transformer models, making it accessible to both beginners and NLP experts.

One of the standout features of the Transformers library is its vast collection of pre-trained models, including popular variants like BERT, GPT-2, and Roberta. These pre-trained models can be fine-tuned for specific downstream tasks with relatively little effort, allowing users to achieve state-of-the-art results on various NLP benchmarks.

The library is built on top of PyTorch and TensorFlow, providing compatibility with both deep learning frameworks. It also offers tools for quickly loading and working with pre-trained models, tokenization of text data, and model inference. Whether you need to build a chatbot, perform sentiment analysis, or tackle a custom NLP problem, the Transformers library from Hugging Face provides a powerful and flexible foundation for NLP tasks.

FIGURE 3.5 – Hugging Face Transformers
[49]

3.3.1.5 Numpy

Numpy is a library for the Python programming language that adds support for large multidimensional arrays and matrices and a large collection of high-level mathematical functions to operate on these arrays.

FIGURE 3.6 – numpy
[50]

3.3.1.6 PyDub

PyDub is a Python library for working with audio files. It provides a simple and easy-to-use interface for various audio-related tasks, such as reading and writing audio files in different formats, editing audio, and applying various audio effects. PyDub can be used for tasks like audio file conversion, cutting and joining audio clips, adjusting volume levels, and more. It's a handy tool for working with audio data in Python applications

3.3.2 Tools

3.3.2.1 Jupyter Notebook

is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. Jupyter Notebook is widely used in the field of AI and machine learning for exploratory data analysis, prototyping, and sharing research results.

FIGURE 3.7 – Jupyter Notebook
[51]

3.3.2.2 VS Code

Visual Studio Code is an open-source code editor developed by Microsoft that is both lightweight and customizable. It can be run on desktop computers and is compatible with Windows, macOS, and Linux. It has built-in support for JavaScript, TypeScript, and Node.js, as well as an extensive range of extensions for other languages and runtimes, including C++, C#, Java, Python, PHP, Go, and . NET. Despite its compact size, Visual Studio Code is a powerful editor with a wealth of features.

FIGURE 3.8 – VS Code
[52]

3.3.2.3 Anaconda

Anaconda, founded in 2012 by Peter Wang and Travis Oliphant, is a comprehensive platform integral to our development environment. It offers an integrated solution for data science, including various tools and libraries. Anaconda facilitates the management of Python environments, packages, and dependencies.

With over 35 million users worldwide, Anaconda has grown alongside Python's popularity, becoming a widely used programming language. It emphasizes data literacy, aiming to empower individuals to derive insights from data. Anaconda actively supports open-source projects, contributing to the data science community.

Driven by a passion for data science, Anaconda promotes transparent, repeatable, and bias-free data science models. Its tools cater to both experienced data scientists and beginners, democratizing data literacy. In our development environment, Anaconda plays a foundational role, ensuring a seamless Python environment and package management. It aligns with our mission to enhance audiobook summarization through cutting-edge technology.

FIGURE 3.9 – anaconda
[53]

3.3.3 Hardware

The hardware configuration used in our implementation is as follows :

TABLE 3.1 – Table of Hardware Specs

Specs	Lenovo Legion 5
Processor (CPU)	AMD Ryzen 7 5800H with Radeon Graphics 3.20 GHz
Graphic-Card(GPU)	Radeon Integrated Graphics + Nvidia RTX 3060 Mobile
Memory (RAM)	16.0 GB
Operation System	Windows 11 Home 64-bit
Storage	1TB HDD + 512GB SSD

3.4 Dataset

The foundation of any successful endeavor in artificial intelligence and machine learning lies in the data itself. It is the lifeblood that fuels the models and guides their learning. Therefore, the utmost care and attention must be devoted to how data is sourced, prepared, and processed. Our data handling and manipulation plan, illustrated in the figure below, comprises three pivotal steps : Data Source and Selection, Data Preparation and Exploratory Analyses, and Data Preprocessing. Each of these steps is fundamental in ensuring the quality and suitability of the data for our audio summarization task. In the subsequent subsections, we will delve into the specifics of each step, shedding light on the meticulous processes that underpin our approach.

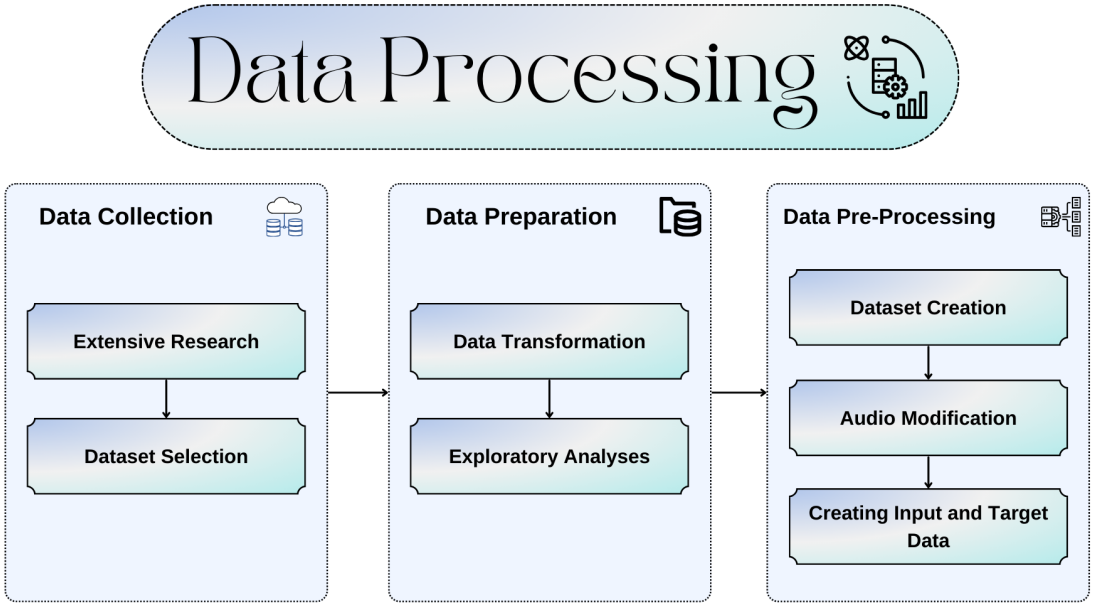


FIGURE 3.10 – illustration of the Data Processing process

3.4.1 Data Source

Our choice of dataset for audio summarization emerged from extensive research, drawing from academic papers, the state of the art, and various dataset repositories. After a thorough evaluation, we selected a Kaggle dataset curated by Nikita Fedorov, a distinguished Data Scientist with over 4 years of experience in machine learning and statistical analysis.

3.4.1.1 Dataset Description :

The dataset comprises 500 audio files, accompanied by their corresponding texts and summaries. This rich collection of data is designed to facilitate audio summarization tasks, offering the flexibility to implement both step-by-step audio summarization (combining speech recognition with text summarization) and end-to-end audio summarization.

3.4.1.2 Usage Flexibility :

The dataset offers versatility in its usage :

- **Text Summarization** : It provides texts and summaries tailored for text summarization tasks.
- **Speech Recognition** : Audio and texts can be employed for speech recognition challenges.

3.4.1.3 License :

This dataset, made available in 2023, is distributed under the 'Attribution-NonCommercial-ShareAlike 3.0 IGO (CC BY-NC-SA 3.0 IGO)' license.

3.4.1.4 Dataset Characteristics :

- **Size** : The dataset encompasses over 9 GB of data.
- **Audio Files** : It comprises more than 500 audio files, each with accompanying text transcriptions and summaries.
- **Audio Length** : The audio files are substantial, with durations exceeding 15 minutes.
- **Sample Rate** : Audio files maintain a sample rate of 48.
- **Diverse Speakers** : The dataset includes a wide array of speakers with varying backgrounds, ethnicities, and genders, ensuring the model's robustness across different English accents.
- **Content Variety** : It covers a broad spectrum of audiobooks, spanning from fiction novels to scientific research publications.

These dataset characteristics underscore its suitability for our project, as it offers a comprehensive and diverse set of data that aligns with the objectives of our audio summarization system.

3.4.2 Data Preparation

In the journey towards building a robust audio summarization system, a critical step lies in ensuring that the raw data is meticulously examined and made ready for further processing. The Data Preparation phase acts as a crucial bridge between the initial dataset and the subsequent Data Preprocessing step. It involves a meticulous analysis of our dataset to identify any irregularities or inconsistencies that could hinder the model's performance. In this phase, we employ a two-step approach to validate and format our data, setting the stage for seamless Data Preprocessing.

3.4.2.1 Data Transformation :

Before initiating the preprocessing stage, it is imperative to ensure that our audio files are in the most suitable format for effective handling. Extensive research has shown that the WAV format provides the best compatibility and treatment for audio data.

Consequently, we developed a custom function to inspect our audio files and convert any non-WAV formats into the WAV format using the PyDub library. The algorithm for this transformation is outlined below :

Algorithm 2: Audio Format Conversion Algorithm

Input: Wrong_Folder_Path, Right_Folder_Path
Output: Clean_Audio_Files

```
for Each file in Wrong_Folder_Path do
    if file ends with ".mp3" or file ends with ".flac" then
        // Create the new .wav filename
        out_file ← Right_Folder_Path + RemoveExtension(GetFileName(file))
         + ".wav"
        // Read in the audio file and export it in WAV format
        AudioSegment.from_file(file).export(out_file, format="wav")
        // Print a message indicating the transformation
        Print("Creating " + out_file)
    end
end
return Clean_Audio_Files
```

3.4.2.2 Exploratory Analyses :

Our data preparation efforts extend to conducting exploratory analyses, with a particular focus on audio files. We created a dedicated function to iterate through our audio dataset and compile essential characteristics into a structured dataframe. This approach enables us to perform in-depth analyses, identify anomalies, and assess the readiness of audio files for further processing. The algorithm for this exploratory analysis is provided below :

Algorithm 3: Audio Exploratory Analysis Algorithm

Input: dataset
Output: audio_df

```
audio_data ← []
for Each audio_file in dataset do
    audio_info ← { 'File Name' : audio_file.name, 'Duration' :
     audio_file.duration, 'Sample Rate' : audio_file.sample_rate, 'Channels' :
     audio_file.channels, 'Bit Depth' : audio_file.bit_depth, 'Size' :
     audio_file.file_size }
    audio_data.append(audio_info)
end
// Convert audio_data into a dataframe for analysis
audio_df ← pd.DataFrame(audio_data)
return audio_df
```

Following the exploratory analysis of our audio files, we obtained insights that confirmed the readiness of the audio data for preprocessing. Additionally, for our text-based data, we executed a script to validate correct labeling and readiness for subsequent preprocessing stages.

These preparatory steps ensure that our dataset is appropriately formatted and analyzed, setting the foundation for effective data preprocessing.

3.4.3 Data Preprocessing

In this section, we elucidate the crucial steps undertaken to prepare our data for model fine-tuning. Data preprocessing plays a pivotal role in ensuring that our multimodal Transformer-based model can effectively learn from our dataset. Our process encompasses multiple stages, each tailored to address specific requirements for audio transcription and text summarization.

3.4.3.1 Dataset Creation

The initial step in data preprocessing involves structuring our raw dataset for efficient access and manipulation. Leveraging the Dataset library, we transform the dataset from directory files into a structured format in our memory. This transformation ensures that every audio file is associated with its corresponding transcript and summary, providing essential context for our subsequent tasks.

Algorithm 4: Dataset Creation Algorithm

Output: custom_dataset_dict
`// Define paths to data folders`
Specify data folder paths
`// Create lists of file paths`
Create lists of audio, transcript, and summary file paths
`// Check file count consistency`
Ensure the number of files in each folder matches
`// Create a dataset dictionary`
Form a dataset dictionary from file paths
`// Create a Hugging Face Dataset`
Generate a custom dataset using the dictionary
`// Split dataset`
Split the dataset into training, validation, and test sets
`// Create a DatasetDict`
Organize datasets into a DatasetDict
return *custom_dataset_dict*

3.4.3.2 Audio Modification

To ensure compatibility with the Whisper model's input requirements, we perform audio modifications. Specifically, we adjust the sample rate of each audio file to 16,000 Hz, as Whisper is trained on audio files with this sampling rate. This harmonization step is crucial to ensure that our model processes the audio inputs correctly.

3.4.3.3 Creating Input and Target Data

For fine-tuning our model, we need to create both input and target data in a format suitable for training. This involves employing Whisper's feature extractor and tokenizer. Whisper relies on Mel spectrograms as input representations, which capture the frequency characteristics of speech. These spectrograms are created from the audio inputs, ensuring that our model can effectively process them.

Simultaneously, we encode the target text data into numerical tensors using the Whisper tokenizer. The tokenizer maps sequences of text tokens to their corresponding indices, facilitating the alignment between audio and text during training. The pre-trained Whisper tokenizer, encompassing an extensive vocabulary, streamlines this process, offering multilingual capabilities. using the example of latex structure write the pseudocode :

Algorithm 5: Dataset Preparation Algorithm

Input: batch
Output: processed_batch
// Load and resample audio data from 48 to 16kHz
audio ← batch["audio"]
// Compute log-Mel input features from input audio array
batch["input_features"] ← feature_extractor(audio["array"], sampling_rate=audio["sampling_rate"]).input_features[0]
// Encode target text to label ids
batch["labels"] ← tokenizer(batch["transcript"]).input_ids
return *batch*

Our data preprocessing pipeline, outlined above, is meticulously designed to prepare our dataset for fine-tuning the Whisper model effectively. These steps ensure that both audio and text data are appropriately encoded, setting the stage for model training and, ultimately, the audio summarization task.

3.5 Model Fine-Tuning

3.5.1 Audio Transcript Model Using OpenAI WHISPER

After conducting extensive research and a thorough review of the state-of-the-art, we have chosen to employ the Whisper model for fine-tuning. Whisper aligns perfectly with our research objectives, owing to its unique characteristics and capabilities, which we will delve into in the following sub-subsections. This section will comprehensively introduce and elucidate the Whisper architecture, detail our fine-tuning process, and present the outcomes of our fine-tuning efforts. the fine-tuning process we undertook is highlighted in the following figure :

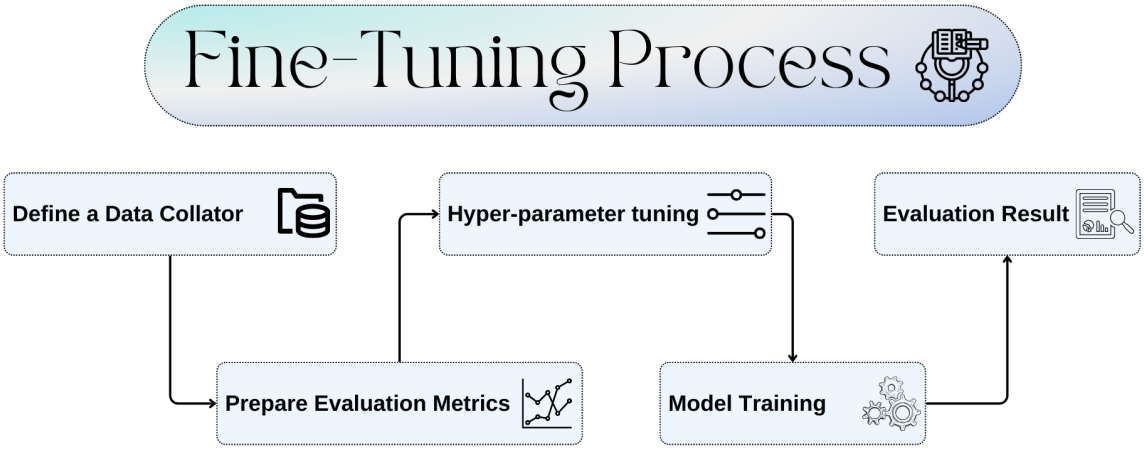


Figure 3.11 – Fine-tuning approach

3.5.1.1 Whisper Architecture

In this section, we delve into the theoretical foundations and architectural details of the Whisper model, which serves as the cornerstone of our audio transcription system. The insights presented here are drawn from the article "Robust Speech Recognition via Large-Scale Weak Supervision." which we have reviewed in our state-of-the-art existence study.

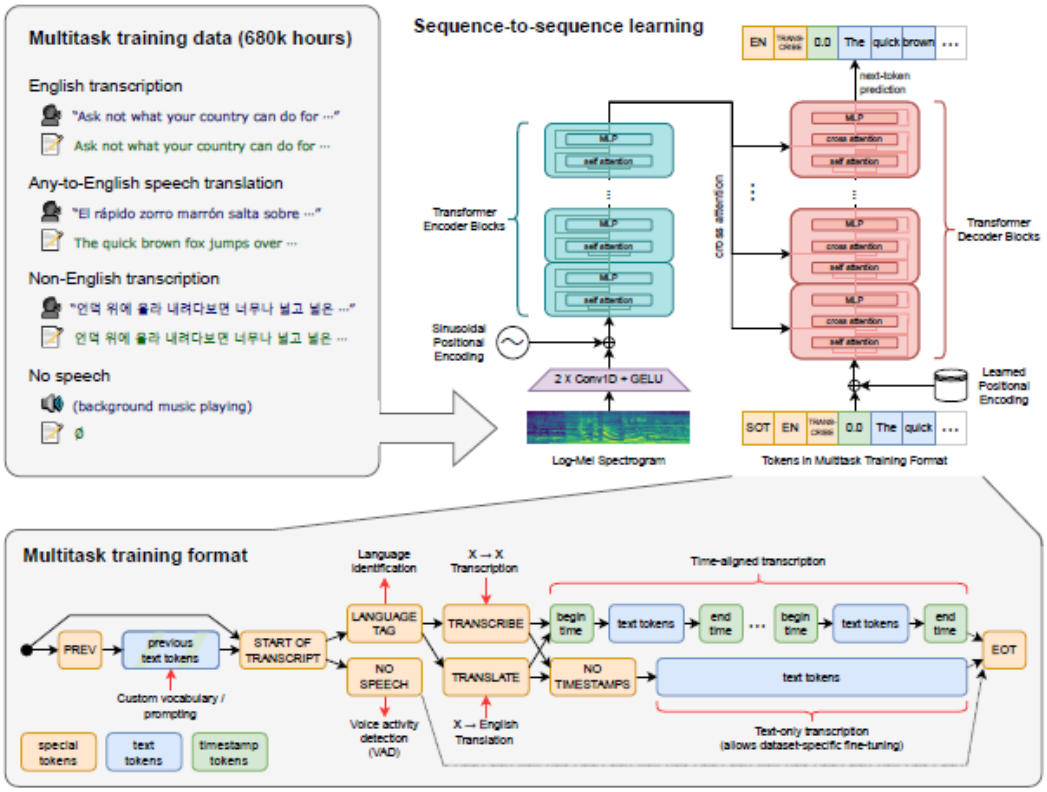


FIGURE 3.12 – illustration of the Whisper Architecture [1]

A. Data Processing Approach Whisper adopts an unconventional approach to data processing, departing from extensive standardization and relying on the expressive capacity of sequence-to-sequence models. Key facets of this approach include :

- **Raw Text Prediction :** Whisper models are trained to predict raw text transcripts without imposing significant standardization.
- **Diverse Dataset Construction :** The dataset is curated from internet sources, encompassing a wide spectrum of audio recordings from diverse environments, speakers, languages, and recording setups.
- **Transcript Quality Enhancement :** Automated filtering methods are employed to address transcript quality issues and improve overall dataset quality.

B. Dataset Construction The Whisper dataset's construction involves the following key steps :

- **Internet-Sourced Data :** Whisper's dataset is compiled from online sources and encompasses a rich variety of audio content.
- **Quality Assurance :** To ensure dataset quality, a stringent filtering process is implemented, filtering out machine-generated transcripts and retaining human-generated content.

- **Language Matching :** An audio language detector verifies that the spoken language aligns with the transcript's language according to CLD2, excluding mismatched pairs.
- **Deduplication :** Fuzzy deduplication techniques reduce redundancy and eliminate automatically generated content.

C. Audio Segment Pairing To facilitate training, audio files are segmented into 30-second intervals, each paired with the corresponding transcript subset within that segment. Even segments lacking speech are included with sub-sampled probability, and these segments are instrumental in training voice activity detection.

D. Additional Filtering After an initial model training phase, an additional filtering pass is conducted, taking into account error rates and data source sizes. This manual inspection identifies and eliminates low-quality data sources, including misaligned and inadequately transcribed content.

E. Whisper Model Architecture The Whisper model architecture is detailed as follows :

- **Encoder-Decoder Transformer :** Whisper employs an encoder-decoder Transformer architecture, well-suited for scalability.
- **Audio Preprocessing :** Audio is resampled to 16,000 Hz, and an 80-channel log-magnitude Mel spectrogram representation is computed from 25-millisecond windows with a 10-millisecond stride.
- **Feature Normalization :** Input features are globally scaled between -1 and 1 with approximately zero mean.
- **Position Embeddings :** Sinusoidal position embeddings are added to the stem.
- **Encoder Processing :** The encoder processes the input representation with pre-activation residual blocks.
- **Decoder Configuration :** The decoder employs learned position embeddings and tied input-output token representations.

F. Multitask Format Whisper's multitask format enables a single model to handle various speech-processing tasks. The format comprises a sequence of input tokens specifying tasks and conditioning information, including language prediction, task specification, and timestamp prediction. Text history is integrated to enhance audio understanding.

G. Training Details Training the Whisper model involves the following key considerations :

- **Parallel Training :** Whisper is trained using data parallelism across accelerators.
- **Precision and Scaling :** FP16 with dynamic loss scaling and activation checkpointing are utilized.
- **Optimization Techniques :** Training employs AdamW and gradient norm clipping, with a linear learning rate decay.
- **Batch Size :** A batch size of 256 segments is employed, and models are trained for a specific number of updates.
- **Generalization Strategy :** The model relies on dataset diversity to encourage generalization and robustness, without additional data augmentation or regularization.

H. Speaker Annotation Handling To mitigate incorrect speaker name predictions, Whisper models undergo brief fine-tuning on transcripts that lack speaker annotations.

This comprehensive overview of the Whisper model's architecture and data processing approach lays the groundwork for subsequent sections on fine-tuning and post-fine-tuning analyses.

3.5.1.2 Fine-Tuning the Model

With our data prepared, the next crucial step is to fine-tune our model to make it suitable for the specific task at hand. This subsection provides an in-depth overview of the training pipeline, highlighting key steps, configurations, and evaluation processes involved.

The fine-tuning process begins with setting up the training and evaluation procedures. This involves leveraging the huggingface Trainer, which simplifies many aspects of model training. The following steps are undertaken :

1. **Define a Data Collator** : In the context of a sequence-to-sequence speech model, the data collator plays a unique role. It treats input features and labels independently, with input features managed by the feature extractor and labels by the tokenizer. Input features, already padded and converted to log-Mel spectrograms, are batched into PyTorch tensors using the feature extractor's `.pad` method. Labels are padded to match the maximum length in the batch using the tokenizer's `.pad` method, with padding tokens replaced by -100 to exclude them from loss calculations. Additionally, the initial transcript token is removed from the beginning of the label sequence.
2. **Evaluation Metrics** : To assess the model's performance during evaluation, the Word Error Rate (WER) metric is employed. WER is widely recognized as a standard metric for evaluating Automatic Speech Recognition (ASR) systems. The WER metric is calculated with the following equation :

$$\text{WER} = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C} \tag{3.1}$$

where the variables are shown as follows :
S is the number of substitutions,
D is the number of deletions,
I is the number of insertions,
C is the number of correct words,
N is the number of words in the reference **(N=S+D+C)**.
The implementation of this metric in our fine-tuning process is shown in the pseudo-code below :

Algorithm 6: Compute Metrics Algorithm

Input: Prediction Data (*pred*)
Output: Computed Metrics
Extract prediction IDs and label IDs from *pred*
for *Each element i in label IDs* **do**
  if *label_ids*[*i*] *is* −100 **then**
    Replace *label_ids*[*i*] with the pad token ID
  end
end
Decode prediction and label sequences
Calculate Word Error Rate (WER) as a metric
return *Computed Metrics, including WER*

3. **Define the Training Arguments** : Key parameters related to training are defined, including :
   - `output_dir` : The local directory where model weights will be saved. This directory also serves as the repository name on the Hugging Face Hub.
   - `generation_max_length` : The maximum number of tokens for autoregressive generation during evaluation.
   - `save_steps` : Intermediate checkpoints are saved during training and asynchronously uploaded to the Hub every `save_steps` training step.
   - `eval_steps` : Intermediate checkpoints undergo evaluation every `eval_steps` training steps.

Once the fine-tuning process is complete, the model is evaluated on the test data to validate the effectiveness of the training.

This comprehensive fine-tuning procedure lays the foundation for achieving optimal performance on the task of interest.

3.5.1.3 Post Fine-Tuning Analyses

In this section, we present a detailed analysis of the post-fine-tuning process, which consists of hyperparameter tuning, model training, and evaluation results.

A. Hyperparameter Tuning Prior to initiating the model training, a rigorous hyperparameter tuning process was undertaken. Over a span of 72 hours, extensive efforts were made to optimize the model's hyperparameters for our specific dataset. This meticulous tuning aimed to strike a balance between maximizing performance and ensuring robustness to enhance the model's adaptability to our task. our best hyper-parameters obtained are shown in the following figure :

```
training_args = Seq2SeqTrainingArguments(
    output_dir="./whisper-small-en-sumerizer",
    per_device_train_batch_size=8,
    gradient_accumulation_steps=2,
    learning_rate=1e-5,
    #warmup_steps=500,
    max_steps=1000,
    gradient_checkpointing=True,
    fp16=True,
    evaluation_strategy="steps",
    per_device_eval_batch_size=5,
    predict_with_generate=True,
    generation_max_length=20,
    save_steps=250,
    eval_steps=125,
    logging_steps=2,
    report_to=["tensorboard"],
    load_best_model_at_end=True,
    metric_for_best_model="wer",
    greater_is_better=False,
    push_to_hub=False,
)
```

FIGURE 3.13 – Training Hyper-Parameters

B. Model training and results The model training process was executed with a keen focus on leveraging accelerated training capabilities provided by CUDA and CuDNN on an Nvidia RTX 3060 mobile GPU. Due to constraints imposed by the GPU's 6GB vRAM, we opted for the smaller variant of the Whisper model, containing over 244 million parameters, specifically the English-only model. The training duration extended beyond 24 hours, emphasizing the utilization of the best hyperparameters for maximizing GPU performance. TensorBoard was employed for continuous monitoring of loss metrics, training steps, and Word Error Rate (WER) scores throughout the training phase. as shown in the following Figures :

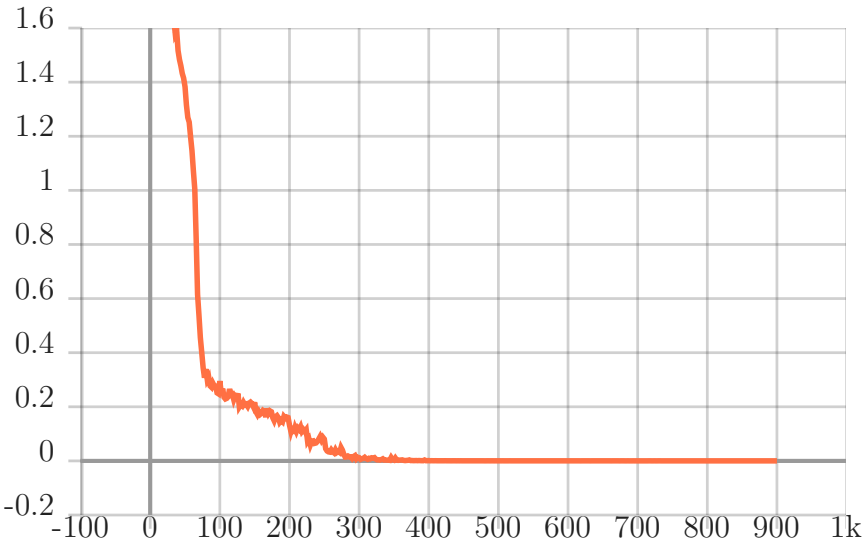


FIGURE 3.14 – Model Loss during Trining

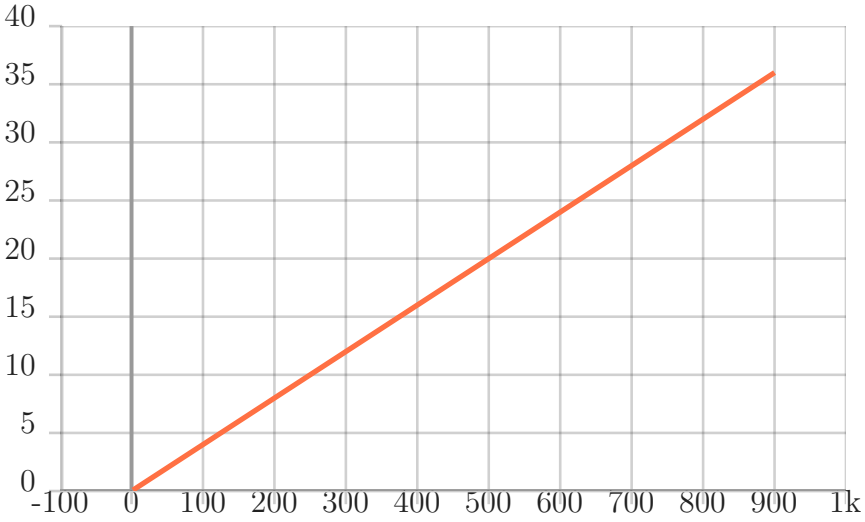


FIGURE 3.15 – Model Training Epochs

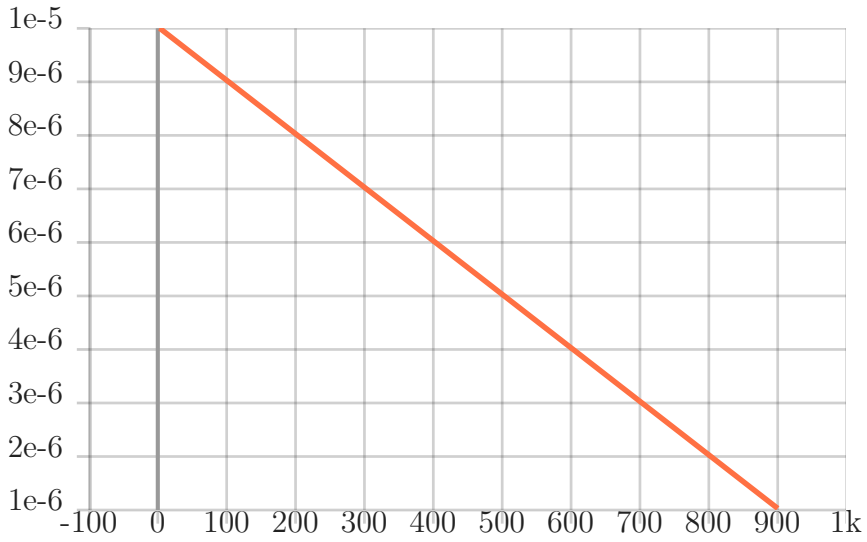


FIGURE 3.16 – Model train learning Rate evolution

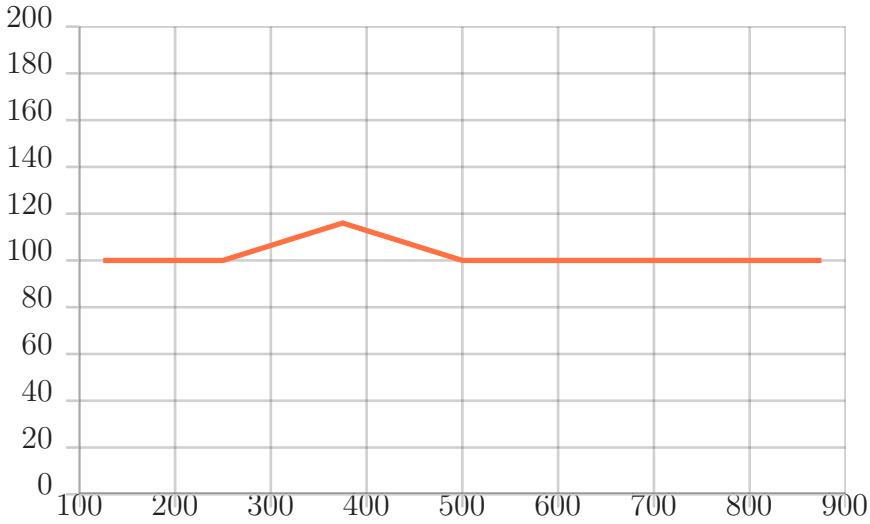


FIGURE 3.17 – Model Evaluation WER during Training

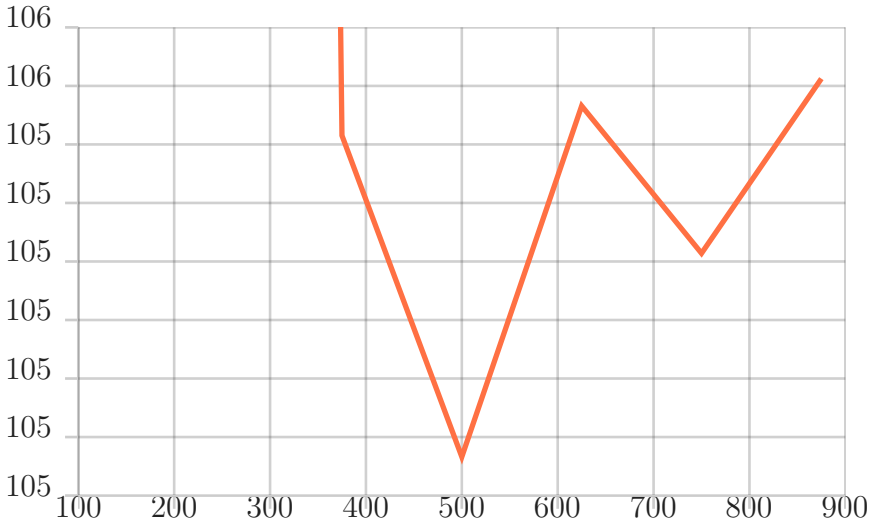


FIGURE 3.18 – Model Evaluation runtime

C. Evaluation Results Upon completing the training process, a thorough evaluation of the model was conducted. The evaluation results indicate that our model exhibits a strong ability to generalize effectively on our dataset and task. Notably, the evaluation WER score, which stands at 100, outperforms the Hugging Face scores on the Common Voice dataset, where the score is 141. This performance is particularly noteworthy considering the average transcription rate of 90 words per minute for audio files that often exceeded 15 minutes in duration.

These results underscore the effectiveness of our fine-tuning process in adapting the model to our specific task, thereby enhancing its overall performance.

3.5.2 Text Summarization Model using Facebook BART

In this subsubsection, we delve into the architecture and rationale behind choosing Facebook BART for our text summarization model. This section is divided into two parts : the architecture of BART and the fine-tuning process.

3.5.2.1 Architecture of BART

BART, as described in the article "BART : Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," is a denoising autoencoder designed to map a corrupted document back to its original form. It employs a sequence-to-sequence model with a bidirectional encoder responsible for processing corrupted text and a left-to-right autoregressive decoder. During pre-training, BART optimizes the negative log-likelihood of the original document.

The architectural components of BART closely follow the Transformer architecture by Vaswani et al. (2017), with some notable modifications. Similar to GPT models, BART replaces ReLU activation functions with GeLUs and initializes parameters from a Gaussian distribution with a mean of 0 and a standard deviation of 0.02. In its base configuration, BART employs six layers in both the encoder and decoder, while the large model utilizes twelve layers in each. Distinct from BERT, each layer of BART's decoder additionally performs cross-attention over the final hidden layer of the encoder, akin to the transformer sequence-to-sequence model. Notably, BERT includes an extra feed-forward network before word prediction, a component absent in BART. In summary, BART boasts approximately 10% more parameters than an equivalently sized BERT model.

3.5.2.2 Pre-training with BART

BART's pre-training is executed by introducing document corruptions and subsequently optimizing a reconstruction loss, which is measured as the cross-entropy between the decoder's output and the original document. What distinguishes BART from other denoising autoencoders is its adaptability to various document corruption types. It accommodates a range of transformations, enabling diverse document corruptions. These transformations include :

- **Token Masking :** Similar to BERT, random tokens are masked and replaced with [MASK].
- **Token Deletion :** Random tokens are deleted from the input, requiring the model to infer the missing positions.
- **Text Infilling :** Text spans are sampled, with their lengths drawn from a Poisson distribution. Each span is replaced with a single [MASK] token, including 0-length spans.
- **Sentence Permutation :** Documents are divided into sentences based on full stops, and these sentences are shuffled randomly.

- **Document Rotation :** A token is uniformly chosen, and the document is rotated so that it begins with that token, training the model to identify the document's start.

These transformations equip BART to handle a wide array of document noise scenarios and offer the potential for further exploration of new corruption types.

The versatile architecture and robust pre-training approach of BART make it a compelling choice for our text summarization model.

3.5.2.3 Fine-Tuning BART in our approach

In our approach, we recognize that the Facebook BART model, owing to its substantial size and complexity, poses challenges for fine-tuning, especially when access to large datasets is limited. The need for substantial data and computational resources makes it a daunting task for most researchers. Fortunately, given the model's power and its availability for several years, numerous research teams have taken the initiative to create fine-tuned versions of BART tailored to specific tasks.

For our project, we adopted the **Facebook BART-CNN** version, which has been meticulously fine-tuned on the **CNN/DailyMail Dataset**. This dataset is an extensive collection of English-language news articles sourced from CNN and the Daily Mail. It contains over 300,000 unique news articles written by professional journalists. While the dataset originally served purposes like machine reading and comprehension, it has also been adapted for both extractive and abstractive summarization tasks, including abstractive question answering.

As of January 2023, the Facebook BART-CNN model, fine-tuned on the CNN/DailyMail Dataset, serves as an ideal candidate for our text summarization project. Leveraging this fine-tuned version, we are able to harness the powerful capabilities of BART for generating high-quality abstractive summaries.

3.6 System Development

In this section, we elucidate the design and development of our system, which plays a pivotal role in transforming our research into a practical application. Our system is conceived as a web application dedicated to audio summarization. The system architecture prioritizes user-friendliness and efficiency, ensuring that users can obtain desired audio summaries with minimal complexity.

3.6.1 Web Application Architecture

After extensive research and exploration of contemporary development patterns and technologies, we decided to implement our system as a web application. The core processing takes place on the server side, where audio data is collected from the web interface and routed through an API to the pipeline of machine learning models responsible for audio-to-text transcription and summarization.

To build the web application, we chose to utilize the **Streamlit** library, a popular choice for creating interactive web applications with Python. Streamlit offers a simple yet powerful framework for developing data-driven applications, aligning with our goal of providing a user-friendly interface for our system.

3.6.2 User-Centric Design

The central focus of our system's design is to provide a user-centric experience. We aimed to minimize complexity and streamline the user interaction process, ensuring that

even users with minimal technical expertise can effortlessly utilize the application.

The user interface is designed with a minimalist approach, emphasizing clarity and ease of use. To generate audio summaries, an end user is only required to upload an audio file and initiate the process with a single button click. here is the system design in the figure below :

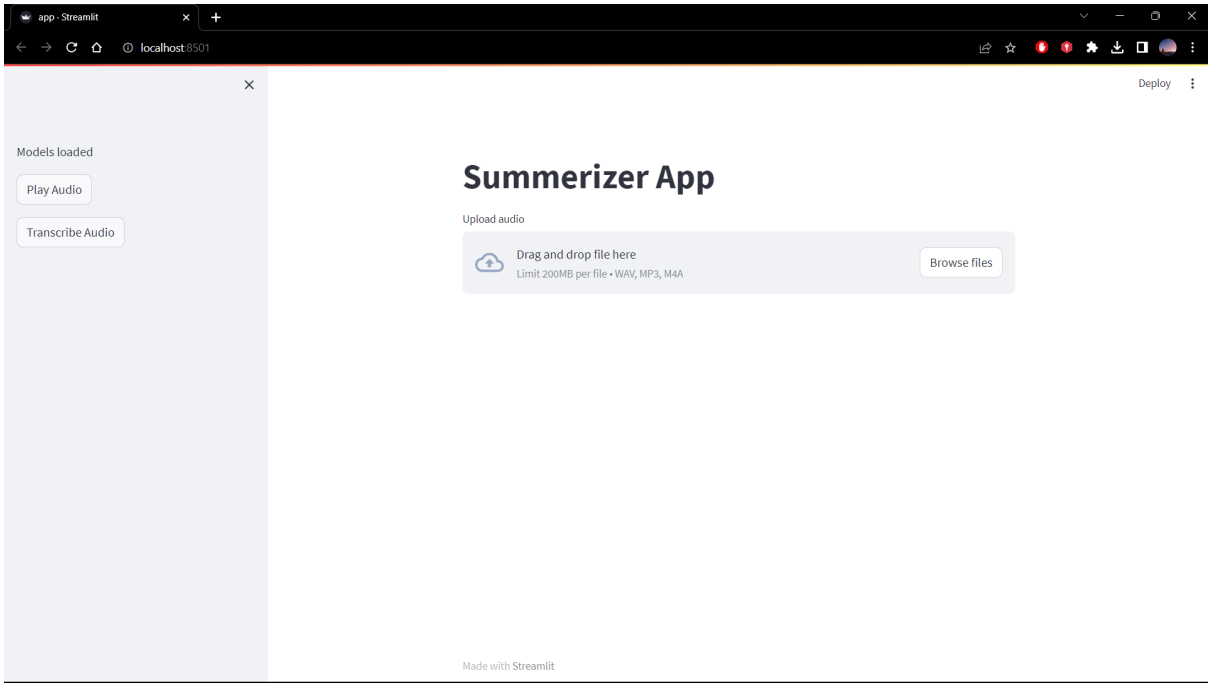


FIGURE 3.19 – System Interface

3.6.3 System Workflow

The workflow of our system is meticulously designed to optimize performance and minimize redundancy. Here is a breakdown of the key steps within our system :

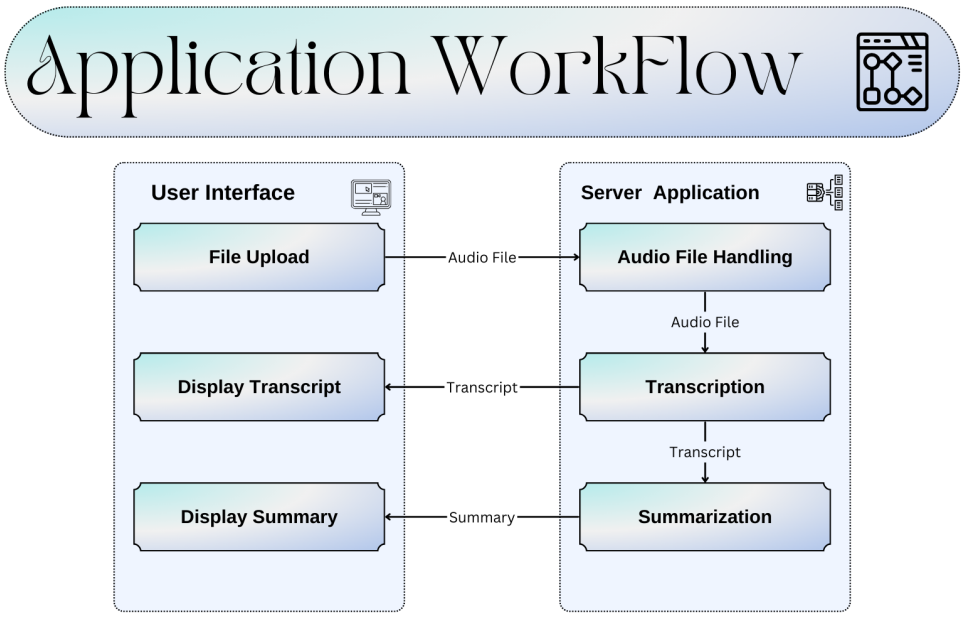


FIGURE 3.20 – System Workflow

3.6.3.1 Audio File Handling

Upon receiving an audio file from the user, the system creates a temporary instance on the server side. This temporary instance serves to minimize network consumption and runtime by avoiding redundant processing of the same audio file if uploaded again.

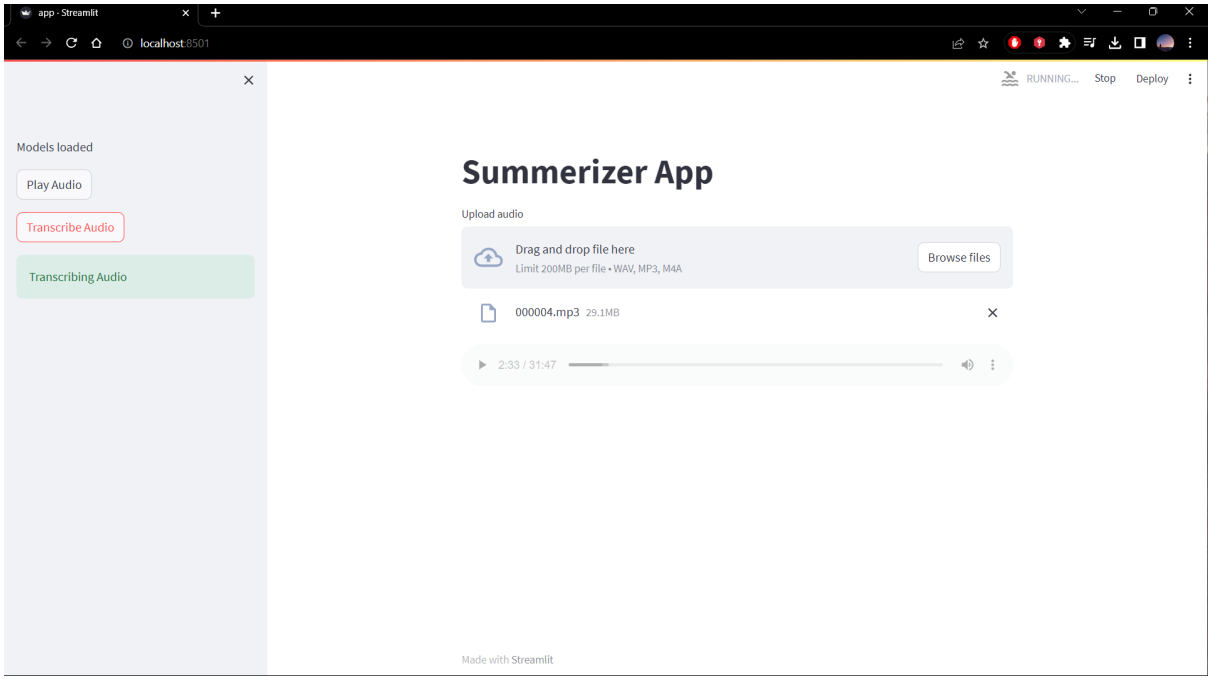


FIGURE 3.21 – System Audio Loading Interface

3.6.3.2 Transcription

The preprocessed audio is passed through our transcription pipeline, where our advanced speech recognition models, such as the Fine-tuned Whisper model, convert the spoken content into textual form. This transcription is a crucial step in generating audio summaries.

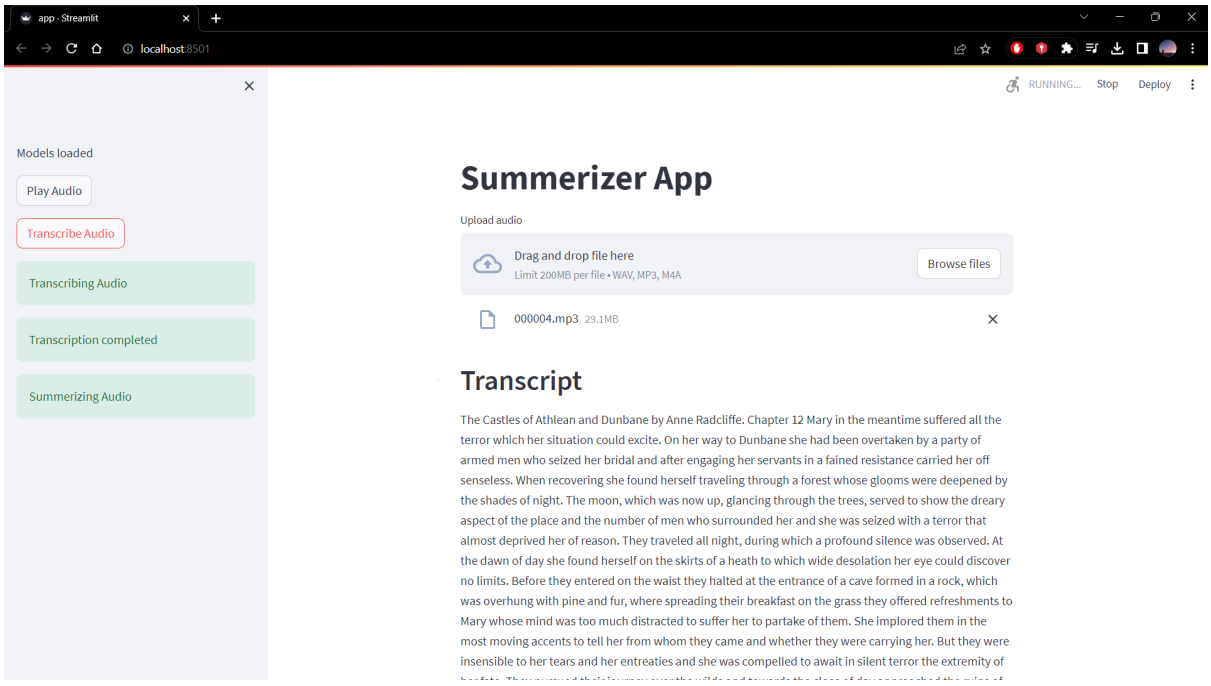


FIGURE 3.22 – System Transcript Display Interface

3.6.3.3 Summarization

The transcribed text is then forwarded to our BART model for summarization. This step ensures that the user receives a concise and informative summary of the audio content, which can be invaluable for various applications.

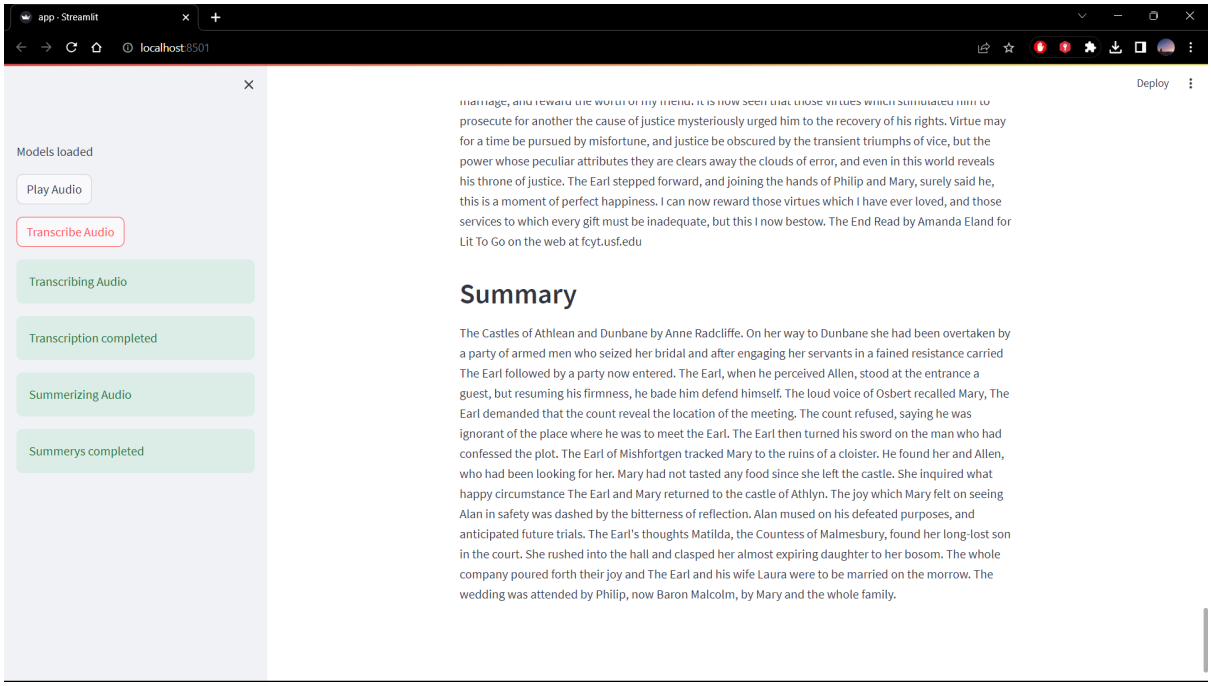


FIGURE 3.23 – System Summary Display Interface

3.6.4 Data Handling and Efficiency

In the development of our system, meticulous attention is given to data handling to ensure seamless operation without exceptions. This includes efficient data storage, retrieval, and processing to deliver timely audio summaries to end users.

3.6.5 Discussion

Our system represents a significant advancement in audio summarization technology. By automating the process of generating concise textual summaries from audio inputs, we offer a solution that can be valuable in numerous domains, such as content indexing, accessibility, and content review.

In the era of rapidly evolving technology, our system exemplifies the potential for artificial intelligence and machine learning to enhance audio content understanding and accessibility. The user-friendly design makes our system accessible to a wide range of users, promoting its practical application in real-world scenarios.

In conclusion, the development of our web application, tailored for audio summarization, serves as a testament to the fusion of cutting-edge technology and user-centered design. It holds the promise of making audio content more accessible and comprehensible, ultimately benefiting various industries and users.

3.7 Conclusion

In this concluding chapter, we delved into the design and implementation of our audio summarization system, underpinned by transformer-based models and transfer learning. This chapter provided insights into the system's architecture, dataset handling, model fine-tuning, and development process.

Our system represents a culmination of extensive research and development efforts, aiming to harness the power of deep learning to transform audio data into concise and informative summaries. With a user-friendly web application interface, our system minimizes complexity, making audio summarization accessible to a wide audience.

The combination of data collection, model fine-tuning, and streamlined workflow showcased in this chapter embodies the practical application of cutting-edge technologies to real-world challenges. Our work holds the potential to enhance audio content understanding and accessibility across various domains.

As we progress into the final chapter of this thesis, we will present our findings and evaluate the performance of the system, shedding light on the impact and effectiveness of our approach in the realm of audio summarization.

GENERAL CONCLUSION AND FUTURE WORK

Conclusion

In this thesis, we embarked on a journey to harness the power of transformer-based models and transfer learning for the task of audio summarization. The rapid growth of audio data across various domains has underscored the need for efficient methods to distill valuable insights from audio content. Our research has focused on addressing this need through the development of an advanced audio summarization system.

Through rigorous exploration and experimentation, we successfully crafted a system that employs transformer-based models to extract salient information from audio recordings and generate coherent and concise summaries. By leveraging transfer learning techniques, we optimized the performance of our system even with limited audio data, making it adaptable and versatile.

Our commitment to user-friendliness led us to create a web-based application that simplifies the audio summarization process, ensuring accessibility to a wider audience. We have demonstrated the real-world impact of our work by showcasing the significance of audio summarization in diverse applications, ranging from content indexing to accessibility for the hearing-impaired.

Future Work

While we have made substantial strides in the field of audio summarization, there remain promising avenues for future research and enhancements. These include :

- End-to-End Audio Summarization : A prospective development is the creation of an end-to-end audio summarization system that streamlines the entire process using a unified model. This would enhance efficiency and usability further.
- Exploring Other Transformer Architectures : The field of transformers continues to evolve. Future work may involve exploring and comparing various transformer architectures to ascertain their effectiveness for audio summarization tasks.
- Multimodal Summarization : Expanding our system's capabilities to handle both audio and text data for multimodal summarization is a promising avenue. This would enable more comprehensive insights by synthesizing information from multiple sources.
- Real-time Summarization : As audio data often originates from live events or streaming sources, future research can focus on real-time audio summarization techniques, enabling timely insights from ongoing events.

BIBLIOGRAPHIE

[1] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pages 28492–28518. PMLR, 2023.

[2] Kdd in data mining assists data prep for machine learning | techtarget. `www.techtarget.com/searchenterpriseai/feature/KDD-in-data-mining-assists-data-prep-for-machine-learning`.

[3] Arthur L Samuel. Machine learning. *The Technology Review*, 62(1) :42–45, 1959.

[4] Hodhaifa Abdelghani BARAKA. *Étude sur les Méthodes d'Optimisation Utilisée dans l'Apprentissage Automatique*. PhD thesis, Directeur : Mr. RIMOUCHE Ali/Co-directeur : Mme. HANDOUZI Wahida, 2020.

[5] How does machine learning work - javatpoint. `www.javatpoint.com/how-does-machine-learning-work`.

[6] Julien Ah-Pine and Anne-Françoise Yao. Une approche par noyaux multiples pour l'apprentissage non-supervisé de représentation de données fonctionnelles dans des espaces de sobolev.

[7] Linear regression in machine learning - javatpoint. `www.javatpoint.com/linear-regression-in-machine-learning`.

[8] Machine learning polynomial regression - javatpoint. `www.javatpoint.com/machine-learning-polynomial-regression`.

[9] K-nearest neighbor(knn) algorithm for machine learning - javatpoint. `www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning`.

[10] Support vector machine (svm) algorithm - javatpoint. `www.javatpoint.com/machine-learning-support-vector-machine-algorithm`.

[11] Nilkanth Deshpande, Shilpa Gite, and Rajanikanth Aluvalu. A review of microscopic analysis of blood cells for disease detection with ai perspective. *PeerJ Computer Science*, 7 :e460, 04 2021.

[12] Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer, 1996.

[13] Introduction to random forest in machine learning | engineering education (enged) program | section. `www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/`.

[14] Tianqi Chen and Carlos Guestrin. Xgboost : A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[15] Zhuo Wang, Jintao Zhang, and Naveen Verma. Realizing low-energy classification systems by implementing matrix multiplication directly within an adc. *IEEE Transactions on Biomedical Circuits and Systems*, 9 :1–1, 12 2015.

[16] Xgboost - geeksforgeeks, NaN. `www.geeksforgeeks.org/xgboost/`.

[17] Alan Jeffares. Supervised vs unsupervised learning in 2 minutes, Sep 2020. `towardsdatascience.com/supervised-vs-unsupervised-learning-in-2-minutes-72dad148f242`.

[18] Li Deng, Dong Yu, et al. Deep learning : methods and applications. *Foundations and trends® in signal processing*, 7(3–4) :197–387, 2014.

[19] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1) :1–127, 2009.

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

[21] Jürgen Schmidhuber. Deep learning in neural networks : An overview. *Neural networks*, 61 :85–117, 2015.

[22] Vidyaesampally1998 Vidya. Artificial neural network v/s biological neural network, Feb 2021. `vidyaesampally1998.medium.com/artificial-neural-network-v-s-biological-neural-network-a0862d12e9a8`, journal=Medium.

[23] Andy Betts. Going deep with deep learning : Martech insights, action amp ; impact, Oct 2021. `martech.org/going-deep-deep-learning-martech-insights-action-impact/`.

[24] Timothée Lesort. Continual learning : Tackling catastrophic forgetting in deep neural networks with replay processes. 07 2020.

[25] Nikhil Buduma. *Fundamentals of Deep Learning : Designing next-generation Artificial Intelligence Algorithms*. O'Reilly, 2017.

[26] Valentino Zocca, Gianmario Spacagna, Daniel Slater, and Peter Roelants. *Python deep learning : Next generation techniques to revolutionize computer vision, AI, Speech and Data Analysis*. Packt Publishing, 2017.

[27] Ping-Huan Kuo, Ssu-Ting Lin, and Jun Hu. Dnae-gan : Noise-free acoustic signal generator by integrating autoencoder and generative adversarial network. *International Journal of Distributed Sensor Networks*, 16 :155014772092352, 05 2020.

[28] Pat Nakamoto. *Neural Networks amp ; Deep Learning : Explained to your granny*. Createspace Independent Publishing, 2017.

[29] Single layer perceptron in tensorflow - javatpoint. `www.javatpoint.com/single-layer-perceptron-in-tensorflow`.

[30] Marc Parizeau. Réseaux de neurones. *GIF-21140 et GIF-64326*, 124, 2004.

[31] Deep learning via multilayer perceptron classifier - dzone. `dzone.com/articles/deep-learning-via-multilayer-perceptron-classifier`.

[32] Valentino Zocca, Gianmario Spacagna, Daniel Slater, and Peter Roelants. *Python deep learning*. Packt Publishing Ltd, 2017.

[33] Andrea Volpini. How to build a keyword suggestion tool using tensorflow, Mar 2021. `wordlift.io/blog/en/keyword-suggestion-tool-tensorflow/`.

[34] basic-cnn-architecture, Oct 2022. `www.upgrad.com/blog/basic-cnn-architecture/`.

[35] ODSC Community. Understanding the mechanism and types of recurrent neural networks, Mar 2021. opendatascience.com/understanding-the-mechanism-and-types-of-recurring-neural-networks/.

[36] David Vint, Matthew Anderson, Yuhao Yang, Christos Ilioudis, Gaetano Di Caterina, and Carmine Clemente. Automatic target recognition for low resolution foliage penetrating sar images using cnns and gans. *Remote Sensing*, 13 :596, 02 2021.

[37] Autoencoder structure. www.upload.wikimedia.org/wikipedia/commons/2/28/Autoencoder_structure.png.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[40] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv :1607.06450*, 2016.

[41] Raul Monteiro and Diogo Pernes. Towards end-to-end speech-to-text summarization. *arXiv preprint arXiv :2306.05432*, 2023.

[42] Ayham Alomari, Norisma Idris, Aznul Qalid Md Sabri, and Izzat Alsmadi. Deep reinforcement and transfer learning for abstractive text summarization : A review. *Computer Speech Language*, 71 :101276, 2022.

[43] Yu Zhang, Wei Han, James Qin, Yongqiang Wang, Ankur Bapna, Zhehuai Chen, Nanxin Chen, Bo Li, Vera Axelrod, Gary Wang, et al. Google usm : Scaling automatic speech recognition beyond 100 languages. *arXiv preprint arXiv :2303.01037*, 2023.

[44] Zheheng Luo, Qianqian Xie, and Sophia Ananiadou. Chatgpt as a factual inconsistency evaluator for abstractive text summarization. *arXiv preprint arXiv :2303.15621*, 2023.

[45] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart : Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv :1910.13461*, 2019.

[46] Python. www.python.org/.

[47] Pytorch Team. Pytorch : Deep learning for humans. https://pytorch.org/.

[48] streamlit. https://streamlit.io/.

[49] Huggingface. https://huggingface.co/.

[50] Numpy. www.numpy.org/.

[51] Project jupyter. www.jupyter.org/.

[52] Visual studio code - code editing. redefined. www.code.visualstudio.com/.

[53] Anaconda. https://www.anaconda.com/.