A Multimodal Approach: Acoustic-Linguistic Modelling for Neural Extractive Speech Summarisation on Podcasts

by

Berk Calik

A thesis submitted to
the Department of Computer and Information Science
in partial fulfillment of the requirements for
the degree of

Master of Science in Business Informatics - Applied Data Science

1st Supervisor: Marijn Schraagen
2nd Supervisor: Heysem Kaya

Universiteit Utrecht

Department of Computer and Information Science

Utrecht University
June 2022

Abstract

Podcasts, a contemporary medium of audio-only content, have rapidly progressed in consumption and generation across the internet. Along with the accelerated pace of its popularity in recent years, effectively publicising the podcast shows has become a need for all podcast creators, listeners and streaming platforms. To improve the overall visibility of podcast contents and enhance user engagement, a summary of an episode has become a need for the users or utilising in searching and recommendation systems, which can be a replacement of or in addition to keywords, manual descriptions and transcripts.

Since manual summarisation for podcast episodes takes ample time, automatic summarisation becomes a valuable task. Specifically, in the context of automatic summarisation task for spoken documents, we need to consider that the extracted salient information relies on what is said but also how it is said. In the wake of this, this thesis investigates summarisation models for podcasts and proposes a multimodal approach exploiting acoustic and linguistic features. Accordingly, we aimed to explore how to automatically generate an extractive summary from a podcast episode in a multimodal way. For our research, we have employed a lexical-only pre-trained transformer model (i.e. SentenceBERT) for embedding sentences in the transcripts. In this work, speech summarisation of podcasts is defined as a classification task; in respect to that, our purpose was to extract meaningful sentences from the transcribed text where the importance is predicted by combining acoustic and linguistic information. To build an experimental setup for analysing the impact of acoustic features, we have integrated a two-layer multilayer perceptron on the top layer of the SentenceBERT model. Feature projection, ranking and selection were also performed for feature importance analysis of acoustic information. After projection and selection of acoustic features, our proposed multimodal model outperforms the baseline (text-only) and achieved moderately better ROUGE scores; with this project, we aim not to find a complete solution for the automatic summarisation of podcast episodes, but to understand the critical part of the puzzle linked to incorporating acoustic features into podcast summarisation.

Keywords: Podcast Summarisation; Speech Summarisation; Multimodality; Extractive Summarisation; Acoustic-Linguistic features

Acknowledgement

The outcome of this research required a lot of guidance, and I feel incredibly fortunate to have gotten this all along from great minds while completing this work. Specifically, I would like to express my deepest gratitude to Dr. Marin Schraagen, my first research supervisor, for his valuable suggestions and helpful critiques during the planning and development of this thesis, which wouldn't be possible without his constructive guidance, and his contribution are gratefully acknowledged. My grateful gratitude is also extended to Dr. Heysem Kaya, my second research supervisor; his willingness to give his time to enlighten me and expand my knowledge in the field so generously has been much appreciated and truly acknowledged.

Having an idea and turning it into something tangible can be as demanding as it sounds. I am forever grateful for my dear mom and dad, whose unconditional love and constant support keep me hopeful and motivated. I would like to extend my heartfelt thanks to my beloved sibling Baris and brother-like cousin Mert; I owe my deepest gratitude to them and am forever thankful for being always supportive throughout my adventures. My accomplishments and success are because my family believed in me and has given me hope; without hope, this thesis would not have been possible. Lastly, to all those who have been a part of my journey and getting here, my great friends worldwide, who keep me grounded and make me feel at home wherever I go, you have my honest gratitude as we keep trying to find our authentic selves together in this journey called life.

Table of Contents

Chapter 1

Introduction

1.1 Introduction

Along with the extensive generation of spoken audio documents on the internet, podcasts, one of the mediums of recorded spoken audio, have become widespread in recent years with an accelerated consumption rate. Whereas at present, people are overwhelmed by the abundance of information from different multimedia resources. Podcast summary, an essential factor for affecting end-users listening decisions, has often been considered a critical feature in podcast recommendation systems and many downstream applications [107]. Such summaries would inform listeners about the podcast episode and convey all-important attributes (e.g. topical content, genre, and participants), triggering the desire to use efficient systems. Eventually, these frequent release schedules of new podcast episodes would make the manual production of summaries impractical [13]. Therefore, automatic summarisation systems have become a need for addressing this issue, and the work is becoming more critical on the existing automatic summarisation systems and reforms them to meet the demands based on user needs.

Automatic summarisation approaches are generally categorised into three: Extractive, Abstractive and Hybrid. As the name indicates, extractive ones solely focus on extracting essential sentences, phrases or segments from a document or set of documents. The abstractive approaches aim to rewrite the content more precisely, and the hybrid approach stands for a combination of abstractive and extractive approaches. The summarisation task for spoken documents contains non-linguistic information, mainly expressed by prosody, while written text carries only linguistic information. Podcast summarisation brings a unique set of challenges compared to similar speech summarisation tasks (e.g. broadcast news [9], meeting transcripts [89]). In such a manner, podcast datasets have a wider variety of structural formats (i.e. monologue, interview, conversation, debate, documentary etc.), level of formality, number of speakers, genre, subject matter, speaking style and geographical background [13]. Hence, these contents add another room for research on information retrieval tasks, specifically on automatic summarisation through conversational characteristics of the spoken texts.

This thesis is influenced by the task curated in Textual Retrieval Conference [13] (TREC) 2020 and inspired by two papers: Alexander et al. leverage the acoustic

features for Spotify podcast episodes [1] that are provided to promote utilising linguistic and acoustic features for detecting salient part of the spoken audio. Secondly, the paper reported by Inoue et al. [38] accredits that prosody plays an important role in expressing non-linguistic information such as intension, topic change, and emphasising words or phrases. Hence, we are motivated to investigate the potential that the quality of speech summarisation can be improved by introducing prosodic parameters.

In this work, speech summarisation of podcasts is defined as extracting meaningful sentences from the transcribed text where the importance is predicted by combining acoustic and linguistic information. Accordingly, this research aims to investigate how to automatically generate an extractive summary from a podcast episode in a multimodal way. For our research, we have employed a lexical-only pre-trained transformer model (i.e. BERT) and built an experimental setup for analysing the impact of acoustic features from podcast audio on the summarisation task. Hence, we hypothesise that a multimodal approach of extractive summarisation improves the quality of produced podcast summaries compared to the lexical-only models. We will present the gap in the literature where there is a need for utilising acoustic features for podcast summarisation. Eventually, we seek to generate a textual summary for the episodes from the podcast dataset provided by Spotify [13] based on our proposed technique over the baseline models, which possibly empower the impact of acoustic features in recommendation systems and are practical for users who want to have insight when deciding whether to listen to the podcast episodes.

The document is structured as follows: Chapter 2 introduces the comprehensive background, starting with a subsection of the Natural Language Processing (NLP) field; its related sub-topics and exploited concepts and techniques are discussed. This section follows a sub-section on deep learning, in which the neural models are used for this thesis. After that, recent speech summarisation tasks related to our work are introduced. Chapter 3 covers the overall methodology for our proposed work detailed with our ensemble model; speech features and feature projection and selection methods come along with the section. In the following, Spotify Podcast Dataset and pre-computed acoustic features are being explored. After exploring the preprocess results and experimental setup, we dive into findings with results and discussion in Chapter 4.

1.2 Motivation: TREC Podcast Track

With the recent growth in popularity, podcasts bring a great opportunity and unique challenges to existing content discovery and recommendation systems. According to a recent study by Podcast Insights[74], there are currently over 2 million podcast shows and more than 48 million episodes in the world. Consequently, to encourage research in podcast retrieval and approach by researchers from the information retrieval, the NLP and speech analysis fields, a TREC podcast track was first organised in 2020 and carried on with minor changes in 2021.

The Podcast Track at TREC has two challenging tasks: segment retrieval and podcast episode summarisation. The segment retrieval task in TREC 2020 consisted of retrieving the two-minute-long segment from the complete podcast dataset, given

a query that could come in finding a segment of a particular topic, re-finding a segment or information about a known item. The summarisation task in TREC 2020 consisted of generating a relevant and informative textual summary as short as the description of a given podcast. In this project, we plan to focus on the summarisation task. Apart from improving the Speech-of-Text generated transcripts in 2020, none of the participants used the audio data of the podcasts for the tasks [41]. In preference, all participants used Speech-of-Text generated transcripts exclusively as the topical relevance is expected to be primarily and mainly induced by the spoken words within each segment. Nevertheless, there is a great potential for exploiting heterogeneous features specifically acoustic information for podcast summarisation since the sentences, phrases or segments in podcasts might be important not only based on what is said but also on how it is said [60]. Thus, we plan to enhance the research for understanding the impact of acoustic features in the speech summarisation problem through detecting salient segments of the podcast episodes in a multimodal way. Along with that inspiration, Alexander et al. [1] provided pre-computed acoustic features of each podcast and made them readily available to encourage the participants in TREC 2021. Thus, we will be driven by the earlier papers and available data to improve the prediction of sentence importance by utilising both acoustic and linguisic information of the podcasts.

1.3 Research Goal: Multimodal Summarisation

Granting that various successful podcast summarisation techniques have been proposed in recent years, there is still a considerable gap between the quality of automatic spoken audio summarisation and manual one by humans and that brings a great potential for utilising audio signals to improve these models.

In this framework, the problem of speech summarisation becomes automatic scoring of sentence importance for the transcribed text from podcast episodes. The proposed system will be responsible for generating an extractive summary where the multimodal analysis will be executed on audio and transcripts of the episodes. Subsequently, the selected segments potentially carry the podcast episode's salience and topic relevance will be based on the explanations, discussions over topics and expression by the speakers through textual and non-textual information. Consequently, we aim to investigate whether the importance of what is said is also carried out by how it is said. To understand the "how it is said" part, pre-computed features from the audio file will be investigated, as it has been reported that they can cover speaker features (e.g. speaker's intention, pitch, emotion, mood) and the utterance (e.g. statement, question) [3].

With this project, we do not aim to find a complete solution for the automatic summarisation of podcast episodes but to understand the critical part of the puzzle conceptually. In particular, the focus is on determining whether acoustic features derived at the sentence-level can be effectively represented and used to estimate important segments in speech, and eventually whether these acoustic information combined with lexical features can improve speech summarisation performances in podcasts episodes.

Conclusively, the project aims to investigate the following hypotheses:

- When it comes to speech summarisation task, the saliency of information exists not only in the textual data but also in the audio signals, as the importance of what is said correlates with how it is said in the spoken audio contents.
- The acoustic-linguistic multimodal approach can improve the performance of extractive summarisation models compared to the performance of lexical-only models.

In this regard, there are a main and three sub-research questions that we plan to answer with this work:

How to incorporate acoustic and linguistic information for multimodal analysis in the state of the art models that can cultivate the performances of extractive summarisation systems for podcasts?

1. Which combination technique is to utilise for multimodal analysis of heterogeneous information in neural summarisation models that can assist in detecting salient parts of the speech in mono/multi-agent dialogues?
2. To what extent do the selection, projection and representation of acoustic features improve the accuracy of the extractive summarisation systems?
3. In which way to normalise acoustic features effectively so that speaker-neutral representations can be utilised for podcast summarisation models?

1.4 Spotify Podcast Dataset

The Spotify Podcast Dataset[1] is a collection of 100,000 English-language podcast episodes provided by Spotify AB, an audio streaming and digital media service provider founded in 2006 by Daniel Ek and Martin Lorentzon [13]. Spotify is currently the largest music and podcast streaming service provider, with over 381 million monthly active users, including 172 million paying subscribers. According to a survey[74] curated by Spotify, there have been 48 million podcast episodes as of April 2021.

Spotify has released the 100,000 podcast dataset to promote research into the emerging field of this spoken audio and proposed a task that has been conducted at TREC 2020. Accordingly, the scheme as a podcast track covers summarisation and segment retrieval tasks. The TREC workshop series encourages research in information retrieval and related application by providing a large set collection, uniform scoring procedures, and a forum for organisations interested in comparing their results. The podcast dataset sampled episodes from professional and amateur podcasts representing a wide range of audio quality, topics, and structural formats. Each of the podcasts episodes in the dataset includes an audio file, a text transcript and some associated metadata. Accordingly, the thesis focuses on dealing with the input sequences of the transcriptions aligned with audio clips of the episodes. The project will perform on the domain of transcribed text from podcasts and their audio signals, which carries sub-problems similar to speaker normalisation, discussion summarisation, segmentation, saliency detection etc.

[1] https://podcastsdataset.byspotify.com/

Chapter 2

Background

Overview

This chapter contains comprehensive background, starting with NLP and respective language data processing techniques are explained in section 2.1 and deep learning and utilised neural networks for our work are explained in the section 2.2. After that, section 2.4 briefly explains summarisation approaches related to our work.

2.1 Natural Language Processing

Natural Language Processing (NLP) is an interdisciplinary field of study with influences from computer science, artificial intelligence, and linguistics [29]. NLP includes Natural Language Understanding (NLU) and Natural Language Generation (NLG), with the aim of giving computers the ability to understand the text and spoken words and performing tasks that involves languages with the same accuracy as humans can have. In the form of human languages, which naturally evolve and do not follow a certain set of clear rules, we speak and communicate ambiguously. For instance, "Okay" can be used multiple times in a conversation, but the intentions and meanings can vary in different sentences. Suppose we want our machines or computers to be accurate with natural languages. In that case, we need to facilitate them with a certain set of rules and must consider various other factors, such as grammatical structure, semantics, sentiments, and the influence of past and future words.

NLP has been incorporated into the statistical, machine and deep learning models where these technologies give more power to analyse the human language in the text, sign, audio, and video data and capture its whole meaning, including the speaker or writer's intention and sentiment. Various NLP tasks break down the human text and voice data that enable handling use cases such as machine translation, virtual agents, social media sentiment analysis, and text summarisation[48]. Accordingly, some NLP tasks such as speech recognition, word sense disambiguation, named entity recognition, sentiment analysis, and natural language generation are required to drive the force behind machine intelligence [29]. Processing any spoken or textual natural language data is dissimilar to processing numerical datasets. To analyse any spoken or textual document, it has to be split into meaningful pieces

and reconstructed to a numerical representation so that they can be used for input to the machine learning algorithms. Consequently, the algorithms encounter difficulties such as how to encode the position of a word relative to the other words, how to assess word importance in the complete set of documents, and words out of vocabulary(OOV) [94]. Likewise, we will need to employ various NLP techniques to explore podcasts' textual and audio data for our project's utilisation in deep learning models.

2.1.1 Language Data Processing

2.1.1.1 Corpus and Dictionary

In linguistics and NLP, corpus denotes a collection of texts, a set of documents that helps to analyze the text data, which is curated and represented as a list of tokens after some preprocessing steps. Accordingly, we need to transform each token into a numerical representation to analyse as a statistical and machine learning corpus. The transformation process to numerical representation carries out a unique integer ID for each unique token in the corpus. Vocabulary V stands for mapping these unique integer IDs with unique tokens.

2.1.1.2 Text Cleaning and Filtering

Text data or transcripted data from audio is often unstructured, noisy and of varying quality and as such there are many different techniques to clean and enhance the training dataset before converting them to their model input representations. Filtering in the preprocess denotes removing or keeping the data points based on various specifications. For instance, removing input which seems too short to make sense, or the documents in different languages the model cannot handle should be removed, or setting through rules such as length (input) > 10 can improve the input quality as well as the performance of models[91].

2.1.1.3 Removal of Noise: Punctuation Marks, Stop-Words

Punctuation marks are considered noisy terms in the text. So, removing them is very helpful before executing most NLP tasks[24]. Stop-words frequently occur in any language which does not add much meaning to a sentence (e.g. pronouns, prepositions, auxiliary verbs, and determiners). Although, we need to ignore or take them into account based on the cases and tasks. Let us say we have to perform a task such as language classification, spam filtering, caption generation, sentiment analysis, or something related to text classification. Then we can ignore them as they do not provide crucial information for these models. On the other hand, if our task contains text generation such as machine translation, question-answering problems, and text summarisation, it is better to keep the stop words as they carry a crucial part of these applications. For our work, we will keep them as we seek to generate summaries from selected sentences.

2.1.1.4 Tokenisation

Tokenisation, also known as word segmentation, is a linguistic technique for splitting given text into meaningful pieces or tokens that can be used for further analysis[63].

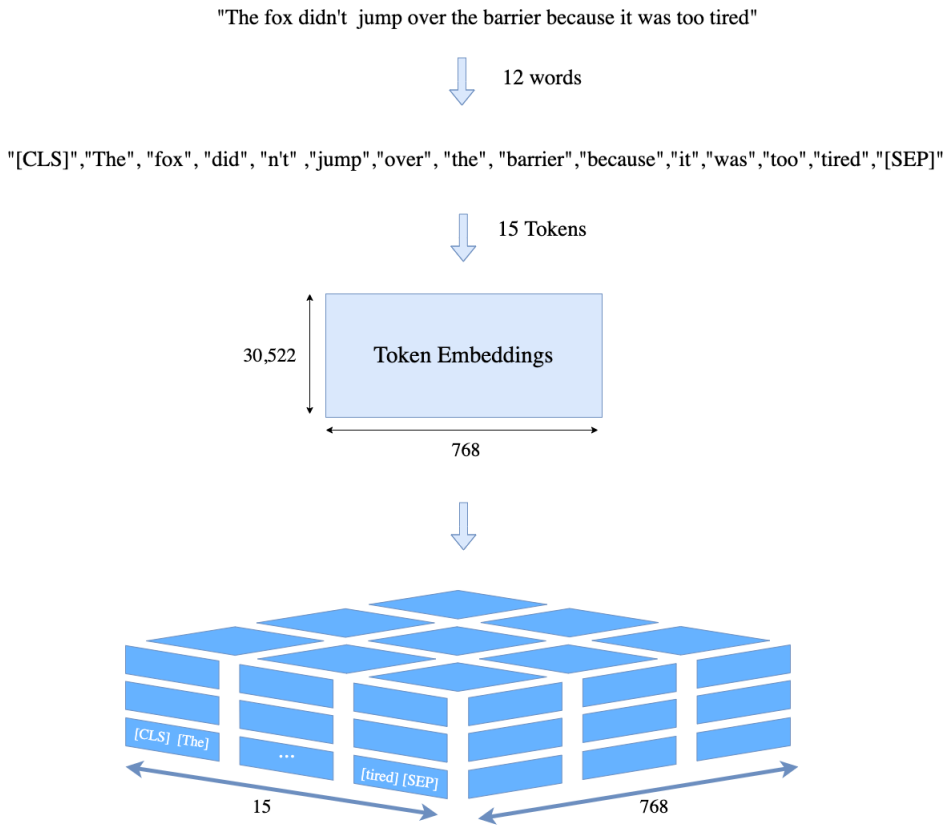

Figure 2.1: An illustrated example for tokenisation and embeddings for a sentence

Naturally, a text can be split into white spaces and punctuation such that tokens can be words, numbers or punctuation marks [28]. However, this technique brings a couple of challenges. For instance, a language model should know how to handle and split apostrophes, such as “shouldn’t” and abbreviations like “e.g.” or specific words used semantically together. In such a manner,“New” and “Zealand” might bring another semantic meaning if taken separately. Hence the concern arises when deciding on how to tokenize text is a language-specific task and is mainly handled by characteristic heuristic rules curated for the language analysis.

2.1.1.5 Embeddings

While searching for a numerical but meaningful representation for words, we need to find an optimal method that helps us to handle the curse of high dimensionality, introduced by Bellman [6]. He indicates that phenomenon appears when analyzing data in high-dimensional spaces, which does not take place in low-dimensional settings; in other words, when the error increases with expanding in the number of features, that might indicate suffering from the curse of dimensionality. For instance, one-hot encoding, an entire case of the Bag of Words approach [104], can help detect the importance by frequency counts or binary check. However, the approach suffers from the curse of dimensionality as the words are represented in the whole vocabulary vector space, which requires many unique categories as well as dimensions. Hereupon this leads the dimensionality of the transformed vector to become unmanageable for representation. Moreover, one-hot representation cannot capture the different meanings of the exact words, and similar words are not placed

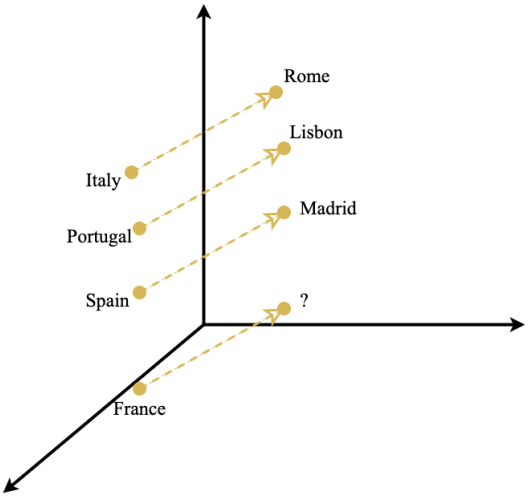

Figure 2.2: Through pre-trained word embedding concepts, such as Word2Vec [62], questions and connections such as "what is to France as Madrid is to Spain" and singular to plural associations can be decomposed by learning the embeddings [12]

closer to each in the encoding space.

Hence, to combat these problems, embeddings came into play for the first time by Salton in 1975 [86]. Embeddings are defined as mapping a discrete - categorical-variable to a vector of continuous numbers. Word embeddings approach enables embodying a word with N elements in an N-dimensional vector rather than one-to-one mapping of the one-hot encoding. This distributed vector representation approach allows machines to understand the words in different senses whereby the semantic concepts can be demonstrated through the interconnection among words or documents. The vector representation of a word or a sentence in a semantic vector space can build these associations and allow machines to understand the relations of contexts such as [62]: $\vec{v}$*(Madrid)* - $\vec{v}$*(Spain)* + $\vec{v}$*(France)* = *?* . Meaning that, let's say sum of the $\vec{v}$(Spain) and a vector leads us to find the vector of $\vec{v}$(Madrid), such that if substract $\vec{v}$(Spain) from $\vec{v}$(Madrid), we will find the vector which leads us finding the capital of a country. Eventually, if we sum this vector with $\vec{v}$(France), we will find the the vector of $\vec{v}$(Paris) represented in the semantic space.

Specifically, to enable machines to discover the similarity among represented documents in a quantifiable form, usually with a vector form, we need to define a way to measure the similarity mathematically. The similarity calculations assist the machines in comparing which documents are most or least similar. Hence, converting a document into a numerical object and defining similarity measures are primarily the two steps required to make models perform this practice. For instance, Cosine Distance/Similarity is one of the most common and effective ways to calculate the similarities among feature vectors. The metric calculates the angular distance between the vectors. To formulate the calculation of cosine similarity between two vectors A and B is defined as follow:

$$S_C(\mathbf{A}, \boldsymbol{B}) := \frac{(\mathbf{A} \cdot \boldsymbol{B})}{\|\boldsymbol{A}\| \, \|\boldsymbol{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{2.1}$$

where A_i and B_i are components of vector $\boldsymbol{A}$ and $\boldsymbol{B}$ respectively. The resulting similarity ranges from –1 meaning exactly opposite, to 1 meaning exactly the same, with 0 indicating orthogonality or decorrelation, while in-between values indicate intermediate similarity or dissimilarity. For instance, when we look at the example illustrated in the figure 2.2, we can state cosine similarity between Rome and Lisbon will have higher than between Rome and some unrelated concepts like "Hogwarts".

Within the neural network concepts, embeddings are low-dimensional and learned continuous vector representations of discrete variables. Similar to the word embedding example aforementioned above, a dense vector of lower dimensionality could be an example for an embedding layer of numerical features [62]. In neural networks, the approach can be combined with the pre-trained word embeddings, where the embedding layer can be randomly initialized or trained on language data itself. Consequently, the embedding layer is then updated by, for instance, backpropagation technique[32] that can re-learn the representation of documents based on a particular task [102]. Hence, if a dataset contains both a set of highly-dimensional categorical variables and numerical variables, we need to use an embedding layer to embody the categorical variable and create a dense layer for the numerical ones. Then they can be concatenated and passed through the next layer [2].

2.2 Deep Learning

2.2.1 Artificial Neural Networks

Artificial Neural Networks (ANN), usually called neural networks (NN), are computing mechanisms inspired by biological neural networks that form human brains. A neural network is built by a collection of connected nodes called artificial neurons connected among each other and in different layers. Each connection, like synapses, can transmit a signal to another neuron.Artificial neurons are connected by some weights where the signal stands in connection with numerical values. The output of each neuron is calculated by some non-linear functions of the sum of its inputs.

The connections, also called edges with neurons, carry a weight that changes as learning continues. Accordingly, the weight adjusts the signal's power at a connection based on a threshold, and it lets the aggregate signals cross over to the other neuron. Typically, neurons are aggregated into layers, and different layers might execute different transformations on their inputs. As it is visualized in the figure 2.3 the overall structure of an NN includes an input layer, an output layer and several hidden layers between them.

The most basic NN is the feedforward neural network (FFNN), or multilayer perceptron (MLP), where the information is transmitted forward from the input layer through an intermediate layer and to the output layer. For an input vector $\mathbf{x}$, there is a hidden layer $\mathbf{h}^{(1)}$ within a network function $\mathbf{h}^{(1)} = g(\mathbf{w}^{(1)}\mathbf{x} + b)$, where g is some activation function, $\mathbf{w}$ is the weight vector in the first hidden layer and b is bias term. The computations proceed in the subsequent layer until the output has been executed. An intermediate layer can be taken into account as a vector representation of the input $\mathbf{x}$ to an MLP.

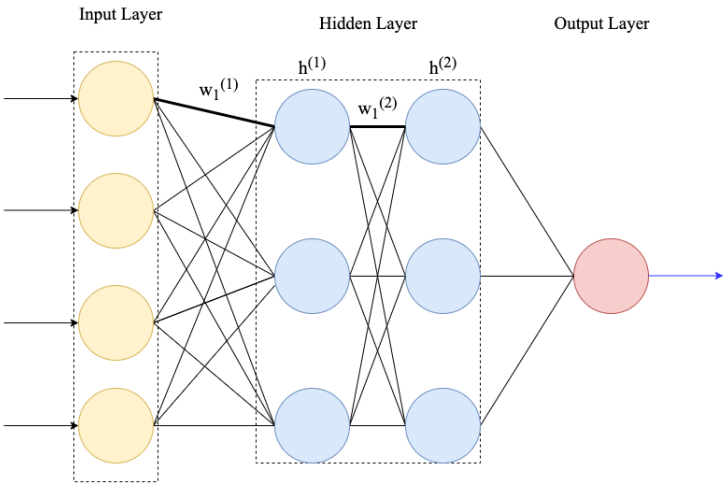

Figure 2.3: The illustration of a multilayer perceptron, a fully connected feed forward artificial neural network architecture [14]. Here, each circular node represents an artificial neuron and each node in one layer connects with a certain weight w_i^y to every node in the following layer.

2.2.1.1 Recurrent Neural Networks

Recurrent Neural Network (RNN) is a network adapted for analyzing sequential data. Dissimilar to a feed-forward network where data is transmitted forward, in RNN, there are feedback connections that can output processing back into the model and are dependent on a chain of inputs. Practically, RNNs operate on a sequence containing vectors as x_t with the time step-index t to τ where the hidden states h_t computed from previous sequences passed forward through time. Mathematically, RNN functionality can be denoted as

$$h_t = tanh(\omega_h h_{t-1} + \omega_x x_t) \tag{2.2}$$

where ω is defined as weights of previous and current states. Compared to a regular artificial neural network, RNN has a memory function that allows remembering the signals from the previous nodes associated with one particular time step. One main downside of RNNs is the vanishing effectiveness of a given input on the hidden layer, as the cycles through recurrent connections cause this information to degenerate over time. This problem called as vanishing or exploding gradient problem, has been practically solved by the design of the Long Short-Term Memory architectures [35]. RNN functionality can be visualized as in figure 2.4.

2.2.1.2 Training Neural Networks

There are two main types of training in machine learning: supervised and unsupervised. Supervised learning exploits labelled data such that every input X has an output y which we call these data ground truth or target data. In contrast, unsupervised learning machines lack information about the outputs as ground truth but understand unseen data structures based on specific tasks, such as clustering.

Specifically, in supervised learning, the aim is to train a network on a specific task where machine learns how to analyse unseen data to classify or assign a label. The machine learning models have been improved by iteratively adjusting the network's

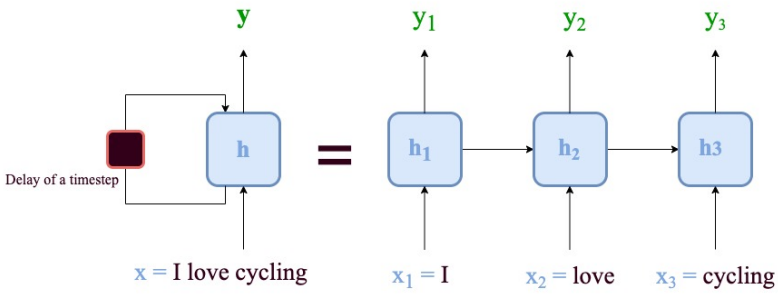

Figure 2.4: Compressed (left) and unfolded (right) basic recurrent neural network

weights until it finds the optimal weights that minimise the loss function or maximise some objective functions based on the seen data. Hence, when training a neural network, the aim is to maximise or minimise some objective functions. A loss function, a type of objective function, measures how well a model prefigures the expected outcome for any data point in the training set, which should be minimised. A common loss function in machine learning is the cross-entropy function, where minimising cross-entropy is associated with maximising the likelihood of the data, in other words, maximum likelihood estimation(MLE), which can be used as an objective function. Moreover, since the weights need to be adjusted based on a layer, backpropagation[32] comes into play. The backpropagation technique computes and distributes the objective functions backwards through the network. The gradient of the objective function is computed with the negative partial derivative based on each weight assigned within the network. Gradient descent is an optimisation algorithm for minimising some loss function where the gradients are calculated by backpropagating the loss concerning its parameters. The learning rate defines the step size to exploit the algorithm for reaching a local minimum. There are also different gradient descent approaches based on the data size that computes the gradient loss function, such as stochastic, batch and mini-batch gradient descent[84]. In batch gradient, the gradient is computed using the whole training set, while stochastic gradient descent updates the parameters for each data point in the training set.

2.2.2 Language Models

Language Modelling is the development of probabilistic models that can make predictions in the sequence of given words according to the context [92]. They are widely utilized in NLP applications in real-time, such as machine translation, speech recognition, information and text summarisation[90]. Language models are built on an algorithm that learns to predict the words of a sentence by determining its probability based on linguistic and statistical features. In a statistical language model for a sequence S having N-words, the probabilities are assigned as shown in the following equations:

$$P(S) = P(w_1 w_2 w_3 w_4 ... w_N) \tag{2.3}$$

$$P(S) = P(w_1)P(w_2|w_1)....P(w_N|w_1 w_2 w_3 ... w_{N-1}) \tag{2.4}$$

Because of the limitations of statistical language models, Neural Network Language Models (NNLM) were introduced. Accordingly, the first model neural network is initialized with an expression defined in the following such that x denotes the feature vector which is formed by the concatenation of input word features expressed in the 2.5 equation

$$x = (C(W_{t-1}), C(W_{t-2}), ...C(W_{t-n+1})) \tag{2.5}$$

Accordingly, y defines the probability of the output word. W, U, and H are weight matrices where b and d are the associated biases of the hidden and output layers.

$$y = b + Wx + Udtanh + H(x) \tag{2.6}$$

2.2.2.1 Neural Network Language Modelling

One of the significant features of using neural language models is employing the vectors as representations of words sequences (i.e. Word2Vec [61]). In such a manner, words with similar meanings are closer to each other in the vector space, which helps neural networks to understand semantic meaning under the sequences. The feed-forward architecture was first used for language models; the following recurrent neural networks came into play, with better results in language modelling tasks. They can take into account the position of words in sentences. Nevertheless, Long Short-Term Memory networks that enable the language models learning the relevant context of longer sequences than feed-forward or RNNs. Thus it had become the dominant approach in the industry so as it achieved the state of the art results [90].

Moreover, the attention mechanism has shown a significant improvement in language models with the sequence-to-sequence framework. Attention mechanism has been proposed to improve the memory mechanism of neural networks by providing the ability to seq2seq to have the sight on the whole context [100]. After exploring the benefits of the attention mechanism in neural language models, the development of transformers was the next major step, which leverages the self-attention mechanism to use the context of the sentence more effectively. These models can analyze the task from both the left and right sides of the sequences, such as BERT [17] or only take into account the left context like GPT [22] model. However, transformer-based language models have their downside: the limited length of proceeded sequences within the mechanism. In other words, the transformers could only consider a limited length of sequences to process and predict, while recurrent networks could work with unlimited sequences. Transformers models like Transformers-XL with long receptive trained on a large amount of data are still capturing large range dependencies as well as reaching the state of the art results.

2.2.3 Seq2Seq Models: Transformers

Sequence-to-sequence modelling, or seq2seq models, have been designed to solve problems such as machine translation, speech recognition, language modelling, Q/A systems, and text summarisation[90]. The initial proposals to combat these problems were based on using RNNs in an encoder-decoder architecture. However, these

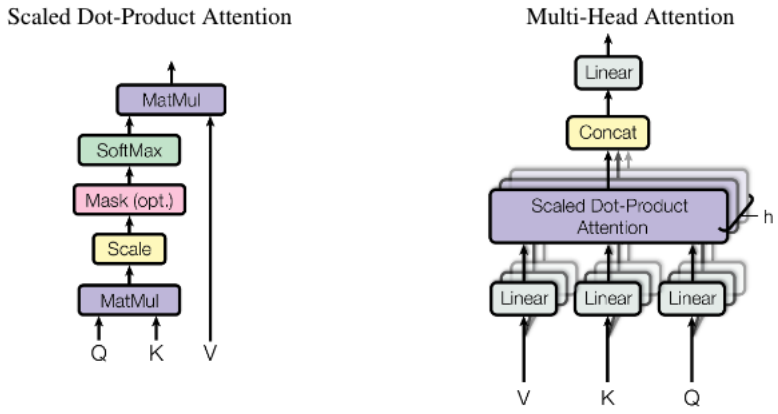

Figure 2.5: Left: Scaled Dot-Production Attention, Right: Multi-Head Attention [95]

architectures have a limitation on long sequences. Since the ability to carry the information from the first elements is missing out when the sequences incorporate other elements, in other words, RNNs (even with bi-directional, multi-layer or memory-based gates) suffer from the vanishing gradient problem. As they all handle a sequence of input one by one, or word-by-word, they all end up with an obstacle towards parallelisation of the process. To handle this limitation, a new concept was introduced called attention mechanism. The attention mechanism is firstly mentioned in the paper [95] by Google to handle long sequences. The role of the attention mechanism is, instead of paying attention to the last state of the encoder as is generally done with RNNs, to extract information from the whole sequence with a weighted sum of all the past encoder states. This allows the decoder to capture global information rather than rely on one hidden state. The learning process of the attention mechanism aims to focus on the right element of the input to predict the following output based on the attention weights. Attention weights are the relevance scores of the input encoder hidden states (values) in processing the decoder state (query); whereby the scores are calculated using encoder hidden states (keys) and the decoder hidden state (query).

Transformers have been a big step forward for sequence learning tasks, also introduced in the paper called "Attention is all you need" [95] by Google in 2017. As indicated in the title, transformers are entirely built on attention mechanisms, which drive the models' focus solely where attention is needed. A transformer learns by executing several steps to find where the attention should be given in the text. Each step applies an attention mechanism to prefigure relationships between all words in a document, regardless of their position. Moreover, transformers process the whole text for each prediction and allow previous states' calculations in a parallelized way, where there is no risk of forgetting important information in a sequence. Nevertheless, this architecture leads to reduced training time and better predictive performances. Similar to other seq2seq models, transformers have been built on encoder and decoder parts. Both parts consist of units and each unit has been built on a multi-head attention mechanism and a feed-forward unit.

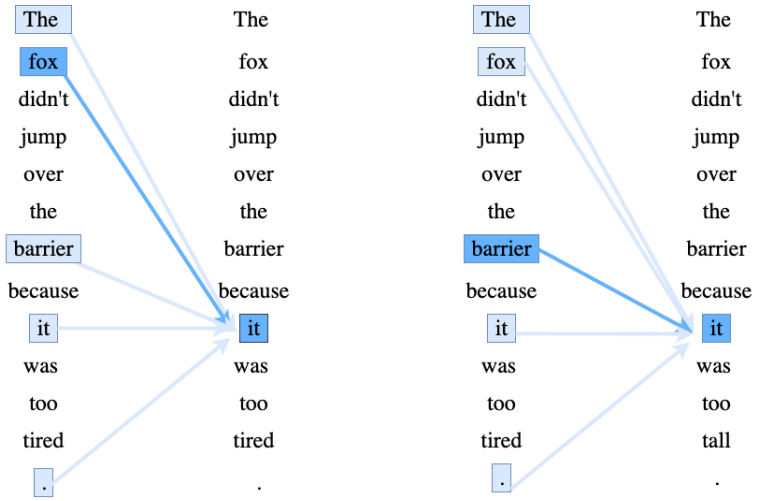

Figure 2.6: The encoder self-attention distribution for the word "it" as in illustrated example

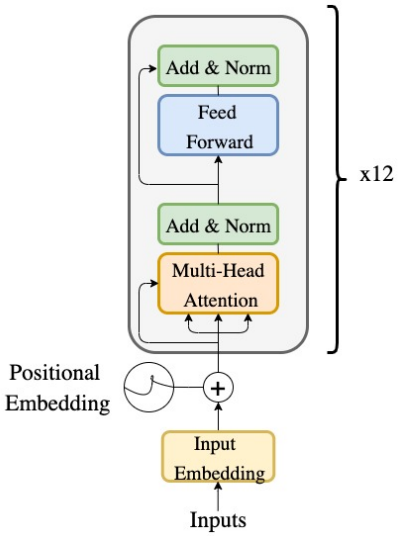

Figure 2.7: Architecture of an encoder in BERT mechanism[17]

2.2.3.1 BERT

Along with the development of seq2seq models, Bidirectional Encoder Representation from Transformers, as known as BERT[17], has become a significant breakthrough that took the Deep Learning community to another perspective by gaining state-of-the-art results on various NLP tasks.

BERT is trained using masked language modelling, where it randomly [*masks*] words in a sentence and tries to predict them based on the context provided by the other, non-masked words in the sequence. Masked language modelling with multi-head attention layers in each encoder unit of BERT enables generating of word embeddings (word vectors) rich in semantics and depend heavily on context. Luckily, BERT is readily available via many different sources (e.g. HuggingFace[1]) and pre-trained on gigabytes of data from various sources (e.g. Wikipedia).

[1] www.huggingface.co/transformers

A primary transformer contains an encoder and decoder that enables reading the text input and generating an output based on a prediction task. In BERT, it only contains an encoder part as its primary goal is to produce a language model and its representation, then pass it to neural networks for specific tasks. Nevertheless, BERT requires some extra metadata with the input: Token embeddings denote where the sentences start and finish, and segment embeddings allow the encoder to differentiate the sentences from each other and positional embeddings denote the position of each token in the sentence.

Model	**MR**	**CR**	**SUBJ**	**MPQA**	**SST**	**TREC**	**MRPC**	**Avg.**
Avg. GloVe Embeddings	77.25	78.3	91.17	87.85	80.18	83	72.87	81.52
Avg. Fast-Test Embeddings	77.96	79.23	91.68	87.81	82.15	83.6	74.49	82.42
Avg. BERT Embeddings	78.68	84.85	94.21	88.23	84.13	91.4	71.13	84.66
BERT [CLS] vector	81.57	86.54	92.5	**90.38**	84.18	88.2	75.77	85.59
InferSent - GloVe	80.09	85.19	93.98	86.7	86.38	**93.2**	70.14	85.1
SBERT-NLI-base	83.64	89.43	94.39	89.86	88.96	89.6	**76**	87.41
SBERT-NLI-large	**84.88**	**90.07**	**94.52**	90.33	**90.66**	87.4	75.94	**87.69**

Table 2.1: Evaluation of SBERT sentence embeddings using the SentEval toolkit [81]. SentEval evaluates sentence embeddings on different sentence classification tasks by training a logistic regression classifier using the sentence embeddings as features.

For many NLP tasks (e.g. semantic similarity comparison, sentence clustering,semantic search) including this thesis, sentence embeddings are useful to automatically extract key information from large bodies of text. The most common BERT-based methods to generate sentences embeddings by simply taking the average of word embeddings of all words within the sequence:

$$[H]s = \frac{1}{|S|} \sum_{w \epsilon S} w \tag{2.7}$$

Mathematically shown in the equation 2.7 it averages the word embeddings w in a sentence to get the sentence embedding.

	NLI
Pooling Strategy	
MEAN	**80.78**
MAX	79.70
CLS	79.80
Concatenation	
(u, v)	66.04
$(\|u - v\|)$	69.78
$(u * v)$	70.54
$(\|uv\|,\ u * v)$	78.37
$(u, v, u * v)$	77.44
$(u, v,\ \|u - v\|)$	**80.78**
$(u, v, \|u - v\|,\ u * v)$	80.44

Table 2.2: SBERT trained on NLI data with the classification objective function[81], For the concatenation methods, the scores are only reported with MEAN pooling strategy (u and v represent two sentence embeddings).

[CLS], a special token that appears at the start of the sentences as input for BERT, can be used to represent the sentence in a semantic space. However, according to the research executed by [81], sentence embeddings generated by the methods such avaraging method or [CLS] representations were not representative enough to outperform GloVe vectors on textual similarity tasks. Moreover, if we take the cosine similarity of the pair of sentence embeddings to pass through BERT, the semantic similarity can be more accurately represented; however, it would be pretty computationally expensive as it scales as $O(n^2)$.

Alternatively, as a pre-training strategy, Siamese fashion came into play [81], where the main idea is to have two sentences together via pooling or concatenation techniques and pass through BERT [17]. Researchers found that a mean pooling strategy worked best (compared to max or CLS aggregation), and among the concatenation strategies $(u, v, |u-v|)$ worked best according to the results of the ablation study on SNLI dataset [7] shown in the table 2.2. Hence, according to the results of BERT with sentence embeddings techniques, shown in the Table 2.1, Siamese BERT or SBERT with mean pooling strategy outperformed compared to the other sentence embedding techniques. For our case, SBERT architecture with mean pooling strategy to combine sentence and transcript embeddings, and classification objective function will be utilized and fine-tuned to extract important sentences in the transcripts of the podcast episodes.

2.3 Speech Features

An earlier work has proposed that prominence is a feature of speech that can attract human attention in a bottom-up manner [44]; in other words, diversification in the physical properties of the speech signal can be indicative of the prominence. Speech analysis can benefit from various features that occur within an audio signal, and the related studies vary based on the used features. The primary division in the research paths in the literature can be split into approaches using linguistic information (e.g. lexical or syntactic structure), acoustic information or both [43]. The second major division can be considered supervised and unsupervised techniques where the difference is in using annotated data to train classifiers.

In linguistics, prosody is defined as the "suprasegmental" characteristics of speech [50]. Informally, it is defined as variation of pitch, duration and loudness of the speech units across time. Prosody is utilised for a variety of purposes in communication, including marking of contrastive emphasis or focus, an indication of the speech act of an utterance, and expression of the speaker's emotions and attitudes [96]. As the acoustic correlates of these features, they as descriptors can be exploited automatically from the speech signal based on the values of the fundamental frequency (F0), duration and signal amplitude. Fundamental frequency concerns the value of the lowest frequency component of a speech waveform, mainly affected by the vibrations of the vocal folds. Duration is the associated length of a speech sound. Loudness is the obtained volume of a speech correlated chiefly with descriptors of signal amplitude, such as energy and intensity. Moreover, attributes related to voice quality, such as creaky, breathy or whispery speech, are also considered prosodic, where the acoustic correlates of these descriptors being as jitter, shimmer and harmonic-to-noise ratio.

Along with prosodic features, spectral features are considered descriptors for speech analysis, calculated from the speech signal spectrum, and contains information about the signal's amplitudes for different frequency rates at a particular point in time. The speech signal spectrum can be acquired by calculating its discrete-time Fourier Transform, separating the signal into its frequency components. Spectral features are also handy for automatic speech recognition as they are characterised by paths of energy peaks and other patterns found in the spectrum. Cepstral features (Fourier transform of the logarithm of a spectrum) are also likely to be useful for speech analysis, where the cepstrum representation can differentiate between components related to the excitation of the signal. Mel frequency cepstral coefficients (MFCCs), another type of cepstral feature, are highly utilised for automatic speech recognition tasks [73]. MFCCs can concisely describe the overall shape of a spectral envelope. They represent phonemes (distinct units of sound) as the vocal tract shape (responsible for sound generation) manifests in them. Most feature vectors were composed of the simple extracted pitch-related, intensity-related, and duration-related attributes, such as the maximum, minimum, median, range, and variability values (functionals). Other feature vectors such as MFCCs have preprocessed attributes which are mathematically transformed.

All in all, it is stated that acoustic features obtained from the analysis of audio can cover many different aspects of the speaker's such as intention, emotion, mood or utterance (e.g. statement, question, command, duration) [83], and the possibility of improvement of speech summarisation has been reported [60]. Specfically, Maskey and Hirschberg [60] used sentence classifiers instead of word classifiers to predict the important the sentences for speech summarisation and compared the predictive power of prosodic, structural, and discourse features; whereby it is stated that a combination of acoustic/prosodic features are enough to build a good summariser when speech transcription is not available.

Moreover, researchers have attempted to utilise prominence information in topic tracking tasks by combining lexical information of words with acoustic features to calculate enhanced term weights [27]. In similar work, Xie et al. [101] investigated the utility of prosodic features for automatic speech summarisation of meeting recordings. Overall, besides their demonstrated effectiveness in topic segmentation, acoustic features have been valuable impact on detecting essential words or sentences in spoken contents. Though there is still less research that manifests the effectiveness of acoustic features in speech summarisation tasks, we aim to investigate these attributes via the Geneva Minimalistic Acoustic Parameter Set[19], as described in the following section.

2.3.1 eGeMAPS

The Geneva Minimalistic Acoustic Parameter Set or GeMAPS is a minimalistic and standardised parameter set of acoustic features curated by Eyben et al.[20] to tackle machine learning problems where the audio vocal signals can be advantageous for the task. These features are based on the time domain (e.g. speech rate), the frequency domain(e.g. pitch), amplitude domain(e.g.loudness) or spectral energy domain(e.g.relative energy in different frequency bands) and due to its minimalistic set of design, the classification tasks and results with these features are pretty

6 voice-related LLDs	**Group**
F0 (Linear and semi-tone)	Prosodic
Formants 1, 2, (freq., bandwidth, ampl.)	Voice Quality
Harmonic difference H1–H2, H1–A3	Voice Quality
log. HNR, Jitter (local), Shimmer (local)	Voice Quality
1 energy-related LLD	**Group**
Sum of auditory spectrum (loudness)	Prosodic
25 spectral LLDs	**Group**
a ratio (50–1000 Hz / 1-5 k Hz)	Spectral
Energy slope (0–500 Hz, 0.5–1.5 k Hz)	Spectral
Hammarberg index	Spectral
MFCC 1–4	Cepstral
Spectral Flux	Spectral

Table 2.3: The 42 low-level descriptors (LLD) provided in the eGeMAPS acoustic feature set[15]

straightforward and less likely to be over-adapted to the training data.

The minimal set contains 18 Low-Level Descriptors(LLDs) interpreting features such as intonation, stress, rhythm, excitation, and various spectral descriptors that exploit speech's base frequency, amplitude and harmonics. Furthermore, with the inclusion of spectral and cepstral parameters have been proved with high success rates in modelling affective states[19], an extension set to the minimalist set with functionals has been proposed to create an "extended" Geneva Minimalistic Acoustic Parameter Set(eGeMAPS). In total, eGeMAPS contains 88 features, including functionals (max, min, mean values). They can be computed from the raw audio waveform by using the openSMILE feature extraction toolkit[20], which is a tool for Speech and Music Interpretation by Large-space Extraction(SMILE).

On account of using the openSMILE toolkit, Alexander et al. [1] have computed each podcast's eGeMAPS functionals and made them ready for further exploitation by other researchers. Fortunately, we will utilise these features as they are already extracted. The features and their functionals (e.g. mean, standard deviation, etc.) of each podcast episode are presented within the Podcast Dataset by implemented over a time window that starts every 0.96s. All the windows overlap both their neighbours by half of their length, which is 0.48s. For our experiment, we consider that these pre-computed features and functionals by averaging over 0.96s of the episode's audio and 0.48s sliding windows of audio signals are synchronised with the textual data.

2.4 Automatic Summarisation

A summary is formulated and defined as a shorter version of an original document that preserves key informational elements and the meaning of content [79]. The summarisation task has captured researchers' attention since the 1950s because manual summarisation consumes much time, effort and cost, leading to tasks impractical [58]. The growing availability of documents has opened a road in the NLP research area for automatic summarisation. However, as stated in the paper by Widyassari et al. [99] that summarisation is a formidable challenge in the field of NLP because it requires precise analysis to produce a good summary. Thus, the researchers still dream of accurate automatic summarisation systems to produce a summary that

1) covers all the main topics in the input text, 2) does not include redundant or repeated data, and 3) is readable and cohesive to the users.

Thereupon, automatic summarisation raises the issues for NLP and AI research community like [105] 1) identification of the most informative segments in the input text to be included in the generated summary, 2) summarisation of single long documents such as books, 3) summarisation of multi documents, 4) evaluation of the computer-generated summary without the need for the human-produced summary to be compared with, and 5) generation of an abstractive summary [59] similar to a human-produced summary.

Hence, many different approaches, techniques, and methods have been created to handle the aforementioned tasks. Mainly, automatic summarisation systems are classified based on three approaches: Abstractive, extractive and hybrid review. The extractive text summarisation approach selects the most important sentences in the input document(s), and the output summary is composed by concatenating the selected/extracted sentences [99].

Abstractive summary is generated by a mechanism that identifies the main concepts in the input documents, then uses Natural Language Processing(NLG) methods to paraphrase the text to express those concepts in fewer words with clear language [99]. Compared to the extractive summaries, generated abstractive summaries are more like the manual summaries created by humans. However, abstractive summaries are very complex and relatively more difficult than extractive summaries since producing abstractive summaries requires extensive natural language processing [80]. Abstractive summarisation approaches mainly include two main steps: 1) building an internal semantic representation [11], and 2) generating a summary using natural language generation techniques to create a summary that is closer to the human-generated summaries [64].When it comes to the hybrid approach, extractive and abstractive summarisation techniques are combined to extract the essential parts of the text and rephrase them with NLG techniques. For our work, we have utilised benefits of extractive summarisation approaches which is explained in the section 3.6.1.

2.4.1 ROUGE

There are multiple ways to asses the accuracy of a model, each stands with their own variations, strength and weaknesses. On the subject of summarisation assessment, traditional evaluation by human judgements with different quality metrics (i.e. coherence, conciseness, grammatically readability and content [59]) is possible to valuate the quality of machine generated summaries. Though even with the simple manual evaluation of summaries with a few linguistic quality questions on a large scale would require extensive hours of human efforts [71]. Hence to have the assessment task in a frequent basis, how to evaluate summaries automatically has drawn attention in the summarisation research community. In this regard Lin [52] carries out a evaluation technique that contains measures to automatically determine the quality of a summary by comparing it to the other (ideal) summaries created by humans. Namely, Lin introduces Recall-Oriented Understudy of Gisting Evaluation (ROUGE) [52] metrics that compute counts of the number of overlapping word pairs, word sequences and n-grams as the granularity of texts. Subsequently,

the metrics in the original paper are defined as follow:

ROUGE-N is an overlap of n-grams where ROUGE-1 stands for how many shared word n-grams that exists both in the generated and reference summaries; likewise ROUGE-2 compares bi-gram matching and so on. ROUGE-L measures longest matching sequence of words using LCS (Longest Common Subsequence) which takes into account sentence level structure similarity naturally and identifies longest co-occurring in sequence n-grams automatically that can be additional advantage as no-predefined n-gram length. Moreover, ROUGE-S calculates the overlaps of any pair of words in sentence order while allowing gaps between words. Complimentarily, these metrics generates ROUGE scores based on performance metrics (i.e. precision, recall and F-measure) which are defined as follows:

$$Precision = \frac{tp}{tp + fp} \tag{2.8}$$

$$Recall = \frac{tp}{tp + fn} \tag{2.9}$$

where

tp = true positive, tn = true negative, fp = false positive, fn = false negative

$$F - measure = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{2.10}$$

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{2.11}$$

Consequently, in the context of ROUGE recall interprets how much of the reference summary is captured by the machine generated summary. Even though recall is a solid measure to understand quality, the generated summary might be too long or noisy so that it can capture all the n-grams in the reference, so then it cannot be counted as a good summary. Besides, precision allows measuring how much of the machine generated summary was in fact relevant in relation to the reference summary. In like manner, F-measure computes an harmonic mean of recall and precision.

$$Recall = \frac{\text{number of overlapping n-grams}}{\text{number of n-grams in reference summary}} \tag{2.12}$$

$$Precision = \frac{\text{number of overlapping n-grams}}{\text{number of n-grams in system summary}} \tag{2.13}$$

Even supposing that ROUGE scores have become a standardized metrics they have been found to achieve high correlation with human judgements in summarisation tasks[52], they have the limited capability for rephrasing, synonyms, topical content and other metrics which can reflect the true quality of summary. In addition that LCS suffers one disadvantages that it only counts the main in-sequence words; hence other alternative LCSes and shorter sequences are not reflected in the final score. For instance, as shown in the following the sentences.

- R: vaccine eradicated the virus
- S1: vaccine eradicates the virus
- S2: the virus eradicates vaccine
- S3: the virus vaccine eradicated

Accordingly, when we only consider the ROUGE-2 score based on R as the reference and the rest as summary sentences, S1 and S2 would have the same ROUGE-2 score because both have one bigram, i.e. "the virus". In the case of ROUGE-L, S1 has a higher score than S2 as ROUGE-L can work reliably at the sentence level. Nevertheless, when we look at S3 and take R as its reference, LCS counts either "the virus" or "vaccine eradicated", but not both; hence S3 has the ROUGE-L score as S2. In parallel with this example, scoring matched paraphrasing correctly might lead to underestimating semantically equal phrases since they do not match on a shallow n-gram level. Though this can be possible through contextualized token embeddings [25]. Accordingly, contextualized embeddings (i.e. BERT [17] and BLEURT [88]) are trained to manage ordering and dependencies.

2.4.2 Automatic Speech Summarisation

Speech summarisation is a technique to identify the most critical subjects within human speech and produce a condensed form suitable for the requirements of a given task. The technique plays a crucial role in enhancing the review of documents, video content retrieval, and automatic meeting summarisation. Moreover, with the demand for online lectures and telecommunication over the years, speech summarisation techniques have become a more attractive field in NLP and AI community. The benefits of speech summarisation vary from improved efficiency and cost reduction in call centres (e.g. by identifying call topics, automatic user satisfaction evaluation) to tracking in project meetings.

Speech summarisation generally contains a series of different technical components. As shown in the figure 2.8, the system initially produces a transcription from audio into text. Summarisation modules, which usually contain sentence segmentation, extraction, and/or compaction submodules, summarise the transcripts. Some summarisers skip the speech signal processing or transcription stage and go directly into the summarisation modules[60].

Similar to text summarisation, there are two types of approaches to speech summarisation: Extractive and Abstractive. Abstractive summarisation aims to produce a fluent and concise summary, paraphrasing the intent, but not necessarily the exact content, of the original speech. Extractive summarisation seeks to highlight and select the important sentence or segments from a speech or document and assemble them to form a main theme. Moreover, extractive speech summarisation allows unique sources that do not exist for text summarisation. For instance, speakers' inherent characteristics of prosody/acoustics can moderate the determination of a spoken document's essential parts and structures. Along with the improvement of deep learning techniques, the seq2seq models with attention mechanisms have become a backbone architecture for solving these kinds of tasks[87]. Recently, in the wake of natural language tasks with pre-trained models, state-of-the-art models have

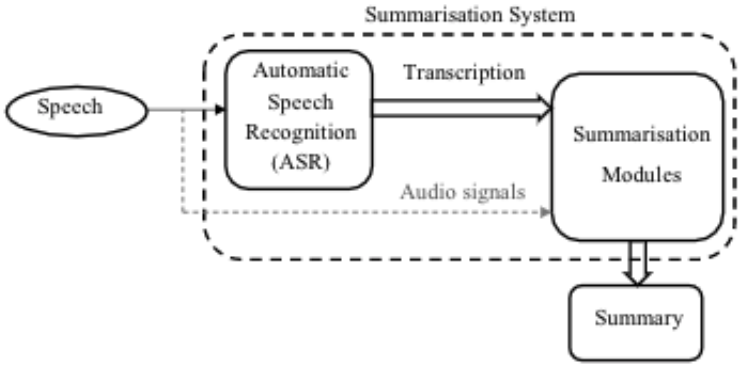

Figure 2.8: General structure of speech summarisation systems[83]

become the de-facto way for speech summary generation[90].

Speech summarisation has been employed in different contexts such as broadcast news, meetings, lectures, TED talks, conversations, and interviews [37]. On that account, broadcast news is mainly recorded under ideal acoustic conditions, and the speech is produced by a professional following a structured script. This allows the summarisation to lower the error rate affected by Automatic Speech Recognition (ASR) sources. Furthermore, for broadcast news summarisation [30] found that lexical, acoustic and linguistic features in combination carry out the best performance. For lecture summarisation, the best indicative features were seen as relevance (semantic similarity between utterances) and discourse (targeting the presence or absence of critical words that refers to an action) features due to their resilience to problems like synonyms and recognition error[83]. Since the podcast summarisation has been just a few pieces of research executed in recent years, to understand speech summarisation practically, we explore meeting summarisation as a different domain that seems to be the most similar to podcasts in terms of conversational, multi-party and multi-stream characteristics in which deep learning methods have also been applied.

2.4.2.1 Meeting summarisation

Meetings are a crucial part of our daily lives, and they allow us to share information and collaborate with others, like podcasts. For efficiency, meetings have also been a hot topic within summarisation tasks, which greatly value both participants and non-participants to grasp the critical notion of the discussions. Along with the development of neural networks, there has been remarkable work for the meetings summarisation task [21]. For instance, to handle long meeting transcripts, Zhaeo et al. [106] propose a technique that allows automatically segmenting meetings into topically relevant parts based on a hierarchical adaptive segmental encoder-decoder network. Recently, Zhong et al. [109] proposed a query-based meeting summarisation task, which allows summarizing the specific part of the meeting according to the indicated query.

Even though deep-learning methods carry out promising results, taking only lexical information into consideration is not sufficient [21]; since meetings include various

types of non-verbal information that the participants present. Hence, to model summarising the meetings in a better representative way, there is a need to incorporate auxiliary information and features [21]. Multi-modal extractive summarisation for meetings has been studied by fusing verbal and non-verbal features to enhance the representation of the utterance [68]. Accordingly, in the papers [101], [103] linguistic features have been identified as the most useful feature category for the summarisation. Nevertheless, a combination of lexical, structural and acoustic features was also found to achieve the highest score of ROUGE in the papers [57], [66]. Despite the studies that do not clearly compare different feature types, there was little consensus on which useful features were most useful for a specific summarisation. However, studies that used more than one feature type typically concluded that the best results could be obtained by combining different features [8], [23], [65]. Thus, as the format of the meetings is similar to podcasts, we presuppose that a combination of heterogeneous features would also improve the quality of podcasts.

2.4.3 Podcast Summarisation

To automatically capture the concepts within the podcasts for the recommendation systems as well as the end-users, the summarisation Task of the TREC 2020 Podcast Track was initialized, and researchers were invited to summarise a podcast episode using its transcript and/or audio. In return, summaries should be grammatically complete and include the most important parts of the podcast episode. Subsequently, all participants in 2020 used to Speech-To-Text generated transcripts and almost none of the participants used the audio data for retrieval (apart from improving the transcription) [1]. Accordingly, TREC and Spotify community released and published several experiments on the podcast dataset with current state-of-the-art summarisation models including BART [51] and T5 [77]. To better understand the SOTA models' performance for the summarisation task, also they had used pre-trained models and fine-tuned them using the CNN/DailyMail [33] dataset to prefigure the performances of the models to be improved.

With the different methods and techniques to enhance the baseline models, there were successful results with different techniques and methods as well as improved ROUGE scores. For instance, [108] had developed a two-phase approach to handle the podcast summarisation task. The approach has two main tasks, first is to select the most important sentences from the transcript via ROUGE-based or topic-enhance approach, which covers the key information of the input, and the second task is to generate the abstractive summary using the selected sentences. [45] proposed an approach that enables capturing the long input sequences by using the seq2seq model with the combination of BART and Longformer. This method was useful to tackle the obstacles of limited input length for transformers and considers much long input sequences and thus outperformed the descriptions made by the podcast creator themselves over the ROUGE scores. Rezapour et al. [82] have built an abstractive summariser over the baseline models by informing the models with categories of the podcasts, which generated better summaries with the higher ROUGE score than the baselines. In the [72] approach, they have built approaches to generate abstractive summaries curated by ensemble models. These ensemble models had two parts where the important sentences are firstly selected (by extracting the first 15 sentences) or detected (by SpanBERT [42]) and passed into a fine-tuned T5 model to produce

abstractive summaries. Their results provided evidence that effective summarisation can be done only by focusing on the beginning parts of the podcast episode.

All in all, none of the summarisation task participants in TREC 2020 used audio data for information retrieval; whereby to encourage the next year participants and promote the use of the raw audio within the domain, Alexander et al. [1] have recently extracted a series of high-level features from the complete dataset and made them readily available for others to use. Specifically, they have used the OpenSMILE toolkit [20] and made eGeMAPS [19] features available with calculated functionals for each of the podcasts, where we plan to exploit these features for our methodology. In such a manner, on top of the baseline models, Alexander et al. [1] have curated three audio metrics computed from eGeMAPS [19] features to carry out an audio-enhances retrieval mechanism. This mechanism elevated the selection of important segments as either defining whether it is entertaining, subjective, or containing a discussion. Thereupon, it is highly likely that audio features can be useful to improve the retrieval of podcast segments and capture the saliency in the transcripts and eventually implement summary generation, whereas they have not been utilized yet to improve podcast summarisation mechanisms. For this purpose, we aim to use these pre-computed acoustic features to improve the quality of podcast summaries and analyse the performance based on ROUGE scores over the baseline models.

Chapter 3

Methodology

The following section will explain how the methods are utilised for our experiment. Respectively, objective of our experiment, podcasts data exploration (section 3.2), pre-processing (section 3.3) will be discussed initially. After that, we explain the metholodogy for feature normalisation (section 3.4) and selection (section 3.5) where we aim to find answers for second and third sub-research questions. In the following, overview of the proposed methods (summarisation with multimodality in the sections 3.6.1, 3.6.2) and the tools we have used will be described.

3.1 Objective

Since human language is predominantly multimodal, which perceives a mixture of natural language, facial gestures and acoustic behaviours, there is great potential and rich information to understand human behaviours and intents [78]. However, the heterogeneity across modalities carries a complication in analysing the human language.

In multimodal language, a linguistic input is accompanied by visual or/and acoustic information; simply put, gestures and prosody co-occur with language. Consider a semantic space that captures latent concepts (positions in the latent area) for individual words and sentences. When there are no multimodal accompaniments, the semantic space is solely focused on the language manifold. In other words, each sentence falls within some part of this semantic space, based only on the meaning in a linguistic structure (i.e. sentence). However, we assume nonverbal behaviours can have an impact on the importance of words, as well as on the position of words in the semantic space. In combination, language and nonverbal complements can determine the new position of the word or sentence in the semantic space. In this section, we introduce our multimodal approach methodology for the podcast summarisation task. Our main focus will be how to combine and represent acoustic and linguistic features as metric to select the salient sentences, which potentially contain prominence in speech of podcast episodes.

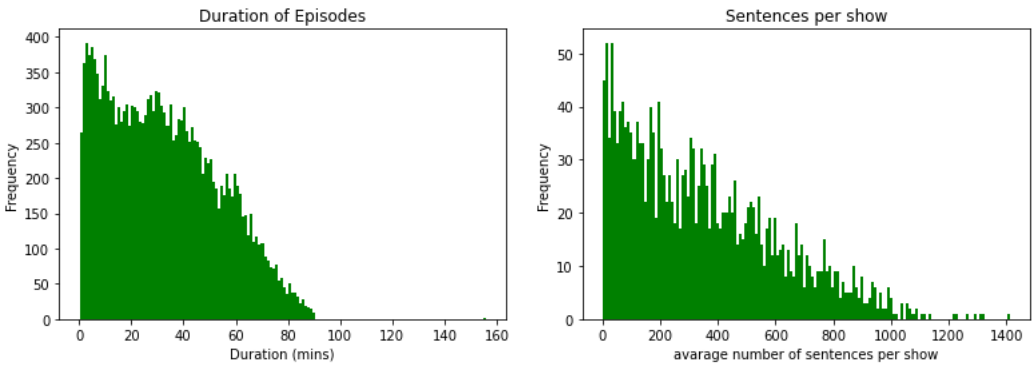

Figure 3.1: Duration of the episodes and sentence frequency in the corpus [13]

3.2 Data Exploration: The Podcast Dataset

For our work, we will be exploiting podcasts episodes from Spotify 100,000 Podcasts Dataset[13] released by Spotify in 2020 to promote research into the emerging field of this spoken audio content, which is the first large scale corpus of podcast audio data that automatically generated transcripts by using Google's Cloud Speech-to-Text API [1], which provides word-level alignments for each word as well as speaker diarisation, casing and punctuation.

Obtained characteristics of the podcast dataset are itemised as follows:

- The corpus with 105,360 podcast episodes published between January 1,2019 and March 2020, covers around 60,000 hours of audio
- The podcast shows are curated by a variety of creators, ranging from professional podcasters with high value to amateurs recording podcasts using an application on their mobile phone.
- The corpus contains variety of topics: lifestyle culture, storytelling, sports, health, documentary and commentary.
- This context multiplicity in the dataset has been carried out by different structural formats, number of speakers and levels of formality, forms of narrative, conversation or debates.
- The episodes are curated from 18,376 podcast shows
- Average duration of episodes is 33.8 minutes and 5,700 transcribed words with large variance (the longest can be over 5 hours and the shortest 10 seconds).
- The creator-provided descriptions are 85 words length in average.
- The dataset contains some related metadata such as name and description of shows and episodes, RSS link, episode duration

Given that multiple speakers are in a podcast, speaker identification and role prediction are relevant problems of interest; in the information retrieval domain, the dataset presents challenges in the noisy nature of the data and the highly varied ways of speaking. According to Clifton et al. [13] from a sample of ASR-generated

[1]https://cloud.google.com/speech-to-text/docs/video-model

transcripts carried out a named entity recognition accuracy of 81.8% and with a word error rate of 18.2%; despite the ratio, they also stated that the transcribed corpus is still valuable to NLP, speech and linguistic communities, as long as noisy nature of data (e.g. static noise, background music, overlapping dialogues) is considered during algorithm development and analysis. Moreover, it is also stated that the automatic diarisation is noisy; on manually checking 20 random episodes, 11 have errors in the number of speakers, and another 4 errors in speaker boundaries [13]. We still take the speaker tags into account and will discuss the impacts of these noises in the section 4.3 after comparing the results of normalisation techniques in speaker, sentence and feature levels.

3.3 Preprocessing

Preprocessing step for this project contains transforming the data into different formats, filtering out the noises and bad data points, synchronisation of heterogeneous features, normalizing the variables while taking the sample to represent the overall dataset (2000 podcasts episodes, 2% of the corpus). To make our dataset ready for further implementation with neural networks, we have performed following filtering steps to reach representative 2000 podcast episodes:

- Keep episodes with descriptions that are not too long (>750 characters) or not too short (<150 characters). We select these thresholds to get more insight while creating oracle summaries from the transcripts and description
- Remove episodes with duplicated descriptions (high lexical overlap over 50%) with other episode descriptions and their show descriptions
- Remove episodes with non-English descriptions
- Keep the episodes in which the speakers have spoken at least 30 sentences (to enable normalising the acoustic features in more speaker-representative way)
- Remove episodes with advertisement-dominated and emoji-based descriptions

The acoustic features that we utilize from audio clips should be within appropriate length to process, defined and split for the further analysis. The final goal of the pre-processing steps is to synchronously split an episode's transcript and audio into synchronized sentences.

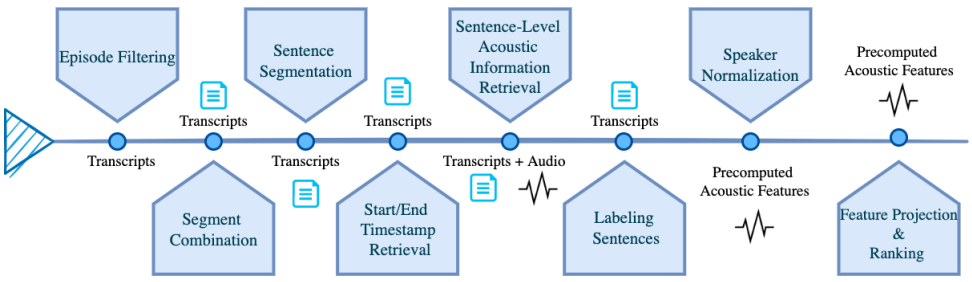

Figure 3.2: Pre-processing pipeline of the podcast dataset

To align the split acoustic features and sentences, we need to execute following pre-processing steps:

- **Segment Combination:**

  All segments in the transcript, which are automatically split by Google Cloud Speech-to-Text API, are merged to obtain a complete and un-segmented transcript of the whole episode.

- **Sentence Segmentation:**

  To split sentences based on where they start and end, we have used SpaCy [36], NLP pipeline toolkit, which contains a dependency parser to detect sentence boundaries, and a pre-trained pipeline to perform an accurate prediction.

- **Start and End Timestamps Retrieval:**

  For each sentence we gathered from the previous step, we will use its first and last words where we have the information of word-level timestamps. Thus, this allows us to identify at what timestamp the sentence begins and ends in the corresponding audio.

- **Word and Sentence-Level Acoustic Information Retrieval:**

  Along with the previous preprocess step, we now know where the audio starts and ends for each word; to match with precomputed acoustic information, we use each word's centroid time (medium value of start and end time). Based on the sliding window values of acoustic features, we allocate the word level acoustic information whereby the centroid time of words overlaps with the corresponding sliding window. Hence, as we obtain the word-level acoustic features, for our model, we compute the sentence-level acoustic features by extracting minimum, maximum and mean acoustic feature values that stand in a sentence. Eventually, we have 264 acoustic features from the calculation above of 88 eGeMAPS Low-level descriptors and their functionals.

- **Labeling Sentences:**

  We will need to have binary labels for our extractive model to execute classification task whether the sentences in the transcripts should be in the summary of the episode. Since we have only episodes' descriptions as reference summaries (which can be under representative for the model to detect meaningful sentences), we will implement oracle summaries of the podcast episodes as an alternative. In other words, we will be assigning the sentences of a podcast episode by ranking their ROUGE scores in accordance with the description of the episode. Subsequently, sentences will be set to 1 if the sentence stands in the 90th percentile; simply put, the sentence with a ROUGE score better than 90 percent of the sentences in the episode will be included in the oracle summary, otherwise the label will be assigned as 0 and not be included in the summary.

Show	Millenials with Money						
Episode Description	the world caught us red handed. yep, our plot to conspire and decimate the economy has been foiled by brilliant, yet cranky gen-exers and baby boomers who weep at the thought of applebees going out of business. in this episode we will discuss why we can't help but systematically kill well-established goods and services.						
Segment-level information of the first segment	**StartTime**	**EndTime**	**Segment**				
	0	9.9	You're listening to Millennials with money a podcast about the intricacies of an entitled generation. I'm Jose and I'm Charlie you're on official source of hot tags				
Word-level information of the first segment	**StartTime**	**EndTime**	**CentroidTime**	**word**	**SpeakerTag**	**loudness_sma3_stddevRisingSlope**	**F0semitoneFrom27.5Hz_sma3nz_stddevNorm**
	0	0.3	0.15	You're	1	3.7937	0.2418
	0.5	0.8	0.65	listening	1	3.7937	0.2145
	0.8	1	0.9	to	1	3.3060	0.2181
	1	1.4	1.2	Millenials	1	3.4898	0.2424
	1.4	1.7	1.55	with	1	3.7422	0.2501
	1.7	1.8	1.75	money	1	3.6379	0.1907
	1.8	2	1.9	a	1	3.1075	0.2233
	2	2.2	2.1	podcast	1	4.1923	0.2240
	2.2	2.5	2.35	about	1	4.8347	0.2554
	2.6	2.8	2.7	the	1	5.0888	0.2181
	2.8	3.1	2.95	intricacies	1	4.7279	0.2201
	3.1	3.6	3.35	of	1	3.3060	0.2397
	3.6	3.7	3.65	an	1	4.7290	0.2746
	3.7	3.8	3.75	entitled	1	6.2506	0.2170
	3.8	4.4	4.1	generation.	1	5.9821	0.2325
	4.4	4.7	4.55	I'm	1	5.9821	0.2408
	4.7	5.4	5.05	Jose	1	16.5290	0.2918
	5.4	6.6	6	and	2	6.7043	0.2065
	6.6	7.1	6.85	I'm	2	8.8505	0.2706
	7.2	7.5	7.35	Charlie	2	13.9456	0.2706
	7.5	7.9	7.7	you're	2	10.8788	0.2433
	7.9	8.3	8.1	offical	2	7.8995	0.2141
	8.3	8.5	8.4	source	2	9.9365	0.3003
	8.5	9	8.75	of	2	7.7292	0.2636
	9	9.4	9.2	hot	2	4.4512	0.1642
	9.4	9.9	9.65	tag.	2	6.3565	0.2568

Figure 3.3: A snippet of an example transcript with related metadata

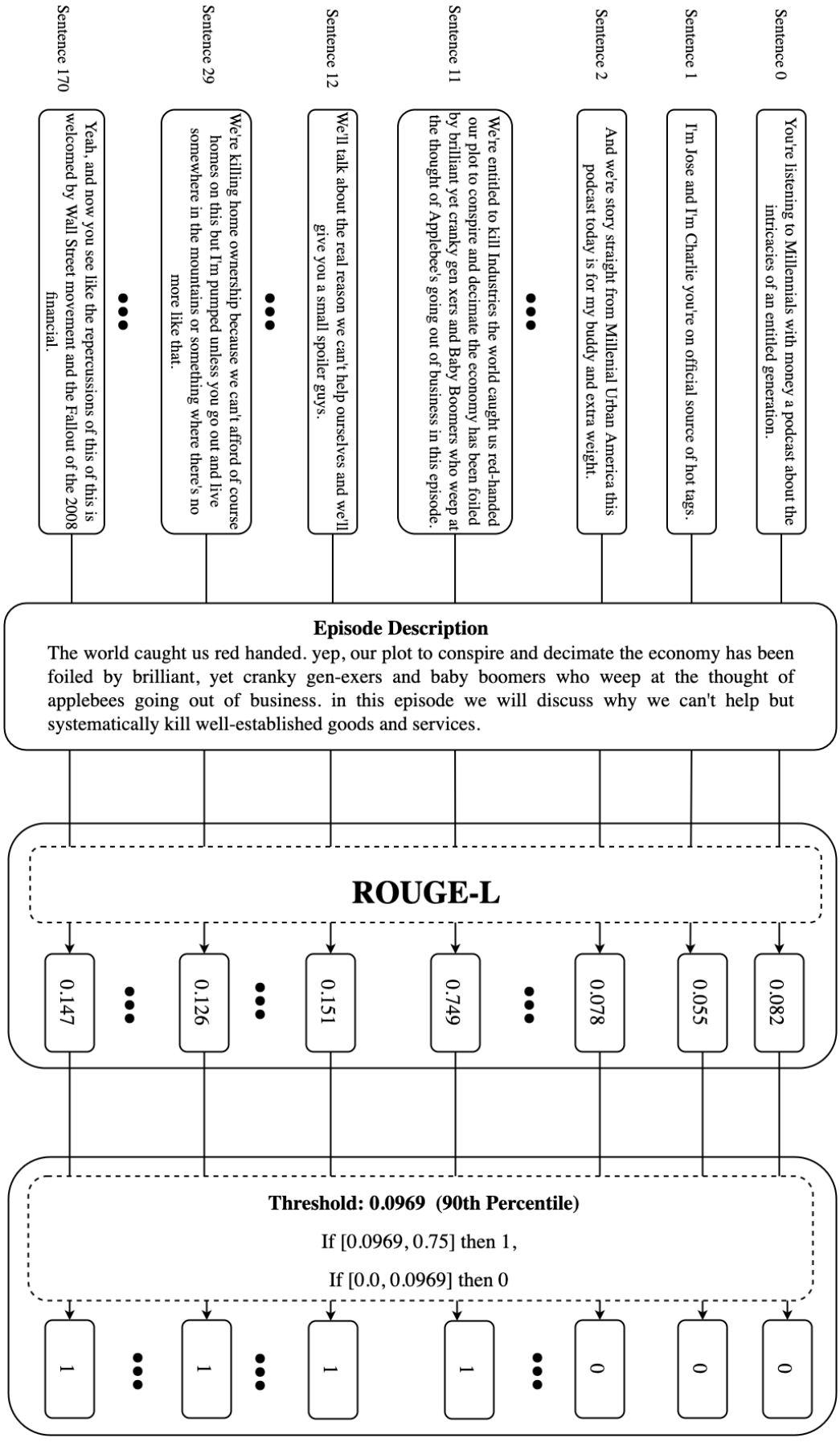

Figure 3.4: Illustrated process of creating an oracle summary (labeling sentences) from the episode description of "Millenials with Money" podcast show

3.4 Feature Normalisation

Normalisation in machine learning is the process of translating data into a range (e.g. [0,1] or any other range), in other words, transforming data onto a unit sphere. This enables machines to identify, aggregate or remove the redundant data; simply put, it is the process of developing clean data to make it appear in all fields and values. In deep learning, having features in different ranges of values might lead gradients to oscillate back and forth; it might take longer to find its way to the global/local minimum. Hence we need to maintain the different values on similar ranges of values so that gradient descent can converge within a shorter time. It is also worth mentioning that there is a need for speaker-normalisation for speech summarisation for podcasts like any other dialogue content. Since non-lexical information carried by the speech signal contains undesirable variability characterised by speakers (e.g. identity, gender, and age), recording conditions or linguistic content [46]. Reducing these variances to physical differences is necessary to exploit the remaining potentially significant features and understand the variation between speakers. Hence, we needed to find a computationally efficient speaker normalisation technique to minimise the effect of these factors. In the following, we explain and apply different implementation-friendly normalisation techniques in speaker, feature and sentence levels to understand the impact of normalisation on the acoustic information where we plan to answer our second sub-research question *"In which way to normalise acoustic features effectively so that speaker-neutral representations can be utilised for podcast summarisation models?"*. Accordingly, we needed to find a computationally efficient speaker normalisation technique to minimise the effect of these factors. Inspired by a paper curated by Kaya et al.[46], we take into consideration linear and non-linear speaker normalisation techniques to reach the level of speaker-independent speech summarisation whereby speech normalisation are based on speaker, feature and instance levels.

$$x' = \frac{x - min(x)}{max(x) - min(z)} \tag{3.1}$$

$$x' = \frac{x - \mu}{\sigma} \tag{3.2}$$

where x is the original feature vector, μ is the mean of the feature , and σ is the standard deviation of the feature vector

$$x' = \frac{x}{||x||} \tag{3.3}$$

means dividing each component by the Euclidean length $||x||$ of the vector.

We apply four different techniques while considering Z-normalisation over corpus as a baseline, formulated in the equation 3.2 and the rest of three techniques are based on Z-normalisation per speaker; the feature values are calculated and applied separately to the data of each speaker. Additionally, we then applied feature level normalisation with linear processing technique (e.g. Min-Max Normalisation, equation 3.1) to each feature. Concerning the instance-level normalisation, each feature vector x is separately L_2 normalised, as formulated in equation 3.3. In instance-level

normalisation, the L_2 normalisation computes the distance of the vector coordinate from the origin of the vector space. It is also known as the Euclidean norm as it is calculated as the Euclidean distance from the origin. Consequently, we calculate the Fisher Scores to understand speaker-level normalisation techniques' performances regarding the empowering discriminability of candidate summary sentences with the acoustic feature values. Eventually, after comparing Fisher's average scores of features normalisation techniques, we select the most discriminative normalisation technique for our acoustic features.

3.5 Feature Selection

Selection of features is a technique for data modelling that has been shown to improve models' supervised and unsupervised learning performance and successfully applied in many real applications, such as pattern recognition [39], text categorisation [40], and image processing [85] and so on. Specifically, when there is noisy data, multi-type features, or lots of frequent features to compare the samples in models, there is a need for feature selection to reduce computational costs in training time or required resources. Moreover, having a large number of dimensions drives the volume of feature space very large, and in turn, the points that we have in the space (sentences in our case) might lead to minor or non-representative samples [54]. This is generally referred to as the curse of dimensionality [31].

The dimensionality reduction techniques aim to mitigate the curse of dimensionality and lessen the number of dimensions of the feature space. Specifically, feature projection [75] and feature selection [53] play a critical role in handling the curse resulting from too many dimensions. Feature projection is to use methods from the field of linear algebra, and the algorithms are used referred to as "projection methods". These methods transform the data from high dimensional space to low dimensional space while preserving the "essence" of the data. Feature selection assists in finding a subset of original data so that there will be minimum loss of information. Supervised feature selection algorithms[98] stand for identifying relevant features for achieving the goal of the supervised model (e.g. classification, regression problem), and they depend on the availability of labelled data. Nevertheless, unsupervised feature selection methods [18] rely on unlabelled data. They have been developed for scoring all data dimensions based on criteria, such as variance, entropy or their capability to maintain the local similarity.

One of the standard projection methods for linear transformation is Principal Component Analysis (PCA), invented by Pearson in 1901 [5] and used for exploratory data analysis (EDA) and predictive models. PCA assists in finding the feature components that maximise the variance, which empowers the separability of the categories; whereas Linear Discriminant Analysis (LDA), a common technique for dimensionality reduction [4], uses features to create new axes and tries to project the data onto new axes, so that maximises the separation of the categories. LDA uses target values to find the new axes as a supervised learning algorithm for dimensionality reduction tasks, while PCA ignores the category labels as an unsupervised technique.

There are three categories for feature selection methods based on how they inter-

act with the classifier: filter [55], wrapper [49], and embedded [16] methods. Filter methods mainly rank the features from best to worst, relying on the feature selection algorithm. The rankings are based on the intrinsic properties of the data (e.g. variance, consistency, distance, correlation). The algorithms have different criteria to measure the relevance of the data or how much the feature influences the class labels. For instance, the Fisher score[26] is a common filter method that utilises mean and variance to rank the features and evaluates components individually; mathematically, the Fisher ratio(FiR) for feature selection is described as the distance between sample means for each class per feature divided by their variance. According to the Fisher score, a good feature should drive instances from different classes far away and make instances from the same class close to each other; in other words, the larger the Fisher score, the better the selected feature is.

Wrapper methods use machine learning algorithms as a black box evaluator to figure out the best subset of features where they are dependent on the classifier [49]. Embedded methods connect filters and wrappers; in other words, they fuse measurable and statistical criteria such as a filter to choose some features and then use a machine-learning algorithm to find the best subset with the best classification performance [16]. They reduce wrappers' computational complexity without re-classifying the subsets in each iteration and can model feature dependencies; in other words, they perform feature selection and train the algorithm in parallel. One disadvantage is their dependency on the classifiers. Decision Tree [76], RandomForest, and XGBoost[10] are standard embedded methods used for feature selection and getting the feature importance. They stand for improving the variance of simple decision trees by incorporating randomness. All in all, these methods have similar purposes for leading to better models which achieve higher performance and/or enable machine learning to train faster and more interpretable. On the subject of answering our third sub-research question, we have applied aforementioned feature projection and selection methods to obtain the most representative way for integrating acoustic information into our neural network and presented and discussed the results in the chapter 4.

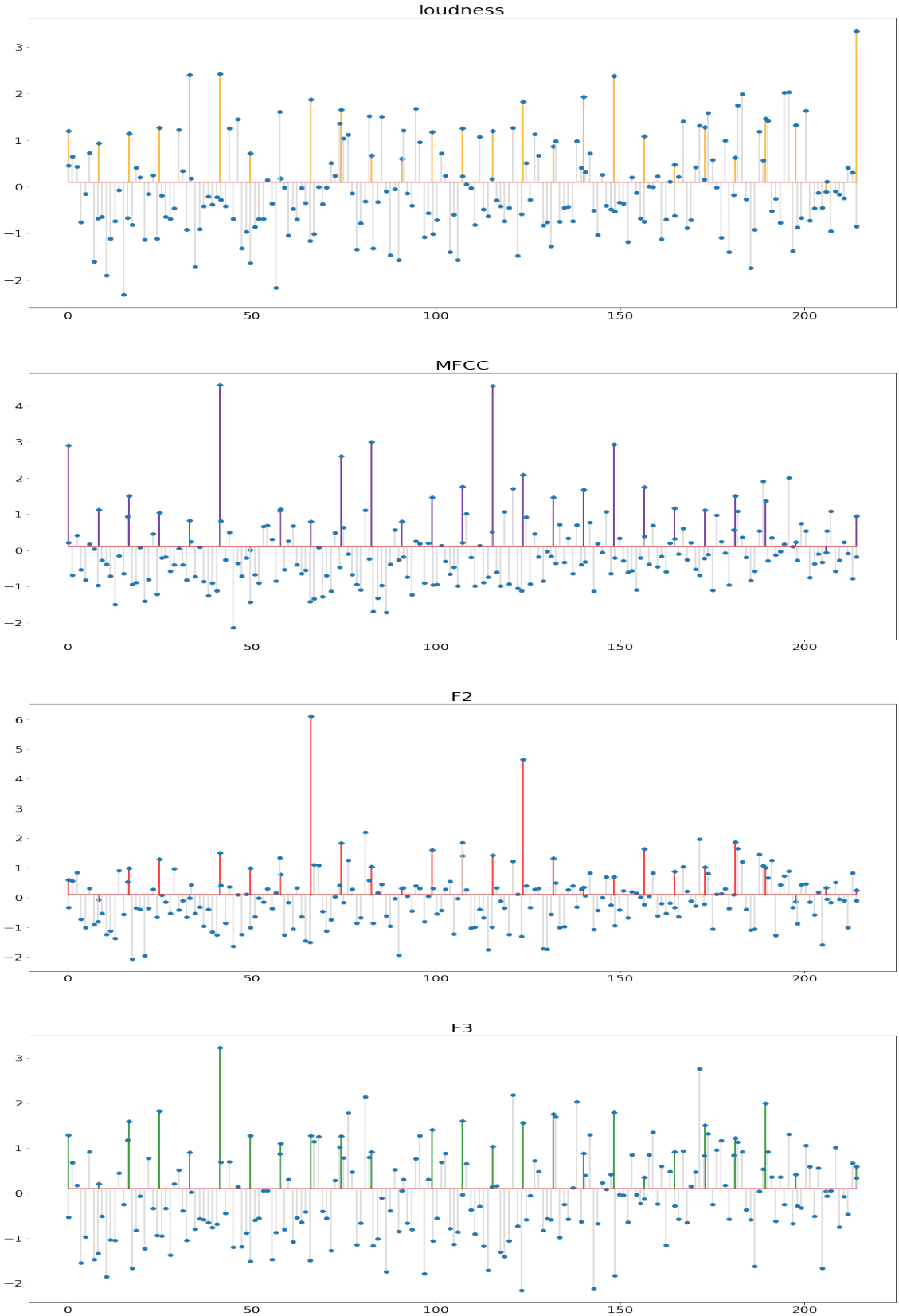

Figure 3.5: Illustration of sentence-level values of acoustic features (i.e. LDA representations of loudness, MFCC, F2, F3) for "Millenials for Money" podcast episode; colored diagonal values represents the sentences that are considerable candidate sentences for the summary (labelled as 1,otherwise 0), grey round values demonstrates the sentences that are not representative enough for including in the summary

3.6 Multimodal Summarisation

3.6.1 Summarisation Method

In this thesis, the problem of podcast summarisation will be addressed as automatic scoring importance of sentences and extracting them from the transcribed text from podcast episodes. In this framework, we aim to create a speech summarisation system using extractive approaches that evaluate textual and acoustic features; hence literature-wise, we will be focusing on the extractive approaches. An extractive summary consists entirely of extracted essence, where the output from the extractive summarisation system is just like the segments obtained from the original document[47]. Length of summaries generated by extractive models relies on the selected compression rate (percentage of whole document) or threshold to limit the length of the summary while keeping the same order of the generated sentences as the input text. It is stated which we found practical for our case that the extractive approach is a primary method, which is faster and more straightforward than the abstractive approach [99]. Moreover, it is helpful and valuable for different applications such as creating a trailer [110] for a movie with linguistic features or selecting essential documents based on the sentence/document scores.

Extractive summarisation approaches contain three main steps which we utilise in our experiment as well:

1. Creating a suitable representation of the input text to facilitate the text analysis (e.g. N-gram, bag-of-words, embeddings etc.) [42]
2. Scoring of sentences: ranking sentences based on the input text representation [99]
3. Extraction of high scored sentences from the input document(s) and concatenating them to create the summary [67]

Along with using the benefits of extractive approaches, to answer our first sub question," *Which combination technique is to utilise for multimodal analysis of heterogeneous information in neural summarisation models that can assist in detecting salient parts of the speech in mono/multi-agent dialogues?*" we needed to find a way to incorporate acoustic features linked with the respective sentences for generating extractive summaries. Touching on, in the following we explain how we build a multimodal extractive summarisation model for our experiment.

3.6.2 Multimodality

Since human language is predominantly multimodal, which perceives a mixture of natural language, facial gestures and acoustic behaviours, there is great potential and rich information to understand human behaviours and intents [34]. However, the heterogeneity across modalities carries a complication in analysing the human language.

Visual or/and acoustic signals supplement linguistic information in multimodal language; broadly speaking, motions and prosody co-occur with language. Suppose a semantic space for individual words and sentences that captures latent concepts. When no multimodal signals are available, the semantic space is focused entirely on

the language manifold. Relying on the meaning in a linguistic structure (i.e. sentence), each word or sentence falls within some portion of this semantic space. We think, however, that nonverbal behaviours can influence the significance of words/sentences and their placement in the semantic space. Language and nonverbal support, when combined, can determine the new position of the word or sentence vectors in the semantic space that enables the better presentation of the information and possibly improves the performance in various tasks (i.e. emotion recognition, summarisation). Illustrated in the figure 3.5, the selected sentence-level acoustic features with per-sentence values of an episodes are shown with label information: the colored diagonal values represents the sentences that are labelled as 1 in the preprocess step, while generating the oracle summaries. Accordingly, it is noticable that some sentences expose itself with higher values in terms of acoustics, which might help representing sentences with multimodality in a semantic space.

Transformer-based models are a tipping point when using unstructured text data and in the real world, text data is often empowered by unstructured data (e.g. audio, visual information), such as expressing the sentiments and stronger signals about the statement. We call these different ways of experiencing data audio and text modalities. This project explores how to exploit text and acoustic data to provide stronger signals for detecting salience in the podcast. Accordingly, we started to investigate the "multimodal learning" field, exploring how to process different multimodalities in machine learning. Within multimodal learning, there are various branches of research; our problem in this thesis will be subjected to what is known as Multimodal Fusion, incorporating information from two or more modalities to make a classification or regression[69]. As we take the text data into account as the primary modality, our research will be focused on the literature that treats text as the main modality and introduces models that utilise the transformer architecture. The main key factor for adapting transformers for multimodal data is ensuring a weighting mechanism between the different modalities. These weighting or attention mechanisms can be integrated at different points of the transformer architecture. It can be encoded as input embeddings, injected in the middle or combined after the transformer encodes the text data.

There are multimodal models integrated with transformers for audio and visual modalities with temporal alignment with text modality. Tsai et al.[93] have worked on multimodal transformers for unaligned multimodal language sequences and created a model that serves as a strong baseline capable of capturing long-range contingencies regardless of the alignment assumption. The paper [78] by Rahman et al. worked on the language-only position with a displacement vector in the light of acoustic information. The proposed technique for multimodality of a language is based on implementing the Multimodal Adaption Gate(MAG) [78], which takes a lexical input vector and its visual and/or acoustic accompaniments. Eventually, attention over lexical and nonverbal dimensions is used to fuse the multimodal data into another vector, subsequently added to the input lexical vector (shifting). In other respects, the knowledge graphs are identified as the crucial key of information in addition to text data; specifically, [70] utilise features from author entities in the Wikidata knowledge graph with metadata features for book category classification. The proposed model has a straightforward and practical approach: the book title's author and metadata features and BERT output text features are simply

concatenated (via MLP) before the final classification layers. For our case, we were influenced by the approach "Enriching BERT with Knowledge Graph Embeddings for Document Classification"[70] regarding the practicality of the problem solution and it allows applying such classification tasks according to a generic ontology, which enables implementing for our project as well. They reported that they achieved considerably better results for the classification task than the standard BERT approach. Thus, we will be relying on the same approach in building the multimodal summarisation system for combining acoustic and linguistic features with SentenceBERT and MLP (see Fig. 3.6 for the illustration).

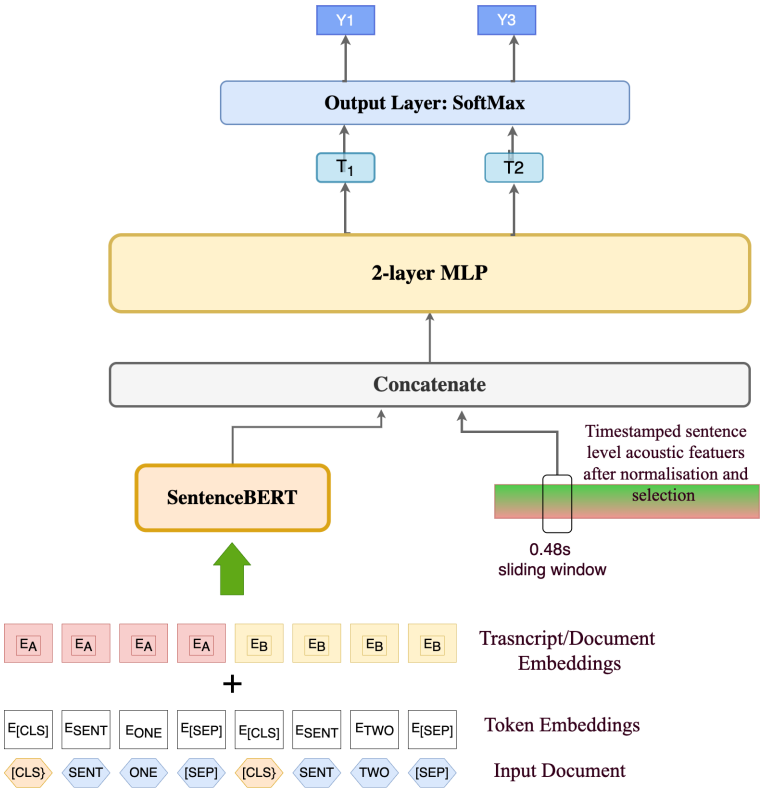

Figure 3.6: Architecture of the proposed model

3.6.3 Model Architecture

As aforementioned in the previous, on account of its practicality, we were inspired by the multimodal transformer model proposed by [70]. Our custom multimodal model's architecture has a similar structure which resembles the original BERT model[17] and combines text and acoustic features with a two-layer MLP. Nevertheless, to analyse the transcripts with acoustics features at the sentence level, there is a need for deriving semantically meaningful sentence embeddings. Hence, we have selected a pre-trained sentence transformer model, readily available in HuggingFace called "paraphrase-MiniLM-L3-v2"[2], which is a modification of the pre-trained BERT network that uses siamese and triplet network structures. The selected sentence transformer MiniLM [97] relies on the self-attention relation distillation for task-agnostic compression of pre-trained BERT, where it speeds up the training

[2]https://huggingface.co/sentence-transformers

time frame 2.7 times compared to the BERT-base model. Similar to BERT-base architecture, it has 12 layers; each layer consists of 384 hidden dimensions, as details of the model shown in the table 3.2.

To obtain the contextualised representations from textual features, the sentences of transcripts are fed through the model, where the input limit is 512 tokens. The transcript (document) embeddings are also represented in a 384-dimensional vector. In the next step, sentence, document and dot products of sentence and document embeddings are concatenated, and the mean pooling technique is applied for the combination. After the last concatenation with additional non-textual (acoustic) features, the sentence representation is passed into an MLP with two layers, 768 units each with a ReLu activation function. During training, the MLP stands for learning the non-linear combination of its input representations. Finally, the output layer does the actual classification. The SoftMax output layer executes the classification task on whether the sentence should be included in the summary of the podcast episode. When the value of an output unit is above a given threshold (0.5 in our case), the corresponding label is predicted. On that account, we are able to answer our first sub-question which is about finding a method for combining acoustic and linguistic features which can assist in finding the important parts of the speech, we will demonstrate the proposed model based on the results in the section 4.

	Train	**Validation**	**Test**
Sentence ratio in corpus %	72	18	10
Number of episodes	1437	360	200
Number of sentences	144249	3602	21188
Number of speakers	1878	412	271
Average number of speakers per transcript	1.30	1.39	1.35
Average number of sentences per transcript	100	101	107
Average number of sentences per speaker	76	75	78

Table 3.1: Train-Validation-Test splits from 2000 episode samples from Spotify 100.000 Podcast Dataset

3.6.3.1 Experimental Setup

As shown in the Table 3.2, training is performed with dropout probability d = 0.1, learning rate ρ = 2e02 (Adam optimiser) and 5 training epochs with the 5-Fold validation technique. These hyperparameters are proposed by Devlin et al. [17] for BERT fine-tuning. All experiments are run on Google Colab that has enabled us to access NVIDIA Tesla V100-SXM2 [3] GPU for using the Google Colab environment. A single training epoch takes up to 90 min on average for 144249 number of sentences.

[3]https://www.nvidia.com/en-us/data-center/v100/

Parameter	Value
MLP Layers	2
SentenceBERT Layers	12
Training Batch Size	16
Epoch	5
Learning Rate	2e-02
Max Input Token	512
Hidden Units	384
Sentence generation limit	3
Dropout Ratio	0.1

Table 3.2: An overview of hyperparameters used

To compare against a relatively simple baseline, we explored the summarisation results with first 3 sentences, as the introduction have potentially indicate what the podcast is about. Moreover, our primary baseline model will be the same proposed model with only text features, whereby we aim to explore the impact of acoustic features in summarisation models.

3.6.3.2 Tools

We have used *Python* as the programming language to use throughout this project. The first season of our choice is because of the wide availability of neural network processing packages, while the second reason is that we are relatively more skilled and more confident in Python compared to other programming languages. Also, we are using Google Colab, which is under the Google distribution, as the interactive IDE for its ease of use during the development of the project.

We chose and utilized several lingusitic processing libraries to complete the first step of the project:

1. PyTorch[4]

   PyTorch is an open source machine learning framework based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Meta AI.

2. HuggingFace[5]

   Hugging Face is a startup in the Natural Language Processing (NLP) domain, offering its library of models for use by some of the A-listers including Apple and Bing. Hugging Face has been able to swiftly develop language processing expertise. The company's aim is to advance NLP and democratize it for use by everyone.

3. NumPy[6]

   NumPy is one of the foundational packages of Python for scientific computing. It provides several useful objects for computation such as multi-dimensional

[4]https://pytorch.org/
[5]https://huggingface.co/
[6]https://numpy.org/

arrays and matrix objects, and manipulation of such objects, including linear algebra operations. This package is used by some of the other numerical and lexical data to handle and manipulate as they include data points in n-dimensional arrays.

4. Pandas[7]

   Pandas is a fast and efficient 2D array and data frame library for data manipulation with integrated indexing. It has several tools which allow to read and write between in-memory data structures and other data formats such as CSV and H5. Pandas has an intelligent and label-based slicing, indexing and subsetting functionalities for large data sets. It also provides high performance in merging and joining two data sets.

5. MatPlotLib[8]

   Matplotlib is a comprehensive library for creating static, animated, and interactive visualisations in Python. We use this library to plot the statistics, distribution, and calculated features of the shape data with simple histograms, line graphs and multi-plot displayed other visualisations. This library is used in almost every step to visualize the results of any process.

6. Scikit-learn[9]

   Scikit-learn is an open software machine learning library for the Python programming language.It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means, and is designed to interoperate with the Python numerical and scientific libraries $NumPy$ and $SciPy$. This library is used for several purposes, such as standardisation of features and distances in variables.

7. spaCy[10]

   spaCy is an open-source software library for advanced natural language processing, written in the programming languages Python and Cython and it contains small English pipelines trained on written web text (blogs, news, comments), that includes vocabulary, syntax and entities to analyse and process the textual data.

[7]https://pandas.pydata.org/
[8]https://matplotlib.org/
[9]https://scikit-learn.org/stable/
[10]https://spacy.io/

Chapter 4

Findings

Overview

The final chapter includes our findings after the experiment, in the section 4.1 we explain the method for selection of features for our task. After that in the section 4.2, we show our training results of our proposed multimodal model with the selected acoustic features in comparison with the Text-Only model. Finally, we discuss our findings and suggest possible future work in the section 4.3.

4.1 Feature Importance: Projection & Ranking

As we discussed the necessity of dimensionality reduction in the section 3.5, we need to obtain the most representative way for fusing the acoustic information into our proposed neural network. For this purpose, we execute LDA for feature projection and obtain a single representative feature for each type of acoustic features (e.g. F0, MFCC) from their related features (e.g.F0semitoneFrom27.RisingSlope). Eventually, we had 15 LDA projections of acoustic features as in Table 4.1. This process aims to expose the discriminability of acoustic features to interpret the importance per feature type. Subsequently, we ranked these LDAs by their respective Fisher scores, which explains the larger the Fisher score, the better the selected feature is in a range of [0,1]. Also we ranked all features by Fisher score (see Appendix A) to compute and understand the effectiveness of classifiers for our supervised task. By looking at the Fisher score comparison of normalisation techniques in the table 4.1, we can have the answer for our second sub-research question *"In which way to normalise acoustic features effectively so that speaker-neutral representations can be utilised for podcast summarisation models?"* that speaker Z-normalisation with MinMax-normalisation on the corpus has given the best Fisher scores. Moreover, Z-normalisation per speaker (without any additional normalisation) has given the higher Fisher scores than L_2 normalisation method; hence we can observe that implementation of instance level normalisation (L_2) has lowered the Fisher scores. Nevertheless, when we compare the Z-normalisation on speaker and overall corpus (as a baseline normalisation technique), we can demonstrate that normalising every speaker in the dataset by taking the mean and standard deviation of all respective speaker values enables us to explore more discriminability.

ZNorm on corpus			**ZNorm per Speaker**			**ZNorm per Speaker + MinMax Norm**			**ZNorm per Speaker + L_2 Norm**		
Rank	**Feature**	**Fisher Score**	**Rank**	**Feature**	**Fisher Score**	**Rank**	**Feature**	**Fisher Score**	**Rank**	**Feature**	**Fisher Score**
1	loudness	0.270	1	mfcc	0.287	1	mfcc	0.310	1	loudness	0.249
2	F2	0.270	2	Segments	0.278	2	loudness	0.303	2	F2	0.243
3	F3	0.268	3	loudness	0.276	3	F2	0.301	3	F0semitone	0.240
4	Segments	0.251	4	F3	0.275	4	F3	0.301	4	F3	0.235
5	F1	0.243	5	F0semitone	0.269	5	Segments	0.288	5	Segments	0.228
6	F0semitone	0.242	6	F2	0.269	6	F1	0.281	6	mfcc	0.225
7	mfcc	0.238	7	F1	0.256	7	F0semitone	0.269	7	F1	0.216
8	slopeV	0.214	8	slopeV	0.242	8	slopeV	0.267	8	spectral_Flux	0.197
9	jitter	0.203	9	spectral_Flux	0.217	9	spectral_Flux	0.236	9	slopeV	0.191
10	spectral_Flux	0.201	10	hammarberg	0.231	10	hammarberg	0.209	10	shimmer	0.187
11	shimmer	0.198	11	alphaRatio	0.205	11	jitter	0.225	11	jitter	0.185
12	hammarberg	0.174	12	shimmer	0.203	12	alphaRatio	0.220	12	hammarberg	0.157
13	alphaRatio	0.158	13	$H1_A3$	0.202	13	shimmer	0.219	13	alphaRatio	0.139
14	$H1_A3$	0.152	14	jitter	0.200	14	$H1_A3$	0.210	14	$H1_A3$	0.116
15	HNR	0.139	15	HNR	0.159	15	HNR	0.169	15	HNR	0.110

Table 4.1: Fisher scores of LDA projection of acoustic features after applying different normalisation techniques

Looking at the table 4.1 and the ranks of LDAs as classifiers, the LDA projection of MFCC acoustic leads to the highest score, followed by F2, F3, loudness, and so on. When it comes to Fisher scores of all acoustic features without projection (see appendix A), we can observe that MFCC, loudness and F0semitone and related features are still leading with the highest Fisher scores; nevertheless, based on the score comparison, LDA projection has still better credibility when it comes discriminability. To strengthen our demonstration from the results, we have also execute the XGBoost algorithm to rank the LDA representation of acoustic features, which are Min-Max normalised over Z-normalisation per speaker. By looking at the figure 4.3 we can still demonstrate that MFCC, loudness, F2, F3 are the most important and discriminative features for our task. When we look at the figure 4.2, we can see that F1, F2, F3 and F0semitone features are highly correlated with each other, which is expected that they are all voice related. Eventually, we have selected the top 4 features with the highes Fisher scores (i.e. MFCC, loudness, F2 and F3) while considering F2 and F3 are highly correlated with each other. Hence, we have used top-ranked LDAs and their related acoustic information as an input for our proposed model, whereby we aim to enrich the discriminability of summary-candidate sentences in terms of linguistic and acoustic features.

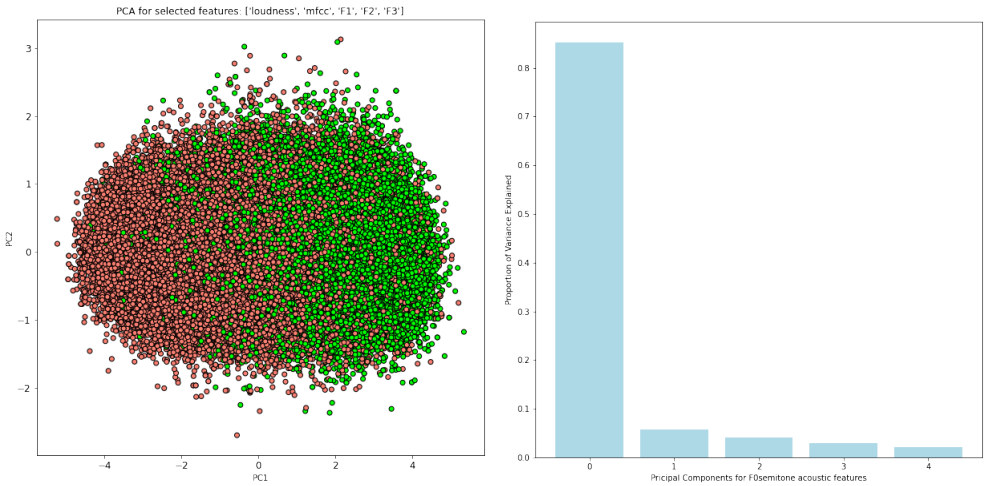

Figure 4.1: On the left figure, PCA visualisation of selected projected(LDA) acoustics features are visualised: green variables indicate the sentence which are labelled as one, which potentially carry saliency for the summary in terms of acoustics, and red values labelled as 0 that is not considerable candidate sentences for summary, on the right figure demonstrates the proportions of variance explained in the components

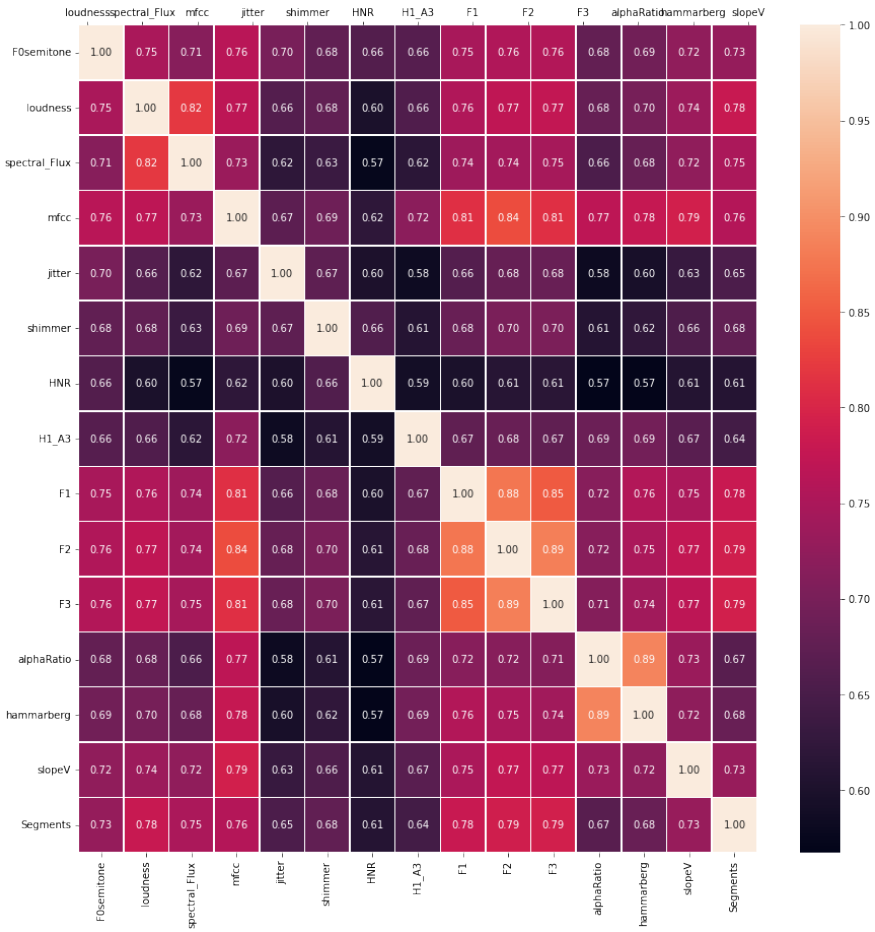

Figure 4.2: Heatmap of acoustic features

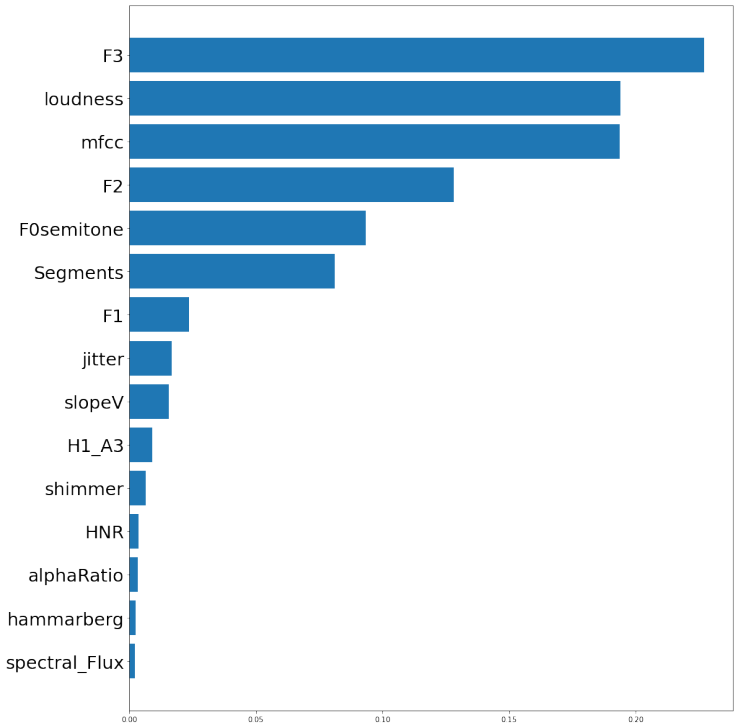

Figure 4.3: Feature Ranking by executing XGBoost

4.2 Results

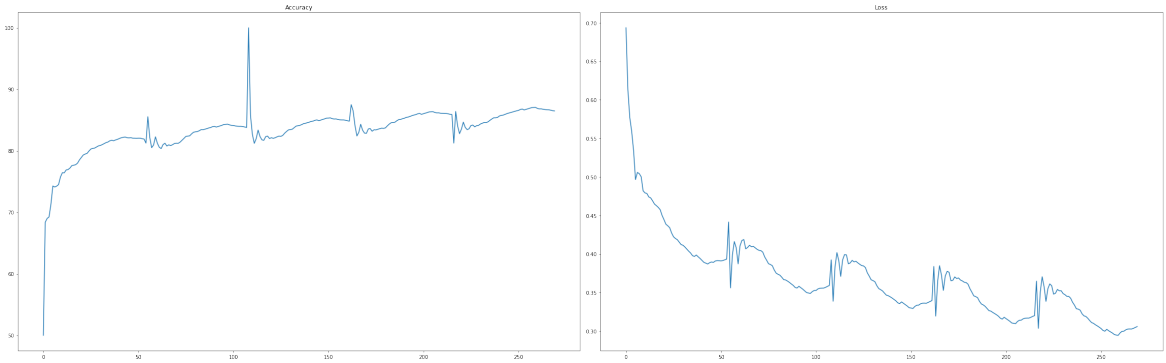

Figure 4.4: Model training accuracy (left) and loss (right) graphs over 5 epochs

Method	Selected Acoustic Features	# of Acoustic Features	AUC	F1	Sensitivity	Specificity	Accuracy
Text-Only	N/A	N/A	0.793	0.737	0.741	0.837	0.804
Multimodal	LDAs of MFCC, F3,F2,Loudness	4	**0.810**	**0.759**	0.754	0.851	**0.821**
Multimodal	MFCC	4	**0.803**	**0.752**	0.764	0.841	**0.812**
Multimodal	Loudness	4	0.795	0.732	0.747	0.843	0.806
Multimodal	F2	4	0.792	0.739	0.719	0.865	0.801
Multimodal	F3	4	0.774	0.717	0.719	0.830	0.788

Table 4.2: Evaluation of proposed and baseline models

Table 4.2 shows the results of our experiment on the validation set concerning the acoustic features utilised for the classification task; simply put, we have trained the model to assign scores for sentences based on the input text representation.

After that, with a threshold of 0.5, we classify the sentences as 1 if its score is above than the threshold, and classify as 0 if the score of the sentence is below than 0.5 (meaning that the sentence is not representative enough for including in the summary). As prescribed by the task, the essential evaluation metrics are the AUC and F1 scores. All scores are obtained using SentenceBERT with 2-layer MLP models trained with the 5-Fold validation technique on the training set and evaluated on the validation set. To demonstrate the importance of features, the setups with acoustic features are based on the top 4 acoustic features with the highest Fisher scores computed in the preprocess task, shown in table 4.1. The best-scoring setup is used for the summarisation task for the final submission. In comparison to baseline, LDA representation of selected features and sentence embeddings leads to better results. As a result, the multimodal model with represented LDA features outperforms the Text-Only model by 0.02, with an AUC score of 0.81 and F-1 score of 0.76.

To establish the information gain of acoustic features, we train the model with selected acoustic features specifically. Compared to Text-Only, the multimodal model with MFCC features has given slightly better results with an AUC and F-1 scores of 0.80 and 0.75. With an AUC-score of 0.71, the multimodal model with F3 acoustic features has given the worst result. However, the information contained in the LDA representation of F3 helped improve the performance of the best performing setup. Based on the results, we are able to answer our third sub-research question here, *"To what extent do the selection, projection and representation of acoustic features improve the accuracy of the extractive summarisation systems?"*. We can demonstrate that supervised based projection leads to better classification results and respectively, single type acoustic features might be also useful to improve accuracy. On that account, we are able indicate our first hypothesis *"When it comes to speech summarisation task, the saliency of information exists not only in the textual data but also in the audio signals, as the importance of what is said correlates with how it is said in the spoken audio contents."* becomes moderately true for the projected and various selected acoustic features (i.e. MFCC). Spefically, MFCC, loudness and related F0 (i.e.F2, F3) features seems to be the most representative acoustic features when it comes to detecting saliency in the speech.

	ROUGE-L			**ROUGE-1**			**ROUGE-2**		
	Recall	**Precision**	**F-1**	**Recall**	**Precision**	**F-1**	**Recall**	**Precision**	**F-1**
First-3	0.146	0.107	0.123	0.170	0.125	0.144	0.022	0.0123	0.015
Text-Only	0.200	0.112	0.134	0.248	0.141	0.179	0.034	0.017	0.020
Multimodal	0.176	**0.150**	**0.161**	0.209	**0.180**	**0.193**	0.036	**0.030**	**0.029**

Table 4.3: ROUGE score comparison for inference of the proposed model

When evaluating the model on the summarisation task with the ROUGE score, we have used the multimodal model with the LDA projections of MFCC, loudness, F2 and F3. We need to take into account that sentences labelled as 1, which have a ROUGE score better than 90 per cent of sentences in the transcript (based on the episode description), are actually from oracle summaries rather than ground truth. Thus, this might not be representative enough for labelling the sentences which potentially carry saliency relying on the acoustic features, which is detailed in the following discussion section.

When comparing the quality of generated summaries and the performance of the summarisation models, we have used ROUGE scores while setting the number of summary sentences for 3 and computed the scores based on the description of the podcast episodes in the test set. Table 4.3 shows the result of our summarisation experiment on the test set. To compare a relatively simple baseline, we compared the results with First-3 sentences and the Text-Only model built on SentenceBERT and 2-layer MLP. Looking at the table 4.3, the proposed multimodal model has considerable better results on ROUGE-L with an F-1 of 0.161, and ROUGE-1 with an F-1 score of 0.193. On that account, our second hypothesis *"The acoustic-linguistic multimodal approach can improve the performance of extractive summarise models compared to the performance of lexical-only models."* can be considered as substantially true in the case the F-1 scores of ROUGE-L and ROUGE-1. Though future scope and solving existing sub-problems discussed in the following section may be further improve the performance of this work.

4.3 Discussion & Conclusions

Summarisation

The best performing setup uses the SentenceBERT and 2-layer MLP with LDA representations of MFCC, loudness, F2, and F3 features. Initially, for combining acoustic and linguistic features, we had planned to use MAG (Multimodal Adaption Gate, discussed in the section 3.6.2); however, as this method takes word-level acoustic features into account, for our experiment, it was too computationally costly in terms of training time. For an alternative transformer model, we could have also used BERTSUM[56], which utilises the interconnection of sentences to produce extractive summaries, which might improve performance where the sentence-level acoustic features could also be interlinked. As we focused on sentence-level acoustic features, we had to neglect those missing acoustic values per word, where the potential saliency can rely on a word to highlight the context. However, this is what we also wanted to discover with this thesis, whereby this task can be considered a low resource scenario; simply put, making specific characteristics of words more explicit in terms of acoustics might lead to better performance to capture saliency.

Participants	Method	ROUGE-L Recall	ROUGE-L Precision	ROUGE-L F1
TREC - Baseline bartpodcasts	BART, Fine tuned on start of transcript	0.21	0.208	0.208
TREC - Baseline bartcnn	BART, No fine tuning	0.272	0.085	0.129
TREC - Baseline onemin	1 minute of transcript	0.282	0.087	0.132
TREC - Baseline TextRank	TextRank	0.165	0.083	0.110
U Glasgow	15 sentence input, T5	0.167	0.237	0.195
U Glasgow	Extractive filtering, SpanBert	0.147	0.22	0.176
U Delaware	Start of transcript, BART	0.161	0.239	0.192
U Delaware	Select sentences by LDA, BART	0.168	0.202	0.183
U Delaware	Select sentences by ROUGE, BART	0.16	0.208	0.180

Table 4.4: ROUGE scores of predicted abstractive summaries generated by the baseline and some of proposed models at TREC 2020 [41]

Moreover, to exploit saliency more in the acoustic information, using more or larger MLP on the top of SentenceBERT might also lead to better results. In addition, as

we discussed in the section 3.6.2, fusing acoustic information at earlier layers of the transformer architecture might lead to better results. Since in semantic space, they allow the sentence vector shifting to happen from the beginning of the network, the higher layers of BERT capture more abstract and higher-level information about the semantic structure of linguistic features. Eventually, it will be more difficult for the model to shift the sentence vector extracted from the transformer, which will be already very abstractive in nature. Hence, the injection of acoustic features in earlier layers could be worth exploring in future work. Possibly assigning more weights to the acoustic features leads better to exploiting acoustic information so the impact of prosody can be more demonstrated.

We also need to mention that extractive models suffer from errors sourced by ASR and the natural disfluency of spoken language; in contrast, abstractive models aim to generalise over these errors and generate relatively fluent written language. Additionally, extractive models aim to detect salient parts of the transcripts where these parts might not translate to an overview of the episode, whereas the abstractive models are capable of generating an overview statement from the transcript. As in TREC 2020, they aim to generate abstractive summaries, and some participants have utilised both extractive and abstractive approaches without using acoustic features (see Table 4.4). Even though we cannot directly compare the models by the participants due to the natural difference of summary forms of abstractive and extractive outputs, we can demonstrate from the table 4.4 that our multimodal has potential to integrate the acoustic features for further implementations according to the baseline models in TREC 2020. Hence We can suggest that a combination of extractive and abstractive approaches (to extract the salient part in a multimodal way and produce an overview statement in an abstractive manner) might improve the quality of summaries while utilising acoustic features.

Normalisation

When analysing the impact of different normalisation techniques, we have demonstrated that speaker normalisation has boosted the discriminability power of acoustic features compared to the normalisation technique on the overall corpus. It is though worth mentioning that the host's sentences might provide better overview statements about the podcast episode; this can be solved by annotating the host speaker and putting more weight on the sentences spoken by the host. Moreover, other vocal normalisation techniques can be utilised as the applied techniques in this experiment might be too static for handling the vocal variations. As we mentioned in the section 3.2, automatic speaker diarisation does not perform well, which might lead not to reaching the unit spheres for acoustic features per speaker. We expect that having access to the higher accuracy rate for automatically speaker tagging will further increase acoustic classifiers' performance and discriminability power along with speaker normalisation. While considering errors in automatic diarisation, which is a subtask of ASR, it is also worth mentioning that the transcripts carry 18.2% word error rate in the automatic transcription. This also lowers the performance of classifying sentences based on linguistic and acoustic features, where the important words or bits might be discarded due to the noisy data. This can be investigated by manual transcription and check which words or bits are actually removed and whether these errors carry potential effect on the performance of the specific tasks.

Acoustic Features

In this setup, LDA projections of the selected acoustic features are made available to the model, indicating that, perhaps not surprisingly, supervised based projection leads to better classification results. But of course, this extreme dimensionality reduction also has downsides, such as if the distributions are significantly non-Gausissian, the LDA projections may not preserve complex structure in the data needed for classification. Moreover, LDA might also fail if discriminatory information is not in the mean but the variance of the data, and it is sensitive to overfit and not applicable for non-linear problems. For an alternative for feature projection methods, t-SNE can be utilised, which is a non-linear technique for dimensionality reduction. As feature selection has an important impact on the performance of the classification task and influences the results, we also need to take into consideration of other feature selection methods. For instance, instead of using the Fisher score, which is a static filtering method for features, wrapper methods (i.e. Forward-Backward feature selection, Recursive feature elimination) could also be utilised for feature selection. In wrapper methodology, it becomes a search problem where the different combinations of features are made, evaluated and compared with other combinations. It trains the algorithm by using the subset of features iteratively. In such a manner, instead of selecting the same type of acoustic features (e.g. all MFCC features) or their LDA representation, combining different single acoustic features which carry the highest importance based on the feature selection algorithm might improve the classification task performance.

When it comes to adding acoustic features, we need to mention the generation of oracle summaries and their impact on our experiment. As we do not have a ground-truth summary set for our case (or labelled sentences in podcast episodes), we had to create the oracle summaries (top-ranked sentences in the 90th percentile, based on ROUGE scores) from the description of the podcast episodes. However, they might still not be representative enough for the podcast's overall dialogue and acoustic saliency. According to that, adding more acoustic features to detect better the oracle summary sentences in the transcripts might not work for this case. Hence this extractive method might not be the perfect fit for this data. Nevertheless, this can be solved by manually annotating the important sentences in terms of prosodic and linguistic information might allow adding more acoustic features as they become naturally representative by humans. Moreover, if there are better representative description of the episode generated by the podcast producer, might lead to have better oracle summaries. Instead of using ROUGE for generating oracle summaries, contextualized embeddings (i.e. BERT[17] and BLEURT[88]) might produce better oracle summaries as they are trained to manage ordering and dependencies with context. In the concern of audio-enhanced retrieval mechanism, we can also exploit another precomputed acoustic features [1] (i.e. YAMNet embedding vector) where we can detect specific audio events of interests (e.g. laughter, cry) that might lead to improve detecting salient parts of the dialogue. All in all, we do expect the suggestions mentioned above for future work to improve the performance of the proposed model, though it is hard to say how much.

Bibliography

[1] A. Alexander, M. Mars, J. C. Tingey, *et al.*, "Audio features, precomputed for podcast retrieval and information access experiments," in *International Conference of the Cross-Language Evaluation Forum for European Languages*, Springer, 2021, pp. 3–14.

[2] Anhtv, *Neural network embedding and dense layers. what's the difference?* Dec. 2019. [Online]. Available: https://medium.com/logivan/neural-network-embedding-and-dense-layers-whats-the-difference-fa177c6d0304.

[3] S. J. Arora and R. P. Singh, "Automatic speech recognition: A review," *International Journal of Computer Applications*, vol. 60, no. 9, 2012.

[4] S. Balakrishnama and A. Ganapathiraju, "Linear discriminant analysis-a brief tutorial," *Institute for Signal and information Processing*, vol. 18, no. 1998, pp. 1–8, 1998.

[5] R. Banerjee, "Principal component analysis," *S. NO. TOPIC PAGE*, p. 185,

[6] R. Bellman and R. Kalaba, "Dynamic programming and statistical communication theory," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 43, no. 8, p. 749, 1957.

[7] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, 2015.

[8] B. Chen, S.-H. Lin, Y.-M. Chang, and J.-W. Liu, "Extractive speech summarization using evaluation metric-related training criteria," *Information Processing & Management*, vol. 49, no. 1, pp. 1–12, 2013.

[9] K.-Y. Chen, S.-H. Liu, B. Chen, *et al.*, "Extractive broadcast news summarization leveraging recurrent neural network language modeling techniques," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 8, pp. 1322–1334, 2015.

[10] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16, San Francisco, California, USA: ACM, 2016, pp. 785–794, ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785.

[11] S. Chitrakala, N. Moratanch, B. Ramya, C. R. Raaj, and B. Divya, "Concept-based extractive text summarization using graph modelling and weighted iterative ranking," in *International Conference on Emerging Research in Computing, Information, Communication and Applications*, Springer, 2016, pp. 149–160.

[12] K. W. Church, "Word2vec," *Natural Language Engineering*, vol. 23, no. 1, pp. 155–162, 2017.

[13] A. Clifton, S. Reddy, Y. Yu, *et al.*, "100,000 podcasts: A spoken english document corpus," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 5903–5917.

[14] R. Collobert and S. Bengio, "Links between perceptrons, mlps and svms," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 23.

[15] N. Cummins, S. Amiriparian, G. Hagerer, A. Batliner, S. Steidl, and B. W. Schuller, "An image-based deep spectrum feature representation for the recognition of emotional speech," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 478–484.

[16] S. Das, "Filters, wrappers and a boosting-based hybrid for feature selection," in *Icml*, Citeseer, vol. 1, 2001, pp. 74–81.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018.

[18] J. G. Dy and C. E. Brodley, "Feature selection for unsupervised learning," *Journal of machine learning research*, vol. 5, no. Aug, pp. 845–889, 2004.

[19] F. Eyben, K. R. Scherer, B. W. Schuller, *et al.*, "The geneva minimalistic acoustic parameter set (gemaps) for voice research and affective computing," *IEEE transactions on affective computing*, vol. 7, no. 2, pp. 190–202, 2015.

[20] F. Eyben and B. Schuller, "Opensmile: The munich open-source large-scale multimedia feature extractor," *ACM SIGMultimedia Records*, vol. 6, no. 4, pp. 4–13, 2015.

[21] X. Feng, X. Feng, and B. Qin, "A survey on dialogue summarization: Recent advances and new frontiers," 2021.

[22] L. Floridi and M. Chiriatti, "Gpt-3: Its nature, scope, limits, and consequences," *Minds and Machines*, vol. 30, no. 4, pp. 681–694, 2020.

[23] P. Fung, R. H. Y. Chan, and J. J. Zhang, "Rhetorical-state hidden markov models for extractive speech summarization," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2008, pp. 4957–4960.

[24] M. Gambhir and V. Gupta, "Recent automatic text summarization techniques: A survey," *Artificial Intelligence Review*, vol. 47, no. 1, pp. 1–66, 2017.

[25] K. Ganesan, "Rouge 2.0: Updated and improved measures for evaluation of summarization tasks," 2018.

[26] Q. Gu, Z. Li, and J. Han, "Generalized fisher score for feature selection," 2012.

[27] C. Guinaudeau and J. Hirschberg, "Accounting for prosodic information to improve asr-based topic tracking for tv broadcast news," in *12th Annual Conference of the International Speech Communication Association, Interspeech'11*, 2011, 4–pages.

[28] V. K. Gupta and T. J. Siddiqui, "Multi-document summarization using sentence clustering," in *2012 4th International Conference on Intelligent Human Computer Interaction (IHCI)*, IEEE, 2012, pp. 1–5.

[29] B. Haney, "Applied natural language processing for law practice," *Brian S. Haney, Applied Natural Language Processing for Law Practice*, 2020.

[30] T. Hasan, M. Abdelwahab, S. Parthasarathy, C. Busso, and Y. Liu, "Automatic composition of broadcast news summaries using rank classifiers trained with acoustic and lexical features," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2016, pp. 6080–6084.

[31] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.

[32] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception*, Elsevier, 1992, pp. 65–93.

[33] K. M. Hermann, T. Kocisky, E. Grefenstette, *et al.*, "Teaching machines to read and comprehend," *Advances in neural information processing systems*, vol. 28, pp. 1693–1701, 2015.

[34] J. Hirschberg and C. D. Manning, "Advances in natural language processing," *Science*, vol. 349, no. 6245, pp. 261–266, 2015.

[35] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[36] M. Honnibal and I. Montani, "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," To appear, 2017.

[37] C. Hori, S. Furui, R. Malkin, H. Yu, and A. Waibel, "Automatic speech summarization applied to english broadcast news speech," in *2002 Ieee International Conference on Acoustics, Speech, and Signal Processing*, IEEE, vol. 1, 2002, pp. I–9.

[38] A. Inoue, T. Mikami, and Y. Yamashita, "Improvement of speech summarization using prosodic information," in *Speech Prosody 2004, International Conference*, 2004.

[39] A. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," *IEEE transactions on pattern analysis and machine intelligence*, vol. 19, no. 2, pp. 153–158, 1997.

[40] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*, Springer, 1998, pp. 137–142.

[41] R. Jones, B. Carterette, A. Clifton, *et al.*, "Trec 2020 podcasts track overview," 2021.

[42] A. Joshi, E. Fidalgo, E. Alegre, and L. Fernández-Robles, "Summcoder: An unsupervised framework for extractive text summarization based on deep auto-encoders," *Expert Systems with Applications*, vol. 129, pp. 200–215, 2019.

[43] S. Kakouros and O. Räsänen, "Automatic detection of sentence prominence in speech using predictability of word-level acoustic features," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[44] O. Kalinli and S. S. Narayanan, "A saliency-based auditory attention model with applications to unsupervised prominent syllable detection in speech.," in *INTERSPEECH*, 2007, pp. 1941–1944.

[45] H. Karlbom and A. Clifton, "Abstractive podcast summarization using bart with longformer attention,"

[46] H. Kaya, A. A. Karpov, and A. A. Salah, "Fisher vectors with cascaded normalization for paralinguistic analysis," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[47] A. Khan, N. Salim, H. Farman, *et al.*, "Abstractive text summarization based on improved semantic graph approach," *International Journal of Parallel Programming*, vol. 46, no. 5, pp. 992–1016, 2018.

[48] D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural language processing: State of the art, current trends and challenges," 2017.

[49] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[50] I. Lehiste and N. J. Lass, "Suprasegmental features of speech," *Contemporary issues in experimental phonetics*, vol. 225, p. 239, 1976.

[51] M. Lewis, Y. Liu, N. Goyal, *et al.*, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," 2019.

[52] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text summarization branches out*, 2004, pp. 74–81.

[53] H. Liu Huan, "Feature extraction, construction and selection: A data mining perspective," 1998.

[54] H. Liu and H. Motoda, *Computational methods of feature selection*. CRC Press, 2007.

[55] H. Liu, R. Setiono, *et al.*, "A probabilistic approach to feature selection-a filter solution," in *ICML*, Citeseer, vol. 96, 1996, pp. 319–327.

[56] Y. Liu, "Fine-tune bert for extractive summarization," 2019.

[57] Y. Liu and S. Xie, "Impact of automatic sentence segmentation on meeting summarization," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2008, pp. 5009–5012.

[58] H. P. Luhn, "The automatic creation of literature abstracts," *IBM Journal of research and development*, vol. 2, no. 2, pp. 159–165, 1958.

[59] I. Mani and M. T. Maybury, "Automatic summarization," 2001.

[60] S. Maskey and J. Hirschberg, "Comparing lexical, acoustic/prosodic, structural and discourse features for speech summarization," in *Ninth European Conference on Speech Communication and Technology*, 2005.

[61] C. McCormick, "Word2vec tutorial-the skip-gram model," *Apr-2016.[Online]. Available: http://mccormickml. com/2016/04/19/word2vec-tutorial-the-skip-gram-model*, 2016.

[62] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[63] I. Mogotsi, *Christopher d. manning, prabhakar raghavan, and hinrich schütze: Introduction to information retrieval*, 2010.

[64] N. Moratanch and S. Chitrakala, "A novel framework for semantic oriented abstractive text summarization," *Journal of Web Engineering*, vol. 17, no. 8, pp. 675–716, 2018.

[65] G. Murray and G. Carenini, "Summarizing spoken and written conversations," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 2008, pp. 773–782.

[66] G. Murray and S. Renals, "Detecting action items in meetings," in *International Workshop on Machine Learning for Multimodal Interaction*, Springer, 2008, pp. 208–213.

[67] A. Nenkova and K. McKeown, "A survey of text summarization techniques," in *Mining text data*, Springer, 2012, pp. 43–76.

[68] F. Nihei, Y. I. Nakano, and Y. Takase, "Fusing verbal and nonverbal information for extractive meeting summarization," in *Proceedings of the Group Interaction Frontiers in Technology*, 2018, pp. 1–9.

[69] B. Nojavanasghari, D. Gopinath, J. Koushik, T. Baltrušaitis, and L.-P. Morency, "Deep multimodal fusion for persuasiveness prediction," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, 2016, pp. 284–288.

[70] M. Ostendorff, P. Bourgonje, M. Berger, J. Moreno-Schneider, G. Rehm, and B. Gipp, "Enriching bert with knowledge graph embeddings for document classification," 2019.

[71] P. Over and J. Yen, "Intrinsic evaluation of generic news text summarization systems," in *DUC 2003. Workshop on Text Summarization*, 2003.

[72] P. Owoicho and J. Dalton, "Glasgow representation and information learning lab (grill) at trec 2020 podcasts track,"

[73] M. G. de Pinto, M. Polignano, P. Lops, and G. Semeraro, "Emotions understanding model from spoken language using deep neural networks and mel-frequency cepstral coefficients," in *2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, IEEE, 2020, pp. 1–5.

[74] D. Prapayont and P. Satawedin, "Customer insights and media exposure of podcast listeners to present and promote podcast channels effectively on facebook in thailand," *Dusit Thani College Journal*, vol. 15, no. 2, pp. 251–265, 2021.

[75] Q. Qin, W. Hu, and B. Liu, "Feature projection for improved text classification," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 8161–8171. DOI: 10.18653/v1/2020.acl-main.726. [Online]. Available: https://aclanthology.org/2020.acl-main.726.

[76] J. R. Quinlan, "Learning decision tree classifiers," *ACM Computing Surveys (CSUR)*, vol. 28, no. 1, pp. 71–72, 1996.

[77] C. Raffel, N. Shazeer, A. Roberts, *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2019.

[78] W. Rahman, M. K. Hasan, S. Lee, *et al.*, "Integrating multimodal information in large pretrained transformers," in *Proceedings of the conference. Association for Computational Linguistics. Meeting*, NIH Public Access, vol. 2020, 2020, p. 2359.

[79] M. Ramezani and M.-R. Feizi-Derakhshi, "Automated text summarization: An overview," *Applied Artificial Intelligence*, vol. 28, no. 2, pp. 178–215, 2014.

[80] N. Rane and S. Govilkar, "Recent trends in deep learning based abstractive text summarization," *Int. J. Recent Technol. Eng.*, vol. 8, no. 3, 2019.

[81] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *CoRR*, vol. abs/1908.10084, 2019. eprint: 1908.10084.

[82] R. Rezapour, S. Reddy, A. Clifton, and R. Jones, "Spotify at trec 2020: Genre-aware abstractive podcast summarization," 2021.

[83] D. Rezazadegan, S. Berkovsky, J. C. Quiroz, *et al.*, "Automatic speech summarisation: A scoping review," 2020.
[84] S. Ruder, "An overview of gradient descent optimization algorithms," 2016.
[85] Y. Rui, T. S. Huang, and S.-F. Chang, "Image retrieval: Current techniques, promising directions, and open issues," *Journal of visual communication and image representation*, vol. 10, no. 1, pp. 39–62, 1999.
[86] G. Salton, A. Wong, and C.-S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.
[87] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," 2017.
[88] T. Sellam, D. Das, and A. P. Parikh, "Bleurt: Learning robust metrics for text generation," 2020.
[89] G. Shang, "Spoken language understanding for abstractive meeting summarization," 2021.
[90] A. A. Syed, F. L. Gaol, and T. Matsuo, "A survey of the state-of-the-art models in neural abstractive text summarization," *IEEE Access*, vol. 9, pp. 13 248–13 265, 2021.
[91] J. Tang, H. Li, Y. Cao, and Z. Tang, "Email data cleaning," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 2005, pp. 489–498.
[92] V. Tretyak and D. Stepanov, "Combination of abstractive and extractive approaches for summarization of long scientific texts," 2020.
[93] Y.-H. H. Tsai, S. Bai, P. P. Liang, J. Z. Kolter, L.-P. Morency, and R. Salakhutdinov, "Multimodal transformer for unaligned multimodal language sequences," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 6558–6569. DOI: 10.18653/v1/P19-1656. [Online]. Available: https://aclanthology.org/P19-1656.
[94] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Information processing & management*, vol. 50, no. 1, pp. 104–112, 2014.
[95] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
[96] M. Wagner, "Prosody and recursion in coordinate structures and beyond," *Natural Language & Linguistic Theory*, vol. 28, no. 1, pp. 183–237, 2010.
[97] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, "Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5776–5788, 2020.
[98] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping, "Use of the zero norm with linear models and kernel methods," *The Journal of Machine Learning Research*, vol. 3, pp. 1439–1461, 2003.
[99] A. P. Widyassari, S. Rustad, G. F. Shidik, E. Noersasongko, A. Syukur, A. Affandy, *et al.*, "Review of automatic text summarization techniques & methods," *Journal of King Saud University-Computer and Information Sciences*, 2020.
[100] T. Wolf, J. Chaumond, L. Debut, *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empir-*

ical Methods in Natural Language Processing: System Demonstrations, 2020, pp. 38–45.

[101] S. Xie, D. Hakkani-Tür, B. Favre, and Y. Liu, "Integrating prosodic features in extractive meeting summarization," in *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*, IEEE, 2009, pp. 387–391.

[102] X. Yu, M. O. Efe, and O. Kaynak, "A general backpropagation algorithm for feedforward neural networks learning," *IEEE transactions on neural networks*, vol. 13, no. 1, pp. 251–254, 2002.

[103] J. J. Zhang, R. H. Y. Chan, and P. Fung, "Extractive speech summarization using shallow rhetorical structure modeling," *IEEE transactions on audio, speech, and language processing*, vol. 18, no. 6, pp. 1147–1157, 2009.

[104] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: A statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1, pp. 43–52, 2010.

[105] Z. Zhang, S. Blair-Goldensohn, and D. R. Radev, "Towards cst-enhanced summarization," in *Aaai/Iaai*, 2002, pp. 439–446.

[106] Z. Zhao, H. Pan, C. Fan, *et al.*, "Abstractive meeting summarization via hierarchical adaptive segmental network learning," pp. 3455–3461, 2019.

[107] C. Zheng, H. J. Wang, K. Zhang, and L. Fan, "A baseline analysis for podcast abstractive summarization," 2020.

[108] C. Zheng, K. Zhang, H. J. Wang, and L. Fan, "A two-phase approach for abstractive podcast summarization," 2020.

[109] M. Zhong, D. Yin, T. Yu, *et al.*, "Qmsum: A new benchmark for query-based multi-domain meeting summarization," 2021.

[110] W. X. Zhu, *Hotspot detection for automatic podcast trailer generation*, 2021.

Appendix A

Feature Ranking

Parameter Groups	LLD Codes	LLD Names	Description
Frequency	F0semitone	Pitch	logarithmic F0 on a semitone frequency scale, starting at 27.5 Hz (semitone 0).
Frequency	jitter	Jitter	deviations in individual consecutive F0 period lengths
Frequency	F1, F2, F3	Formant 1, 2, and 3 frequency	centre frequency of first, second, and third formant
Energy/Amplitude	shimmer	Shimmer	difference of the peak amplitudes of consecutive F0 periods
Energy/Amplitude	loudness	Loudness	estimate of perceived signal intensity from an auditory spectrum
Energy/Amplitude	HNR	Harmonics-to-Noise Ratio (HNR),	relation of energy in harmonic components to energy in noiselike components.
Spectral	alphaRatio	Alpha Ratio	ratio of the summed energy from 50–1000 Hz and 1–5 kHz
Spectral	hammarberg	Hammarberg Index	ratio of the strongest energy peak in the 0–2 kHz region to the strongest peak in the 2–5 kHz region.
Spectral	slopeV	Spectral Slope 0–500 Hz and 500–1500 Hz	Linear regression slope of the logarithmic power spectrum within the two given bands.
Spectral	$H1_A3$	Harmonic difference H1–A3,	Harmonic difference H1–A3, ratio of energy of the first F0 harmonic (H1) to the energy of the highest harmonic in the third formant range (A3)
Spectral	F1, F2, F3 relative energy	Formant 1, 2, and 3 relative energy,	difference of the formant relative energy of consecutive F0 periods
Spectral	$spectral_Flux$	Spectral flux	difference of the spectra of two consecutive frames.
Spectral	mfcc	MFCC 1–4	the overall shape of a spectral envelope
Temporal Features	Segments	Segments	duration-related attributes

Table 4.5: Acoustic Feature Dictionary table

Rank Index	LLD	Feature	Fisher.score	Rank Index	LLD	Feature	Fisher.score
0	**mfcc**	MAX:mfcc3V.sma3nz.amean	0.1139027975	**0**	**loudness**	MAX:loudnessPeaksPerSec	0.1038524903
1	**mfcc**	MAX:mfcc3.sma3.amean	0.1089485044	**1**	**loudness**	MIN:loudness.sma3.stddevFallingSlope	0.08758928117
2	**mfcc**	MAX:mfcc2.sma3.amean	0.09806844924	**2**	**loudness**	MIN:loudness.sma3.stddevRisingSlope	0.08669845552
3	**mfcc**	MAX:mfcc2V.sma3nz.amean	0.09548026742	**3**	**loudness**	MIN:loudness.sma3.stddevNorm	0.08069300893
4	**mfcc**	MAX:mfcc1.sma3.amean	0.08568629682	**4**	**loudness**	MIN:loudness.sma3.percentile50.0	0.07965065413
5	**mfcc**	MAX:mfcc1V.sma3nz.amean	0.08337249878	**5**	**loudness**	MIN:loudnessPeaksPerSec	0.07911664487
6	**mfcc**	MIN:mfcc1.sma3.amean	0.08247915184	**6**	**loudness**	MIN:loudness.sma3.meanRisingSlope	0.07606261212
7	**mfcc**	MAX:mfcc4.sma3.amean	0.07819162351	**7**	**loudness**	MIN:loudness.sma3.pctlrange0-2	0.07473404351
8	**mfcc**	MAX:mfcc4V.sma3nz.amean	0.07495739347	**8**	**loudness**	MAX:loudness.sma3.stddevNorm	0.07398617868
9	**mfcc**	MIN:mfcc4.sma3.amean	0.06857084908	**9**	**loudness**	MAX:loudness.sma3.stddevFallingSlope	0.07021551724
10	**mfcc**	MIN:mfcc1V.sma3nz.amean	0.06820528272	**10**	**loudness**	MIN:loudness.sma3.percentile80.0	0.06917178775
11	**mfcc**	MIN:mfcc4V.sma3nz.amean	0.06709066825	**11**	**loudness**	MIN:loudness.sma3.meanFallingSlope	0.06897947517
12	**mfcc**	MIN:mfcc3.sma3.amean	0.06273428865	**12**	**loudness**	MIN:loudness.sma3.amean	0.06839953699
13	**mfcc**	MIN:mfcc3V.sma3nz.amean	0.06060136164	**13**	**loudness**	MAX:loudness.sma3.meanFallingSlope	0.06520759159
14	**mfcc**	MIN:mfcc2.sma3.amean	0.05915369109	**14**	**loudness**	MAX:loudness.sma3.stddevRisingSlope	0.06477740073
15	**mfcc**	MIN:mfcc2V.sma3nz.amean	0.05287312356	**15**	**loudness**	MAX:loudness.sma3.meanRisingSlope	0.06361371105
16	**mfcc**	MIN:mfcc1V.sma3nz.stddevNorm	0.04080295368	**16**	**loudness**	MIN:loudness.sma3.percentile20.0	0.06322701409
17	**mfcc**	MIN:mfcc1.sma3.stddevNorm	0.0320425166	**17**	**loudness**	MAX:loudness.sma3.pctlrange0-2	0.05735311518
18	**mfcc**	MAX:mfcc1V.sma3nz.stddevNorm	0.03138561726	**18**	**loudness**	MAX:loudness.sma3.percentile20.0	0.04792144896
19	**mfcc**	MAX:mfcc1.sma3.stddevNorm	0.02465012466	**19**	**loudness**	MAX:loudness.sma3.percentile50.0	0.0446446182
20	**mfcc**	MIN:mfcc4V.sma3nz.stddevNorm	0.01440881922	**20**	**loudness**	MAX:loudness.sma3.percentile80.0	0.04379811744
21	**mfcc**	MAX:mfcc2V.sma3nz.stddevNorm	0.01352078533	**21**	**loudness**	MAX:loudness.sma3.amean	0.04022307067
22	**mfcc**	MAX:mfcc3.sma3.stddevNorm	0.01298729425	**22**	**loudness**	MEAN:loudnessPeaksPerSec	0.003259240629
23	**mfcc**	MAX:mfcc2.sma3.stddevNorm	0.01284777352	**23**	**loudness**	MEAN:loudness.sma3.pctlrange0-2	0.0007837882521
24	**mfcc**	MIN:mfcc2V.sma3nz.stddevNorm	0.01242655965	**24**	**loudness**	MEAN:loudness.sma3.stddevNorm	0.0005827786202
25	**mfcc**	MIN:mfcc2.sma3.stddevNorm	0.0122556007	**25**	**loudness**	MEAN:loudness.sma3.percentile80.0	0.0005530899097
26	**mfcc**	MIN:mfcc3.sma3.stddevNorm	0.01203441126	**26**	**loudness**	MEAN:loudness.sma3.amean	0.0003930662849
27	**mfcc**	MAX:mfcc4V.sma3nz.stddevNorm	0.01171627038	**27**	**loudness**	MEAN:loudness.sma3.percentile50.0	0.0002833604014
28	**mfcc**	MAX:mfcc4.sma3.stddevNorm	0.01166358192	**28**	**loudness**	MEAN:loudness.sma3.stddevFallingSlope	0.0001216529684
29	**mfcc**	MIN:mfcc4.sma3.stddevNorm	0.01134601526	**29**	**loudness**	MEAN:loudness.sma3.meanFallingSlope	4.03E-05
30	**mfcc**	MAX:mfcc3V.sma3nz.stddevNorm	0.009391627344	**30**	**loudness**	MEAN:loudness.sma3.meanRisingSlope	3.49E-05
31	**mfcc**	MIN:mfcc3V.sma3nz.stddevNorm	0.008195348318	**31**	**loudness**	MEAN:loudness.sma3.percentile20.0	1.30E-05
32	**mfcc**	MEAN:mfcc3.sma3.amean	0.005163122346	**32**	**loudness**	MEAN:loudness.sma3.stddevRisingSlope	1.03E-05
33	**mfcc**	MEAN:mfcc3V.sma3nz.amean	0.005009616236	**0**	**jitter**	MAX:jitterLocal.sma3nz.stddevNorm	0.09855367475
34	**mfcc**	MEAN:mfcc2V.sma3nz.amean	0.003160284068	**1**	**jitter**	MIN:jitterLocal.sma3nz.stddevNorm	0.09200321085
35	**mfcc**	MEAN:mfcc2.sma3.amean	0.002425916633	**2**	**jitter**	MIN:jitterLocal.sma3nz.amean	0.08214301674
36	**mfcc**	MEAN:mfcc1V.sma3nz.amean	0.0007456710953	**3**	**jitter**	MAX:jitterLocal.sma3nz.amean	0.05948951029
37	**mfcc**	MEAN:mfcc1.sma3.amean	0.0006825244804	**4**	**jitter**	MEAN:jitterLocal.sma3nz.amean	0.002384862178
38	**mfcc**	MEAN:mfcc4V.sma3nz.amean	0.0003298585055	**5**	**jitter**	MEAN:jitterLocal.sma3nz.stddevNorm	0.0002004512927
39	**mfcc**	MEAN:mfcc4.sma3.amean	0.0001145020442	**0**	**shimmer**	MAX:shimmerLocaldB.sma3nz.stddevNorm	0.08821006297
40	**mfcc**	MEAN:mfcc1V.sma3nz.stddevNorm	8.47E-05	**1**	**shimmer**	MIN:shimmerLocaldB.sma3nz.amean	0.08488267005
41	**mfcc**	MEAN:mfcc1.sma3.stddevNorm	3.28E-05	**2**	**shimmer**	MAX:shimmerLocaldB.sma3nz.amean	0.07251590127
42	**mfcc**	MEAN:mfcc4V.sma3nz.stddevNorm	1.33E-05	**3**	**shimmer**	MIN:shimmerLocaldB.sma3nz.stddevNorm	0.06780240744
43	**mfcc**	MEAN:mfcc4.sma3.stddevNorm	9.30E-06	**3**	**shimmer**	MEAN:shimmerLocaldB.sma3nz.stddevNorm	0.001276184916
44	**mfcc**	MEAN:mfcc3V.sma3nz.stddevNorm	7.18E-06	**4**	**shimmer**	MEAN:shimmerLocaldB.sma3nz.amean	0.0002800090557

Table 4.6: Fisher scores (1)

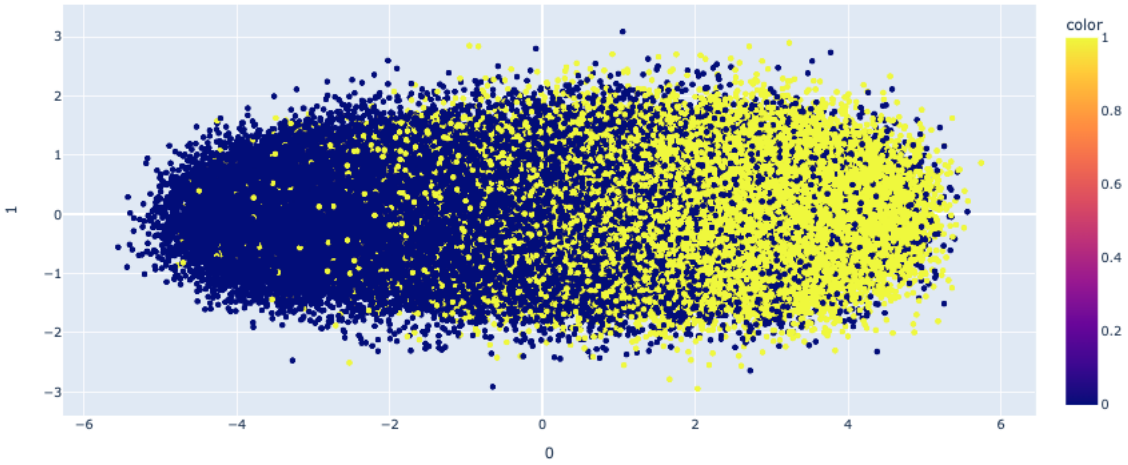

Figure 4.5: PCA visualisation of all projected(LDA) acoustics features: yellow variables indicate the sentence which are labelled as one, which potentially carry saliency for the summary in terms of acoustics, and blue values labelled as 0 that is not considerable candidate sentences for summary

index	LLD	Feature	Fisher.score	index	LLD	Feature	0.08972382049
0	F2	MIN:F2frequency.sma3nz.stddevNorm	0.09669684836	0	F0semitone	MIN:F0semitoneFrom27.5Hz.sma3nz.stddevNorm	0.08284968093
1	F2	MIN:F2bandwidth.sma3nz.amean	0.09325540665	1	F0semitone	MIN:F0semitoneFrom27.5Hz.sma3nz.pctlrange0-2	0.07471556154
2	F2	MAX:F2frequency.sma3nz.stddevNorm	0.09255340381	2	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.stddevNorm	0.07328122335
3	F2	MAX:F2bandwidth.sma3nz.stddevNorm	0.090454476	3	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.stddevRisingSlope	0.07070949542
4	F2	MAX:F2frequency.sma3nz.amean	0.08538006755	4	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.meanRisingSlope	0.06442839295
5	F2	MIN:F2amplitudeLogRelF0.sma3nz.amean	0.08318453744	5	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.pctlrange0-2	0.06204007763
6	F2	MIN:F2bandwidth.sma3nz.stddevNorm	0.08278557687	6	F0semitone	MIN:F0semitoneFrom27.5Hz.sma3nz.amean	0.0601805502
7	F2	MAX:F2amplitudeLogRelF0.sma3nz.amean	0.0753668946	7	F0semitone	MIN:F0semitoneFrom27.5Hz.sma3nz.meanRisingSlope	0.05828500217
8	F2	MAX:F2bandwidth.sma3nz.amean	0.07299047552	8	F0semitone	MIN:F0semitoneFrom27.5Hz.sma3nz.percentile80.0	0.0580462294
9	F2	MIN:F2frequency.sma3nz.amean	0.06790185054	9	F0semitone	MIN:F0semitoneFrom27.5Hz.sma3nz.percentile20.0	0.05607466927
10	F2	MAX:F2amplitudeLogRelF0.sma3nz.stddevNorm	0.05236546652	10	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.amean	0.05362017134
11	F2	MIN:F2amplitudeLogRelF0.sma3nz.stddevNorm	0.03069995478	11	F0semitone	MIN:F0semitoneFrom27.5Hz.sma3nz.percentile50.0	0.05042225401
12	F2	MEAN:F2bandwidth.sma3nz.stddevNorm	0.001465555965	12	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.percentile80.0	0.04896179908
13	F2	MEAN:F2frequency.sma3nz.amean	0.0004736845519	13	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.percentile20.0	0.04555138056
14	F2	MEAN:F2bandwidth.sma3nz.amean	0.0003543781874	14	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.percentile50.0	0.04220459247
15	F2	MEAN:F2amplitudeLogRelF0.sma3nz.stddevNorm	0.0002470908359	15	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.stddevFallingSlope	0.04167048351
16	F2	MEAN:F2amplitudeLogRelF0.sma3nz.amean	0.0001842717773	16	F0semitone	MAX:F0semitoneFrom27.5Hz.sma3nz.meanFallingSlope	0.003376642945
17	F2	MEAN:F2frequency.sma3nz.stddevNorm	2.34E-05	17	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.pctlrange0-2	0.002114482923
0	F3	MIN:F3frequency.sma3nz.stddevNorm	0.09233944842	18	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.percentile80.0	0.001201300077
1	F3	MAX:F3bandwidth.sma3nz.stddevNorm	0.09047267868	19	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.stddevNorm	0.0007198580922
2	F3	MIN:F3bandwidth.sma3nz.stddevNorm	0.08484991797	20	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.percentile50.0	0.0005762622331
3	F3	MAX:F3frequency.sma3nz.amean	0.08448094619	21	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.meanRisingSlope	0.0004707836205
4	F3	MAX:F3frequency.sma3nz.stddevNorm	0.08387956303	22	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.amean	0.0004222402815
5	F3	MIN:F3amplitudeLogRelF0.sma3nz.amean	0.08236013276	23	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.stddevRisingSlope	0.0001911689724
6	F3	MIN:F3bandwidth.sma3nz.amean	0.08076176649	24	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.stddevFallingSlope	0.0001077415855
7	F3	MAX:F3bandwidth.sma3nz.amean	0.0794407398	25	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.meanFallingSlope	1.44E-05
8	F3	MAX:F3amplitudeLogRelF0.sma3nz.amean	0.07787764707	26	F0semitone	MEAN:F0semitoneFrom27.5Hz.sma3nz.percentile20.0	0.08237175385
9	F3	MAX:F3amplitudeLogRelF0.sma3nz.stddevNorm	0.06960906768	0	slopeV	MIN:slopeUV500-1500.sma3nz.amean	0.07826398177
10	F3	MIN:F3frequency.sma3nz.amean	0.06912953781	1	slopeV	MAX:slopeUV500-1500.sma3nz.amean	0.07748333781
11	F3	MIN:F3amplitudeLogRelF0.sma3nz.stddevNorm	0.0460906151	2	slopeV	MIN:slopeV500-1500.sma3nz.amean	0.06584472469
12	F3	MEAN:F3bandwidth.sma3nz.stddevNorm	0.000588243019	3	slopeV	MAX:slopeUV0-500.sma3nz.amean	0.06482084558
13	F3	MEAN:F3amplitudeLogRelF0.sma3nz.amean	0.000353854438	4	slopeV	MAX:slopeV500-1500.sma3nz.amean	0.06093393805
14	F3	MEAN:F3frequency.sma3nz.amean	0.0002594057655	5	slopeV	MIN:slopeUV0-500.sma3nz.amean	0.05470640362
15	F3	MEAN:F3frequency.sma3nz.stddevNorm	3.66E-05	6	slopeV	MIN:slopeV0-500.sma3nz.amean	0.05151850703
16	F3	MEAN:F3bandwidth.sma3nz.amean	1.27E-05	7	slopeV	MAX:slopeV0-500.sma3nz.amean	0.02577597993
17	F3	MEAN:F3amplitudeLogRelF0.sma3nz.stddevNorm	4.22E-08	8	slopeV	MAX:slopeV500-1500.sma3nz.stddevNorm	0.01941532302
0	F1	MIN:F1bandwidth.sma3nz.stddevNorm	0.08440053363	9	slopeV	MIN:slopeV500-1500.sma3nz.stddevNorm	0.01829014321
1	F1	MIN:F1frequency.sma3nz.stddevNorm	0.08412201456	10	slopeV	MAX:slopeV0-500.sma3nz.stddevNorm	0.01392657555
2	F1	MAX:F1bandwidth.sma3nz.stddevNorm	0.08236379608	11	slopeV	MIN:slopeV0-500.sma3nz.stddevNorm	0.0007372258259
3	F1	MAX:F1amplitudeLogRelF0.sma3nz.amean	0.08106462627	12	slopeV	MEAN:slopeV500-1500.sma3nz.amean	0.0001585145493
4	F1	MAX:F1frequency.sma3nz.stddevNorm	0.07998166069	13	slopeV	MEAN:slopeV0-500.sma3nz.amean	8.48E-05
5	F1	MIN:F1amplitudeLogRelF0.sma3nz.amean	0.07930770334	14	slopeV	MEAN:slopeUV500-1500.sma3nz.amean	4.48E-05
6	F1	MIN:F1bandwidth.sma3nz.amean	0.07895420599	15	slopeV	MEAN:slopeV0-500.sma3nz.stddevNorm	3.45E-05
7	F1	MAX:F1bandwidth.sma3nz.amean	0.07504022662	16	slopeV	MEAN:slopeUV0-500.sma3nz.amean	3.26E-05
8	F1	MIN:F1amplitudeLogRelF0.sma3nz.stddevNorm	0.02089061236	17	slopeV	MEAN:slopeV500-1500.sma3nz.stddevNorm	0.1057342946
9	F1	MAX:F1amplitudeLogRelF0.sma3nz.stddevNorm	0.02066706658	0	Segments	MAX:StddevVoicedSegmentLengthSec	0.08151016682
10	F1	MEAN:F1amplitudeLogRelF0.sma3nz.amean	0.0008239751451	1	Segments	MAX:MeanUnvoicedSegmentLength	0.07837937568
11	F1	MEAN:F1frequency.sma3nz.stddevNorm	0.0002395229795	2	Segments	MAX:StddevUnvoicedSegmentLength	0.07411773422
12	F1	MEAN:F1amplitudeLogRelF0.sma3nz.stddevNorm	0.0001781316717	3	Segments	MAX:VoicedSegmentsPerSec	0.07057101876
13	F1	MEAN:F1bandwidth.sma3nz.stddevNorm	0.0001209274199	4	Segments	MIN:MeanVoicedSegmentLengthSec	0.05396057421
14	F1	MEAN:F1bandwidth.sma3nz.amean	1.60E-05	5	Segments	MAX:equivalentSoundLevel.dBp	0.04936907061
				6	Segments	MIN:equivalentSoundLevel.dBp	0.0004240657109
				7	Segments	MEAN:StddevUnvoicedSegmentLength	0.0002323150349
				8	Segments	MEAN:StddevVoicedSegmentLengthSec	3.43E-05
				9	Segments	MEAN:equivalentSoundLevel.dBp	9.98E-06
				10	Segments	MEAN:VoicedSegmentsPerSec	3.04E-07
				11	Segments	MEAN:MeanVoicedSegmentLengthSec	1.59E-07
				12	Segments	MEAN:MeanUnvoicedSegmentLength	

Table 4.7: Fisher scores (2)

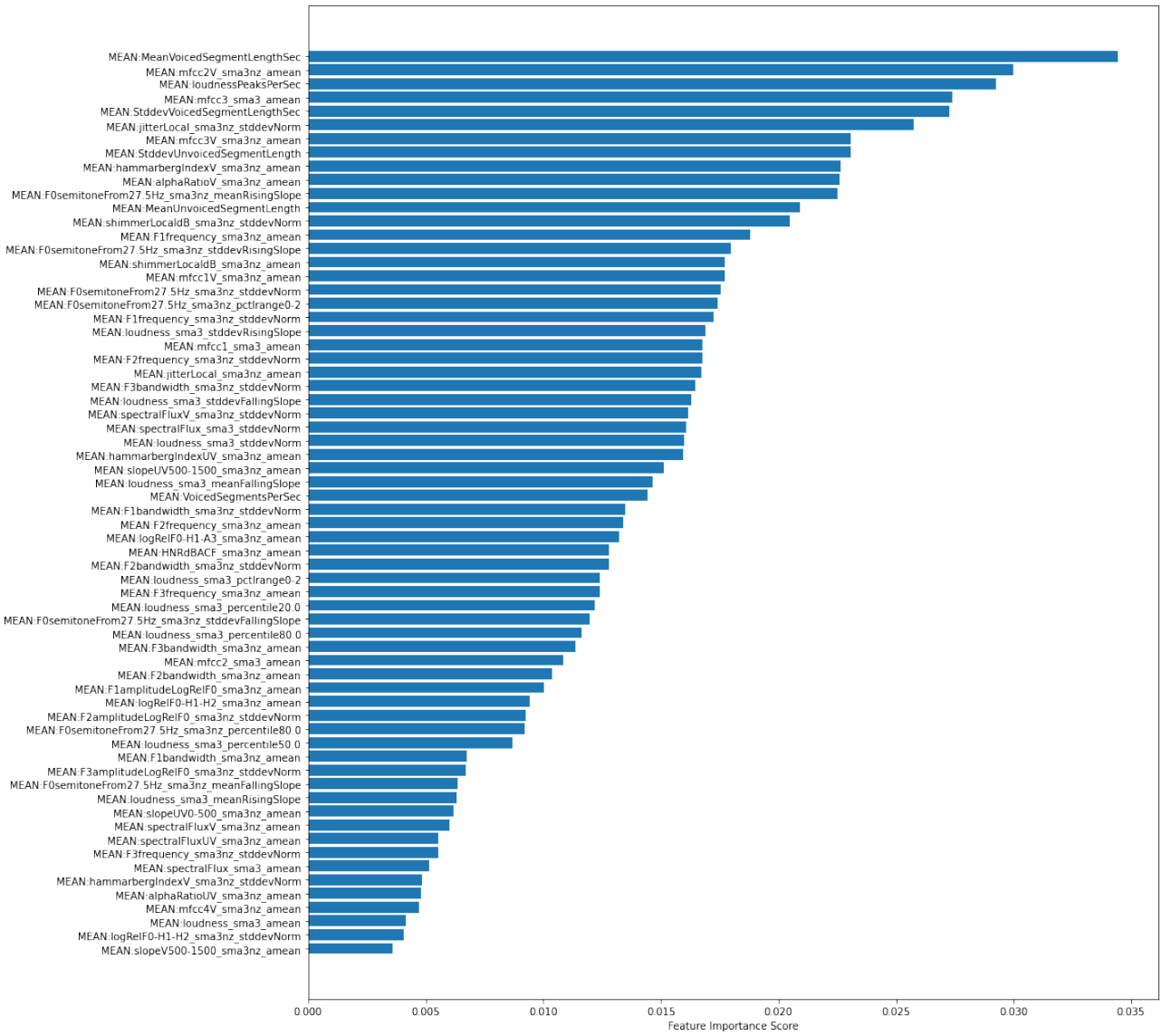

Figure 4.6: XGBoost Scores and Rankings for sentence-level means acoustic features

Appendix B

Model Components

```
==== Embedding Layer ====

l1.embeddings.word_embeddings.weight                    (30522, 384)
l1.embeddings.position_embeddings.weight                  (512, 384)
l1.embeddings.token_type_embeddings.weight                  (2, 384)
l1.embeddings.LayerNorm.weight                                (384,)
l1.embeddings.LayerNorm.bias                                  (384,)

==== Transformer ====

l1.encoder.layer.0.attention.self.query.weight            (384, 384)
l1.encoder.layer.0.attention.self.query.bias                  (384,)
l1.encoder.layer.0.attention.self.key.weight              (384, 384)
l1.encoder.layer.0.attention.self.key.bias                    (384,)
l1.encoder.layer.0.attention.self.value.weight            (384, 384)
l1.encoder.layer.0.attention.self.value.bias                  (384,)
l1.encoder.layer.0.attention.output.dense.weight          (384, 384)
l1.encoder.layer.0.attention.output.dense.bias                (384,)
l1.encoder.layer.0.attention.output.LayerNorm.weight          (384,)
l1.encoder.layer.0.attention.output.LayerNorm.bias            (384,)
l1.encoder.layer.0.intermediate.dense.weight             (1536, 384)
l1.encoder.layer.0.intermediate.dense.bias                   (1536,)
l1.encoder.layer.0.output.dense.weight                   (384, 1536)
l1.encoder.layer.0.output.dense.bias                          (384,)
l1.encoder.layer.0.output.LayerNorm.weight                    (384,)
l1.encoder.layer.0.output.LayerNorm.bias                      (384,)

==== Output Layer ====

MLPclassifier.0.weight                                   (768, 1157)
MLPclassifier.0.bias                                          (768,)
MLPclassifier.5.weight                                     (1, 768)
MLPclassifier.5.bias                                          (1,)
```

```
SentenceBertClass(
  (l1): BertModel(
    (embeddings): BertEmbeddings(
      (word_embeddings): Embedding(30522, 384, padding_idx=0)
      (position_embeddings): Embedding(512, 384)
      (token_type_embeddings): Embedding(2, 384)
      (LayerNorm): LayerNorm((384,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (encoder): BertEncoder(
      (layer): ModuleList(
        (0): BertLayer(
          (attention): BertAttention(
            (self): BertSelfAttention(
              (query): Linear(in_features=384, out_features=384, bias=True)
              (key): Linear(in_features=384, out_features=384, bias=True)
              (value): Linear(in_features=384, out_features=384, bias=True)
```

```
            (dropout): Dropout(p=0.1, inplace=False)
          )
          (output): BertSelfOutput(
            (dense): Linear(in_features=384, out_features=384, bias=True)
            (LayerNorm): LayerNorm((384,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
        (intermediate): BertIntermediate(
          (dense): Linear(in_features=384, out_features=1536, bias=True)
          (intermediate_act_fn): GELUActivation()
        )
        (output): BertOutput(
          (dense): Linear(in_features=1536, out_features=384, bias=True)
          (LayerNorm): LayerNorm((384,), eps=1e-12, elementwise_affine=True)
          (dropout): Dropout(p=0.1, inplace=False)
        )
      )
      (1): BertLayer(
        (attention): BertAttention(
          (self): BertSelfAttention(
            (query): Linear(in_features=384, out_features=384, bias=True)
            (key): Linear(in_features=384, out_features=384, bias=True)
            (value): Linear(in_features=384, out_features=384, bias=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
          (output): BertSelfOutput(
            (dense): Linear(in_features=384, out_features=384, bias=True)
            (LayerNorm): LayerNorm((384,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
        (intermediate): BertIntermediate(
          (dense): Linear(in_features=384, out_features=1536, bias=True)
          (intermediate_act_fn): GELUActivation()
        )
        (output): BertOutput(
          (dense): Linear(in_features=1536, out_features=384, bias=True)
          (LayerNorm): LayerNorm((384,), eps=1e-12, elementwise_affine=True)
          (dropout): Dropout(p=0.1, inplace=False)
        )
      )
      (2): BertLayer(
        (attention): BertAttention(
          (self): BertSelfAttention(
            (query): Linear(in_features=384, out_features=384, bias=True)
            (key): Linear(in_features=384, out_features=384, bias=True)
            (value): Linear(in_features=384, out_features=384, bias=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
          (output): BertSelfOutput(
            (dense): Linear(in_features=384, out_features=384, bias=True)
            (LayerNorm): LayerNorm((384,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
        (intermediate): BertIntermediate(
          (dense): Linear(in_features=384, out_features=1536, bias=True)
          (intermediate_act_fn): GELUActivation()
        )
        (output): BertOutput(
          (dense): Linear(in_features=1536, out_features=384, bias=True)
          (LayerNorm): LayerNorm((384,), eps=1e-12, elementwise_affine=True)
          (dropout): Dropout(p=0.1, inplace=False)
        )
      )
    )
  )
  (pooler): BertPooler(
    (dense): Linear(in_features=384, out_features=384, bias=True)
    (activation): Tanh()
  )
)
```

```
  (MLPclassifier): Sequential(
    (0): Linear(in_features=1157, out_features=768, bias=True)
    (1): ReLU()
    (2): Dropout(p=0.1, inplace=False)
    (3): ReLU()
    (4): Dropout(p=0.1, inplace=False)
    (5): Linear(in_features=768, out_features=1, bias=True)
    (6): Sigmoid()
  )
)
```

EPOCH	**0**	**1**	**2**	**3**	**4**
AUC	0.799	0.786	0.799	0.798	0.852
F1 score	0.739	0.725	0.744	0.744	0.817
Sensitivity	0.720	0.707	0.738	0.738	0.823
Specificity	0.839	0.829	0.845	0.845	0.894
Accuracy	0.811	0.798	0.811	0.811	0.863

Table 4.8: Validation results from training multimodal model over 5 epochs

Show Name	Millenials with Money	Major jobs with teland	The Cut on Tuesdays
Episode Description	the world caught us red handed. yep, our plot to conspire and decimate the economy has been foiled by brilliant, yet cranky gen-exers and baby boomers who weep at the thought of applebees going out of business. in this episode we will discuss why we can't help but systematically kill well-established goods and services.	today, i got to interview the band portugal. the man, who went to #4 in the billboard top 100 charts for their grammy-winning hit song "feel it still," on the off chance you don't know them: follow me on instagram:	cynthia nixon joins us this week as we consider the past and future of abortion. what does it look like to end a pregnancy when the law stands in your way? wendy zukerman from science vs tells us about a group of women in the 60s who developed their own safe abortion kit. and we hear from a dutch doctor who's made it her mission to distribute abortion pills in countries without access... which has meant taking to the seas, and taking on the us government.
First-3	You're listening to Millennials with money a podcast about the intricacies of an entitled generation. I'm Jose and I'm Charlie you're on official source of hot tags.And we're story straight from Millenial Urban America this podcast today is for my buddy and extra weight.	You are now listening to the major jobs podcast. Hello and welcome back to another episode of the major jobs podcast today. I got to talk with the grammy-winning band Portugal the man whose hit song feel it still went to number four in the Billboard Hot 100 charts and they were really fun to talk to definitely some of the most famous people.	From the cut in gimlet media. This is the cut on Tuesdays. I'm your host Molly Fisher.
Text-Only SentencBERT	Yes the main industry obviously and I know that this is not going to be a surprise for anyone is avocado toast. Yeah, and now you see like the repercussions of this of this is welcomed by Wall Street movement and the Fallout of the 2008 financial. If you like what you heard, please rate US and review us and make sure to subscribe to on your pod catcher of choice.	And it's a lot of just it's a it's a lot of just communication and dealing with the dealing with each other and trying to we're trying to create especially with a group of people. There's a lot of you hurry up too late you get up really early and run late to the airport where you get there, but you end up making it early and use weight around or you've got to make it to this show for a load in a certain time, but nothing else is ready. Well, I'll take you around to feel so because because this is this is the The turning point for everything is that you write a song like that and I think to understand that that's always kind of been the goal.	We're going to start with a story from the days before Roe the same arrow and Cynthia's mom had her abortion. This is about fundamental freedoms, when the military starts intervening in women's bodies and actually the campaign in Portugal led to the legalization of abortion. If you want to help you can donate to an abortion fund by visiting the national network of abortion funds their website is abortion funds dot orc you can give directly to the national network or to any of the abortion funds around the country listed on their website.
Multimodal SentenceBERT + 2MLP	We're entitled to kill Industries the world caught us red-handed our plot to conspire and decimate the economy has been foiled by brilliant yet cranky gen xers and Baby Boomers who weep at the thought of Applebee's going out of business in this episode. Yeah, and now you see like the repercussions of this of this is welcomed by Wall Street movement and the Fallout of the 2008 financial. Ask it from SNL was brilliant brilliant depressing commentary on the economic prospects between us and you know the generation of the Boomers, so please watch it on YouTube.	I got to talk with the grammy-winning band Portugal the man whose hit song feel it still went to number four in the Billboard Hot 100 charts and they were really fun to talk to definitely some of the most famous people. It's like we're playing a game of horse and you know, I just made that have court shot and I got to make it a test from the Toyota Corolla. If you have an instant career and want to be featured on the show send us an email and major jobs podcast at gmail.com with your job title and college major if applicable again, thanks for listening and remember.	They had the power you can hear way more about the self helpers on Wendy's show science versus in the episode called the abortion underground. This is about fundamental freedoms, when the military starts intervening in women's bodies and actually the campaign in Portugal led to the legalization of abortion. Mer fund and the science versus team and if you like the show tell some friends to listen to the cut on Tuesdays is a production of gimlet media and the cut.

Figure 4.7: Predicted summary examples from proposed and baseline models