Computational Methods for Assisting Radio Drama Production

Emmanouil Theofanis Chourdakis

A thesis submitted in partial fulfilment of the requirements of the Degree of Doctor of Philosophy

Centre for Digital Music
School of Electronic Engineering and Computer Science
Queen Mary University of London
United Kingdom

November 2020

Statement of Originality

I, Emmanouil Theofanis Chourdakis, confirm that the research included within this thesis is my own work, and that any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

The copyright of this thesis rests with the author and no quotation from it or information derived from it may be published without the prior written consent of the author.

Signature:

Date: 27 Nov, 2020

Abstract

Radio Drama is a theatrical form of art that usually exists solely in the acoustic domain consisting of music, speech, and sound effects and is most often consumed through broadcast radio. This thesis proposes methods for assisting a human creator in producing radio dramas.

Much research has been done to aiding creativity using artificial intelligence techniques in storytelling, music composition, the visual arts, and film. Despite that, radio drama is under-represented in such research. Radio drama consists of both literary aspects, such as plot, story characters, or environments, as well as production aspects, such as speech, music, and sound effects. While plenty of research has been examining each of those aspects individually there is currently no research that combines such studies in the context of radio drama production.

In this thesis, an interdisciplinary approach to assisting a human creator in radio drama production is developed. The task is explored through the joint prism of natural language processing, music information retrieval, and automatic mixing. We show that individual literary aspects of radio drama can be automatically extracted from a story draft provided by a human creator, by using natural language processing methods. Formal rules can be used to express the aforementioned elements in the form of a script able to be read and altered by both the human creator and the computer. We devise recommender systems for sound, music, and audio effects to retrieve the assets required for production. Rules derived from radio drama literature can then use those recorded assets to produce a radio drama mix in a semi-automatic way. Furthermore, an adaptive reverberation effect suggests reverberation settings for each track based on track content and past user choices.

The degree of success for individual tasks in aiding production is demonstrated using examples of radio drama production from raw stories and validated through objective evaluation metrics, and listening tests.

Acknowledgements

Initially, I would like to wholeheartedly thank my supervisor, Prof. Joshua Reiss for his continuous support, supervision, and guidance for the past four years. I would like to extend my thanks to my second supervisor Prof. Ioannis Patras, and independent assessor Dr. Bob L. Sturm who was part of the PhD commitee supervising my progress and giving advice during the yearly stage reports. I would also like to thank Dr. George Fazekas for agreeing to take over as my independent assessor for my last stage report. I would like to also thank the people of RPPtv for sponsoring the first three years of my PhD. I would also like to thank Tim O'Dononghue of Angell Sound for tutoring me for a day on mixing for radio drama. I would also like to thank Matthew Paradis and Matthew Brooks, and my colleagues at the excellent BBC Audio R&D Team for the opportunity to work with them for five months during an internship. I would like to also thank my colleagues, fellow PhD students and C4DM academics for their help answering questions, helping me organize experiments as well as their warm company and pub nights. I would like to especially thank Brecht, Dave & Dave, Tom, Jacob, Rod, Will, Fran, Adan, Saumitra, Delia, Marco, Simin, Siying, Di, Maria, Veronica, for answering questions, and their help in designing and running experiments. Also Giulio for good reason. I would also like to thank Profs. Ioannis Patras, Elaine Chew, and John Schormans for providing me the opportunity to participate in teaching activities. Last but not least, I would like to thank immensely my parents, my brother, and my lifelong partner Dr. Beici Liang for their continuous patience, motivation, and support.

License

Table of Contents

List of Figures

List of Tables

List of Abbreviations

(Bi)LSTM	(Bi-directional) Long Short-term Memory Network
(E)CFG	(Extended) Context-free Grammar
(R)IR	(Room) Impulse Response
AES	Audio Engineering Society
ANN	Artificial Neural Network
APF	All-pass filter
ASG	Automatic Story Generation
ASR	Automatic Speech Recognition
ATMOS	Atmospheric Sound Effects
BBC	British Broadcasting Corporation
BPF	Band-pass filter
CF	Comb filter
CNN	Convolutional Neural Network
CR	Coreference Resolution
CRF	Conditional Random Field
DAFX	Digital Audio Effects
DAW	Digital Audio Workstation
DSP	Digital Signal Processing
EQ	Equalization
FDN	Feedback Delay Network

GMM	Gaussian Mixture Model
GNB	Gaussian Naive Bayes
HMM	Hidden Markov Model
HPF	High-pass filter
HSF	High-shelf filter
HTTP	Hypertext Transport Protocol
IR	Impulse Response
LPF	Low-pass filter
LSF	Low-shelf filter
MLS	Maximum Length Sequence
NF	Notch filter
NLG	Natural Language Generation
NLP	Natural Language Processing
PCM	Pulse Code Modulation
PF	Peak filter
RMS	Root Mean Square
SAFE	Semantic Audio Feature Extraction
SFX	Sound Effects
SM	Studio Manager
SNR	Signal-to-Noise ratio
STFT	Short-time Fourier Transform
STOI	Short Time Intelligibility Index
SVM	Support Vector Machine
TF	Term frequency
TS	Text Simplification
TTS	Text To Speech
WAET	Web Audio Evaluation Tool

List of Symbols

$\mathcal{S}$	A set
$\emptyset$ or $\{\}$	The empty set
$\lvert W \rvert$	The number of elements in W (can be list or set)
p	Precision
r	Recall
f_1	F-1 score
$\rightarrow$	Maps-to (functions) / Produces (e.g. in grammar rules)
$\lvert \mathcal{S} \rvert$	The cardinality of a set $\mathcal{S}$
$W.s$	The “.” operator appends the element s to the list W
$\boldsymbol{v}$	A column vector
$\boldsymbol{M}$	A matrix
z^{-d}	A delay element of d samples in the z-domain
$\lVert \boldsymbol{v} \rVert$	The euclidean norm of vector $\boldsymbol{v}$
μ, σ	Mean and standard deviation
‘‘literal’’	The exact representation of character sequence *literal*
‘‘literal’’i	The representation of character sequence *literal* (case-insensitive)
literal	A regular expression.

Chapter 1

Introduction

1.1 Defining Radio Drama

Before introducing radio drama we must observe how the medium has been referred to historically as well in the modern digital age by theoreticians and practitioners. It is used to communicate drama on radio and differs from simply recording and transmitting blind versions of theatrical plays. The stories in radio drama are developed specifically for the medium of radio, with scenes, characters, events, and emotions augmented using sound effects, music, and clever use of silence; the last three elements being as important as the stories themselves [3, p. 30]. While broadcast to multiple receivers through radio, radio drama heavily relies on the imagination of the individual listener to synthesise the imagery of the play. It is also relatively short in duration (e.g. compared to cinema film) so as not to let go of the interest of the listener. Throughout the rest of this thesis therefore, we define *radio drama* as the theatrical interpretation of a story using speech, music, sound effects, silence, and the imagination of the listener which is consumed via digital or analog broadcast in a relatively short time-frame (usually under an hour).

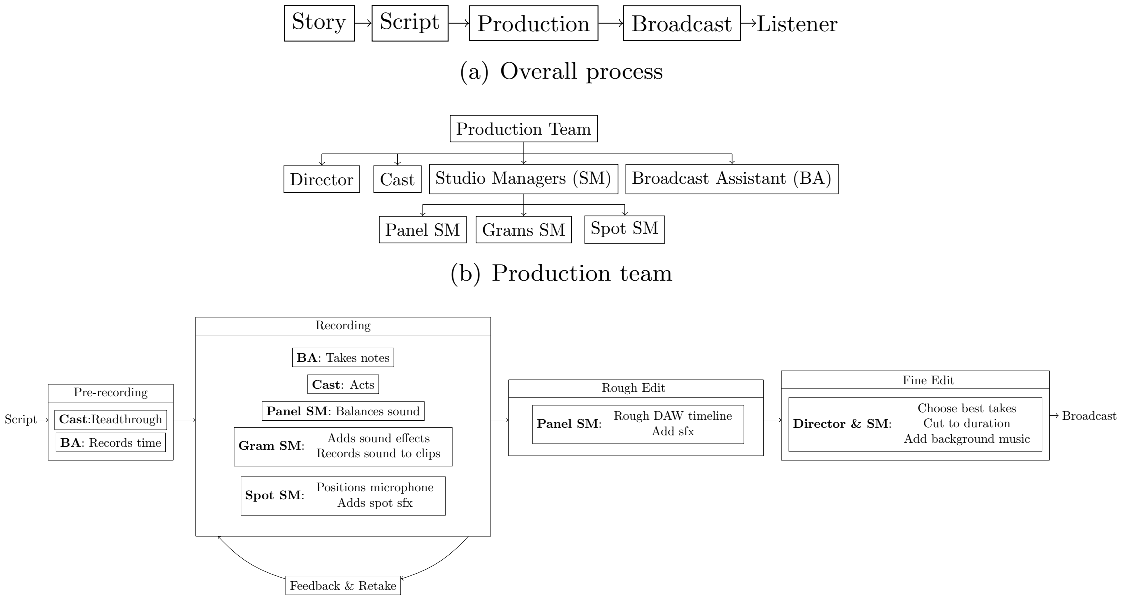
Radio drama has its origins on the *théâtrophone*, a telephone-based system back from 1881 which was used to broadcast sound from theatrical drama to restaurants, hotels, and rich households using a “pay-per-view” system not dissimilar to modern digital distribution approaches [18, p. 15]. This first approach of broadcasting drama however was crude, with problems related to capturing sound, an entertainment form not specifically designed for sound, and of course accessible only to places the telephone network could extend. The last constraint was of no concern with the invention of the radio around the beginning of the 20th century. The first stories through radio can be pointed at broadcasts of children’s stories at the beginning of the century but those were merely readings with no extra sound effects, placing the medium closer to what today we would call *audiobooks*, rather than radio drama adaptations [3, p. 14]. While it is difficult to pinpoint the first radio drama, the first series can be recognised in Eugene Walter’s “The Wolf” (August 1922, New York) where the play was split into 40 minutes episodes, so as to not lose engagement with the audience [19]. This duration was close to the 30 minutes used later in the Golden Age of radio drama. [3, p. 18]. In Britain, the first radio drama was Richard Hughes’ “A Comedy of Danger” in January 1924 [3, p. 16]. While those reminisced of radio drama as we know it, the term was not used to describe them until at least the 1950s [3, p. 33].

Radio drama has gained a place as a technically and culturally significant form, mainly due to radio. The first radios were bulky devices that relied on a steady electricity source and only wealthy households could afford them. This changed with the invention of the crystal radio in the 1920s and also in the 1950s with the transistor-based battery-powered radio. The little dependence of the medium on electricity allowed people with no access to steady power, such as people in developing nations or in war, access to news, education, and entertainment [3, p. 8]. This affordability of the medium continues in the digital world. While access to the internet requires much more complicated hardware than simply a crystal and a copper wire, a low-bandwidth connection of under 1kbps is required to transmit intelligible speech [20]. While this is still an increase in complexity and cost requirements from the point of the consumer, production of radio material has become much cheaper since the only requirements are a good mono microphone, a cheap computer, an internet connection, and a free DAW application. This is in contrast to the expensive hardware airwave radio transmission requires. Radio is also affordable in time: contrary to other kinds of entertainment, like theatre, cinema film, or even novels, does not demand our full dedication but allows us to perform other activities while listening to it [21].

Radio drama's impact on culture does not rely solely on radio. The infamous CBS' *War of the Worlds*, through its use of first-person narrative, ingenious mixing, and spot effects led to societal panics about a Martian invasion with extended consequences that even included human casualties [18, p. 111]. Echoes of that play resonate until today, the show and the subsequent panic having become part of the western pop culture [3, p. 27]. In World War 2, radio drama served as a substitute for theatre-going during the blackouts of London [3, p. 33] but did not just lend itself just to entertainment; it played a major role in the efforts of the United States to increase the faith of its people in the Allies [19].

1.2 Radio drama as a Visual Medium

One of the first misconceptions that need to be clarified, is that of a radio drama as a blind medium. It can be argued that *radio drama* as a term, even more as a drama form, is a contradiction [21, p. 26]. After all, the word *theatre* comes from the Greek word *theomai*, which means “to see”, “to behold”. *How can one see when all that is available is sound?* To answer this question we need to understand the importance of sound, not only in radio but also in film, and theatre. Hand and Traynor [3, p. 3] propose a simple test: to watch any horror film with muted sound; the film is transformed from a scary horror experience to a succession of images. Sound is much more than the audible artefact of what is happening in the scene, it is an evoking of the viewer's or listener's *mind's eye* or *imaginative spectacle* [18, 22].



(a) Overall process

(b) Production team

(c) Production process

Figure 1.1: Radio drama pipeline. The team organisation shown above is the one used at the production of BBC radio dramas and is described in [1].

1.3 The Radio Drama Pipeline

To provide methods for assisting radio drama production we must first identify the processes taken in producing drama. We defined radio drama as the interpretation of a story using music, sound effects, silence, and the imagination of the listener (Section 1.1). We can think of the form as a pipeline with the beginning being a story and the end node being the listener as seen in Figure 1.1. At first the *story* is adapted to a *script*, then *production* takes place and results in audio *broadcast* to a group of *listeners* [3, p. 34]. The imagination of the listener is a vital part of radio drama so we begin by identifying who the listener is: we would make a different adaptation to a children's audience than an adult one. A next step would be to adapt the story to a script suitable for radio production. Part of this step is to identify elements of radio drama that can be used to convey to the listener the various story parts such as characters, locations, and emotions [23]. Such elements might be assets, such as music or sound effects, or special effects such as panning, fading, and reverberation. The production stage usually employs a small production team, consisting of a director, actors, and studio managers

(SM). An example of a production team is described in [1] and can be seen in Figure 1.1(b). The roles of the production team are summarised below:

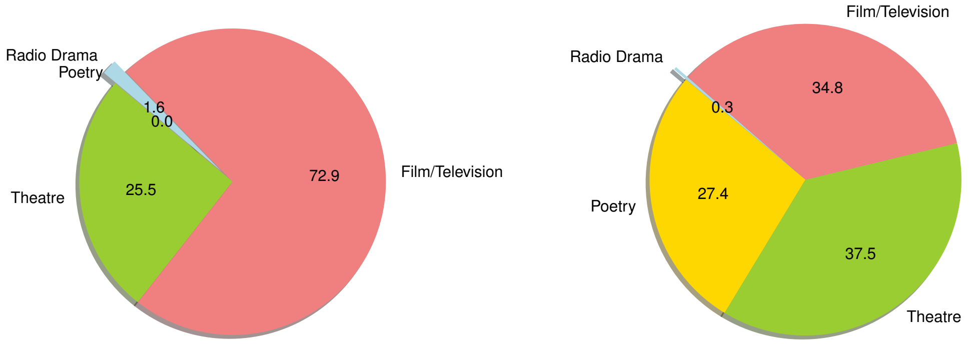
- *The Director* is the 'leader' of the drama. They are the ones whose vision the radio drama interpretation takes.
- *The Cast* are the actors that perform the roles of the characters in a drama.
- *The Studio Managers* (SMs) manage recordings and relevant assets. According to [1], radio drama production for the BBC uses a Panel SM that balances sound, a Grams SM that adds *pre-recorded* sound effects and manages recordings, and a Spot SM who creates *spot* sound effects.

A summary is also given about the various production steps:

- **Pre-recording** – A simple reading of the script is done by the cast to get an estimate of the duration of the drama.
- **Recording** – The actors play their role. The Gram SM adds sound effects and stores the recording to disk. The Spot SM creates perspective by moving the microphone in the studio and also adds spot sound effects. The panel SM balances sounds. The BA adds keeps detailed notes about the performances and adds relevant recommendations to the script. The recording process is repeated until the team is satisfied.
- **Rough Edit** – A studio manager creates a rough DAW timeline to identify the best takes and process the mix, in order to have a consistent sound level throughout.
- **Fine Edit** – The Director and the Sound Managers cut the radio drama to the required length, choose the best 'takes' and add background music.

The above processes can take up to several weeks and require experienced staff to fulfil the roles discussed above [1]. An important goal of this thesis is to devise of methods that allow the above processes to be sped up, and also be performed by people that are interested in radio drama generation but might be lacking resources or expertise.

1.4 Why Radio Drama?



(a) Audio Engineering Society

(b) Semantic Scholar

Figure 1.2: Percentages of publications with each term in their Title or Abstracts computer science literature stores as of August 7, 2019.

Until the time [18, 24] were released, radio drama as a research subject had been largely ignored by academic researchers. Contrary to the expectations of the authors, radio drama as a literary medium seems to continue to be overlooked well into the second decade of the millennium [25]. While the trend reported refers to the domain of critical studies, similar figures can be seen when considering research from technical communities, such as the *Audio Engineering Society* (AES). AES, given its relevance to sound research, was expected to host plenty of relevant material. Searching its e-library[1] in 2019 we find only 4 articles which include the term *radio drama* or *radio play* in their titles and abstracts, compared to 63 for *theatre* and 180 for *film* or *television* placing radio drama research to a 1.6% among these topics. Searching in the broader computer science community we arrive at a dimmer result. The *Semantic Scholar*[2] website, which provides results from 20 websites with scientific papers, gives only 169 results for *radio drama* or *radio play* compared to 21,300 of the *theater* or the 19,800 of *film* or *television* placing publications of radio drama below 0.5% of the four literary arts. Figure 1.2 gives a visual

[1] `http://www.aes.org/e-lib/`

[2] `http://semanticscholar.org`

perspective of those ratios.

Even in the cases that *radio drama* turns up into academic works, most of them relate mostly to research about the acoustics of the radio medium. From the 4 articles of the AES e-library, 3 of them related to producing and transmitting multi-channel/surround sound and one from 1995 about improving the production process and even then, limited to the final audio mixing process. Radio drama production however is not just sound mixing, and as we discuss in Chapter 2, it cannot be seen as simply the blind cousin of drama. If seen as drama, it encompasses plot writing, character development and dialogue but if seen as a production for radio, constraints about time management, music and sound effect libraries, actors, and others must be taken into account. Those two perspectives are intertwined and must be tackled together, for example the availability of a sound effects library will directly affect writing of the script.

All the aforementioned aspects have been a subject of research on automating or assisting them. As examples, research on automating novel writing goes back to the 1970s [26], and recommender systems for assets with or without accompanying metadata is an active area of research since at least 1992 and are an active research subject [27, 28]. Methods for automatic mixing have existed since the 1970s as well and are also the subject of active research, however almost exclusively in the context of music [29, 30]. None of the research mentioned above has been used in the context of producing radio drama. Huwiler [23] argues that radio pieces must not be analysed as literature since they are written fundamentally different, even when adapted from a previous literary work. For example, writing or analysing narrative for radio drama tends to be different than for stories meant for reading. Narrative for radio drama tends to have simplified structure, with few characters and descriptive language, but a specific story progression focused on being able to capture and hold the attention of the listener. In the rest of the thesis we examine the different aspects of radio drama in the context of radio drama production: from expressing the plot of the drama as a script to mixing for radio.

1.5 Research Assumptions

As discussed in the previous section, producing radio drama is a huge multi-faceted endeavour and there is a need for realistic assumptions to be made for the various aspects to be examined to a reasonable depth. Initially, we assume that stories to be adapted have already been written and the thesis will examine only how to adapt them to radio drama. The direct consequence of this shift is that we try to extract information that is already available in a story, contrary to trying to generate it. The second assumption to be made is that the stories examined are sufficiently short. Golden age radio dramas tend to be 30 minutes in length [3], however we will keep our dramas well below that duration in order to facilitate evaluation of our methods. In the next section we present our research questions and directly link them to the chapters that answer them.

1.6 Research Questions

The main question this thesis explores is the following:

> **In what ways can advances in artificial intelligence and machine learning assist a creator when producing radio drama?**

Our goal is not to answer this question by exhaustively enumerating all such ways, but to identify areas that can directly benefit from recent advances and provide directions towards further research on this neglected field. Below we provide the individual questions answered throughout the text, together with references to the chapters and sections answering them.

1.6.1 Understanding Radio Drama

In what elements can radio drama be deconstructed in order to make computational analysis possible? Chapter 2 explores how radio drama literature has studied the elements of radio drama when examined as a story, as well as an acoustic art form and we seek to construct a taxonomy to facilitate further research.

1.6.2 Information Extraction in Stories

To what extent do recent advances in automatic story generation and natural language processing (NLP) allow us to extract meaningful information from a story expressed in raw text? Can extracting such information be seen as a set of NLP tasks common in the literature? Can we use or devise algorithms to extract information that is either explicit or implicit? In what ways can external knowledge, i.e. knowledge elicited from online ontologies, help? How do we test the extent to which the aforementioned tasks are successful? The main objectives for answering these questions are:

1. To establish a domain of literary elements found in radio dramas that allow for adaptation in dramaturgical speech, music, sound effects. We aim to identify which parts of the story text contain useful information in the context of radio drama production, and what part of the text can be also expressed in a dramatic way or only as narrative.

2. To acquire an understanding of the parts of the story that can be directly derived from the text. To identify those parts that can be understood from the context.

3. To establish links between well-understood problems in NLP and extracting and inferring information from story text that can be used in radio drama production.

Answering the above questions provide us with information extraction methods for story-to-radio drama adaptation. Those are discussed in detail in Chapter 3.

1.6.3 Controlling reverberation based on desired Reverberation Characteristics

In the process of mixing there are cases where the mixing engineer wants to ascribe the characteristic of a space to an audio track (e.g. applying reverb to a footsteps track to give an impression a character is walking on a large hall). Such characteristics include e.g. the time it takes for the sound to decay. How can we adapt a reverberation effect to be able to apply reverberation based on the aforementioned characteristics? For example, allow the mixing engineer to convey the characteristics of a cathedral, with a long reverberation time, should they choose to do so. The main objectives of this question are:

1. To identify appropriate reverb effect architectures that allow answering this question.

2. To understand whether such architectures can be parameterised accordingly to allow a user to convey the characteristics of a space to a radio drama track, based on perceptual characteristics such as reverberation time, echo density, and clarity.

The question is explored in Chapter 4.

1.6.4 Asset Retrieval for Radio Drama Production

Given a radio drama script, how can assets for production be retrieved in an automatic way? How can we retrieve audio effect parameters as well? The objectives in this case are the following:

1. To establish a connection between recommendation systems for music, speech, and sound effects and asset recommendation for radio drama.

2. To devise methods for recommending effect parameter settings for loudness, panning, fading, and reverberation.

And the outcomes, as described in Chapters 5 and 6 are the following:

1. Methods for organising and retrieving assets (speech, music, sound effects) mentioned in the radio drama script.

2. Methods for recommending audio effect settings to aid the user when mixing radio drama.

1.6.5 Radio Drama Production from Script and Asset Libraries

Given a machine-readable radio drama script and a library of assets, how can we produce a radio drama? The main objective here is:

- To provide methods for aiding the user in mixing and mastering a radio drama.

1.7 Contributions

While pursuing answers to the above questions, the following contributions were made via peer-reviewed publications:

- E.T. Chourdakis and J.D. Reiss – "Automatic Control of a digital reverberation effect using hybrid models". Presented at the *60th International AES Conference on Dereverberation and Reverberation of Audio, Music, and Speech* in February, 2016 (Leuven, Belgium).
- E.T. Chourdakis and J.D. Reiss – " A Machine-Learning Approach to Application of Intelligent Artificial Reverberation". Published as an article in the 65th Volume (Issue 1) of the *Journal of the Audio Engineering Society* (JAES) in February, 2017.

The two above papers discuss an approach to automatically applying presets for the effect of reverberation to a source audio file, given the content of that audio file. The two papers describe the same method, with the sole difference being that the second paper allows the preset to be controlled using perceptual characteristics of the Impulse Response. The two main contributions of the above are:

1. They provide a type of retrieval system for reverberation effect parameters where the 'queries' are not given in a text form, but rather as the sound content of the audio file the effect will be applied to.
2. They provide a way to apply reverberation by directly choosing intuitive reverber-

ation characteristics: reverberation time, echo density, central time, clarity, and spectral centroid.

They are discussed in more detail in Chapter 6.

- E.T. Chourdakis and J.D. Reiss – “From my Pen to your Ears: Automatic Production of Radio Plays from Unstructured Story Text” Presented at the *15th Sound and Music Computing Conference* in July, 2018 (Limassol, Cyprus).

This paper describes an approach at an automated story-to-radio drama system. This thesis is centred heavily around sections of this paper. Its main contribution is that it provides an initial approach at analysing a story, retrieving relevant sound effects as well as reverberation and panning settings, and composing a rough radio drama take automatically. Chapter 3 extends and improves the methods of Section 3 of that paper as well as repeats the evaluation presented in Section 5. Chapters 5, and 7 improve and extend Section 4 of the same paper. A formal grammar of the radio drama script used in that paper is also provided in Appendix C.

- E.T. Chourdakis and J.D. Reiss – “Grammar Informed Sound Effect Retrieval for Soundscape Generation”. Presented as a poster at the *13th Workshop of the Digital Music Research Network* in December, 2018 (London, UK).

The above paper discusses a simple yet intuitive method for retrieving assets for generating a soundscape given a sentence in natural language. Its main contribution is a method for constructing queries from literary story sentences, that allow retrieval of relevant sound effects for that story from an online library of such effects. The paper is expanded and discussed in more detail in Chapter 5.

- E.T. Chourdakis and J.D. Reiss – “Tagging and retrieval of room impulse responses using semantic word vectors and perceptual measures of reverberation”. Presented during a lecture at the *146th AES Convention* in March, 2019 (Dublin, Ireland).

This paper describes a method for retrieving reverberation impulse responses from –

possibly noisy– labels. The contributions in this paper are:

1. A content-based retrieval system for Room Impulse responses that retrieves them using only five perceptual measures instead of features derived from the frames of the audio files, which is common in content-based audio retrieval.

2. The ability to use imprecise tags to label or retrieve such files, without losing much in retrieval accuracy.

Parts of it are discussed in more detail in Chapter 5.

1.8 Thesis Structure

The rest of the document is organised as follows: **Chapter 2** discusses radio drama as a format of storytelling medium. The chapter discusses elements that define radio drama as a story, the elements used in expressing it as sound and the relationship between the two. **Chapter 3** discusses methods for extracting literary elements of radio drama that can be expressed using sound. **Chapter 4** focuses specifically on the audio effect of reverberation since it is used extensively in the methods discussed in Section 5.3 and Chapter 6. **Chapter 5** discusses methods for retrieving assets that are relevant to the source story and can be used in radio drama production. **Chapter 6** focuses specifically on retrieving relevant reverberation parameters for sound files based on content. **Chapter 7** discusses how content extracted and generated throughout the rest of the thesis can be mixed to radio drama. Finally, **Chapter 8** concludes this thesis by discussing limitations and possible future directions.

Chapter 2

Story and Discourse in Radio Drama

2.1 Introduction

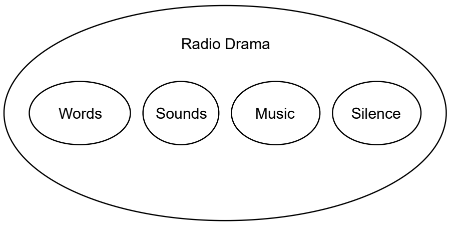


Figure 2.1: Constituents of radio drama

In this chapter, we discuss important elements found in radio drama stories and how they are communicated to the listener. We try to provide formal definitions for important elements in radio drama story and discourse, as well as a formal framework to use them computationally in the rest of the thesis.

Stories made for radio drama are not very dissimilar to stories found in the literature, it is usually the case that radio drama adaptations are made from popular stories written in books. It is, however, important to acknowledge that radio as the story's medium has an influence on the adapted story [23]. The perceived 'blindness' of the medium imposes several limitations. First of all, radio drama frees the listener's vision and allows them to do other activities while listening to it. Therefore, radio drama 'competes' with other activities for the attention of the listener [3, p. 36]. Another constraint comes from the limited capability of radio to communicate ideas and factual information; the word in radio is 'fleeting' and information can be missed by the listener [31]. This constrains radio drama in the way it is written: it must capture early the attention of the listener and provide the story in an easy to follow way. Radio dramas tend to have simple narrative, use descriptive language, and involve few characters. They are also 'to the point': every event or interaction between characters has a reason for being in the drama.

Like every story, radio drama stories are focused around important narratological elements, such as the goal of the main character. Understanding every important element however requires an exhaustive referral into semiotics, something that escapes the scope of this thesis. Since many radio dramas come from adaptations from popular stories, we will assume a story already exists and we need only concern ourselves with adapting it to radio drama. There have been previous approaches at formalising radio drama adaptations. In [23] the author discusses how the various acoustic elements of radio are used to signify elements in the underlying story, and how this signification might differ for each individual drama. While this last fact may seem to discourage a computational framework, we will try to formulate one based on the most directly defined such connections between story and acoustical elements while acknowledging that the aspiring radio drama creator might not completely adhere to such rules.

The basic elements available for radio drama discourse are: *Speech*, *Sounds*, *Music*, and *Silence* (Figure 2.1) [3, p. 40]. In the sections below we briefly discuss the use for each. It is important to note that it is not necessary for an audio drama to include all of the elements. Although usually, we expect to see a little bit of every element, extreme cases exist which completely lack an element. The audiobook format can be thought of radio drama that completely lacks sound effects or music and makes up with increased use of spoken words. On the other end of the spectrum, we have what [3, p. 59] refers

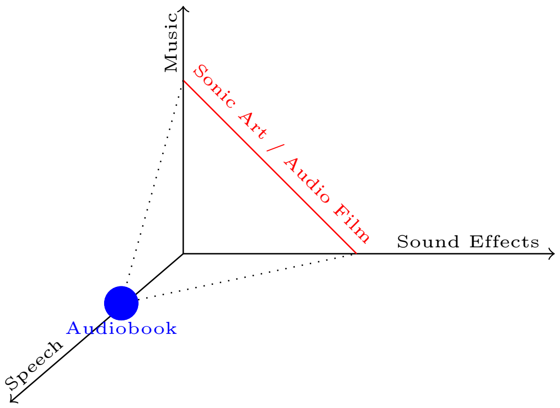


Figure 2.2: Acoustic Dimensions of audio drama. The beginning of an axis signifies lack of an element and the vertices of the triangle that the format consists fully of that element. The blue dot refers to the format of Audio Book, which usually lacks music and sound effects. The red line refers to a type of audio drama with no speech. Silence is not mentioned since it is an integral part of all audio dramas.

to as ‘sonic art’ and [32, 33] as ‘audio film’: audio drama with a complete lack of speech where the whole story is communicated through the exclusive use of sound effects, music, and silence.

The three elements above are signs of the following narratological elements: *Scenes*, *Events*, *(Discourse) Time*, *Tension*, *Emotion*, *Characters*, *Objects*, *Locations*. Obviously there exist many more but we identified those to be those that can be directly communicated with acoustical elements.

In the following sections we distinguish between *story*, and *discourse*, and provide a formal definition for each. We make this distinction for stories written for books as well for stories written for radio. This will help us to clarify the differences between stories in books and stories for radio, and make the distinction between story and discourse clearer. Clarifying those differences will aid us at a later stage when formulating rules for adapting literary stories to radio drama in Chapter C. Finally, we provide a taxonomy of story and discourse where we place the narratological and acoustical elements we mentioned above.

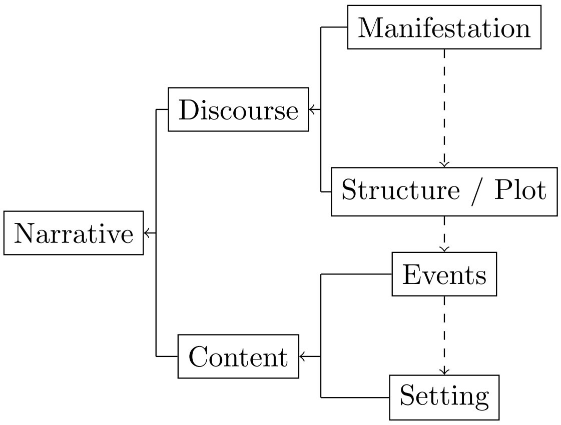


Figure 2.3: Hierarchy of narrative elements adapted from [2]. The arrows denote hierarchical structure and the dashed lines dependency: the higher levels depend on the lower levels.

2.2 *Content* and *Discourse*

Radio drama is an art form that is used to tell stories. Therefore, it has narratological structure. In Figure 2.3 we borrow the hierarchy presented in [2], which facilitates computational analysis. In the basis of the drama is the story, arguably the most important element of radio drama [3, p. 106]. It is the foundation where all other elements of radio drama can stand on. However, the story is hidden from the listener. It is revealed gradually through *discourse*. While discourse in literary stories uses text to communicate to the reader, in radio drama communication it uses the audio elements discussed in Section 2.1. We use the term *manifestation* to refer to the different types of communication of discourse to the reader/listener such as natural text or sound. The elements of the story that happen 'behind the scenes' in the story belong to the story *content* [34, p. 26]. Content can be distinguished into the *story world* which includes all entities and relations between them in the narrative, and a set of chronologically organised *events*. Expressed differently, the story content is the answer to the question *"what happens in the story?"* and discourse is the answer to the question *"how does the reader/listener know?"* [34, p. 20]. Below we discuss in more details the 'leaves' of the narrative hierarchy tree in Figure 2.3.

2.2.1 Discourse *Manifestation* and *Structure*

Discourse answers to the question *"how does the receptor of the story know?"*. Observing the story the answer comes easily: for written stories by reading the text and for radio drama by listening to the story in sound. We say that the story *manifests* itself as text or sound respectively. As an example of how the same story manifests in story and text, consider a text excerpt of the Aesop's fable, *The fox and the crow*:

> "A crow was sitting on a branch of a tree with a piece of cheese in her beak when a fox observed her and set his wits to discover some way of getting the cheese."

When adapting *the crow and the fox* to a radio drama script, we observe a much longer discourse segment:

MUSIC: *Happy medieval music*

Narrator: The fox... and the crow

– First Scene –

ATMOS: *Birds chirping, leaves rusting*

(long pause)

Narrator: One morning a fox was walking through the woods, looking for something tasty to eat for his breakfast when his nose picked up a scent... a scent of something very interesting. He stood still... and sniffed the air.

SFX: *(Fox sniffing)*

(pause)

Fox: Cheeeese...

Narrator: He said...

Fox: *(Emphatically)* I smell cheeese... Now why would there be cheeese in the middle of the wood like this?

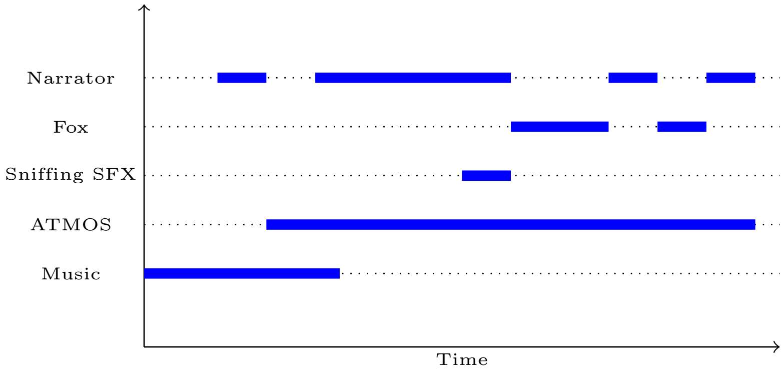


Figure 2.4: A DAW timeline picture of the first scene of the for the BBC School Radio's *"The fox and the crow"*. Horizontal axis denotes time. The depiction of events is not accurate in time but symbolic.

Narrator: The fox didn't have to wait long to find out... because sitting
on a branch... high up in a tree... sat a crow... and in the crow's beak...
was the biggest piece of cheese he'd ever seen.
(short pause)
Now the fox loved cheese more than anything in the whole world and he decided
that come what may he would have that piece of cheese for himself.

The above script[1] when produced will have a DAW timeline picture similar to the one in Figure 2.4. Both the radio drama script and the DAW timeline are considered valid manifestations of radio drama narrative.

We notice several obvious differences between the two forms. The most obvious one is that manifestation in radio drama uses the sound elements we briefly discussed in Section 2.1: *Speech*, *Sound Effects*, and *Music*. The second is the differences in length: the text excerpt is much shorter. Radio drama tends to be a form sparse in information such as not to lose the attention of the listener. Consider the following part:

> "A crow was sitting on a branch of a tree with a piece of cheese in her beak when [...]"

In the same sentence we get the information that the crow was sitting on a branch, which

[1] BBC School Radio's Aesop Fables 1–4: `https://www.bbc.co.uk/programmes/b03g6vqg`

was on the tree, that the crow had some cheese in her mouth. This is three pieces of information in a single sentence. In the radio drama manifestation, we get the following instead.

> “[...] on a branch... high up in a tree... sat a crow... and in the crow’s beak... was the biggest piece of cheese he’d ever seen [...]”

Here simple pieces of information are segmented by short pauses making information much easier to absorb. Another subtle difference that can be seen is the use of timing. In radio, as opposed to written stories, the listener cannot ‘listen forward’ and absorbs information in a much slower rate. This allows the radio drama creator to ‘play’ with time and induce effects such as *suspense*, as evident in the excerpt above: the narrator delays the introduction of an important character of the story, the *crow*, by using pauses and slowly revealing the location *someone* was sitting on before introducing who that was. A final difference between manifestation of the two forms can be observed in the elongation of the word ‘cheese’ which the creator does to introduce *humour*.

We mentioned that radio drama is a sparse medium regarding information. To observe this statement more clearly we need to discuss about *discourse structure* or *plot*. We refer to plot as the arrangement of story events as presented to the reader/listener [34, p. 43]. In its written story form the text excerpt we presented from *the fox and the crow* is presented to the reader with the following structure:

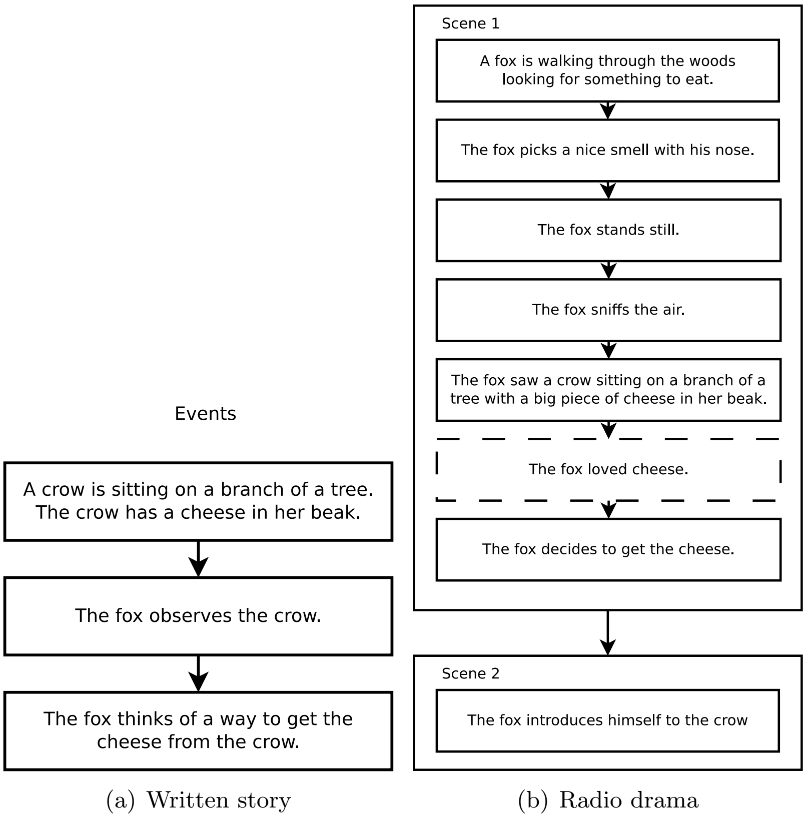
1. A crow is sitting on a branch of a tree. The crow has a cheese in her beak.

2. A fox observes the crow.

3. The fox thinks of a way to get the cheese from the crow.

Another way to represent the progression of those events can be seen in Figure 2.5(a). We can similarly represent the plot as given in the radio drama script:

1. A fox is walking through the woods looking for something to eat.


Events
A crow is sitting on a branch of a tree. The crow has a cheese in her beak.
The fox observes the crow.
The fox thinks of a way to get the cheese from the crow.
Scene 1
A fox is walking through the woods looking for something to eat.
The fox picks a nice smell with his nose.
The fox stands still.
The fox sniffs the air.
The fox saw a crow sitting on a branch of a tree with a big piece of cheese in her beak.
The fox loved cheese.
The fox decides to get the cheese.
Scene 2
The fox introduces himself to the crow

(a) Written story (b) Radio drama

Figure 2.5: A graphical representation of the progression of discourse. Boxes with solid outlines represent events and boxes with dashed line reveal information.

2. The fox picks a nice smell with his nose.

3. The fox stands still.

4. The fox sniffs the air.

5. The fox saw a crow sitting on a branch of a tree with a big piece of cheese in her beak.

6. (Reveal Information) The fox loved cheese.

7. The fox decides to get the cheese.

8. The fox introduces himself to the crow.

We observe that the structure is similar (although re-ordered and expanded) to the

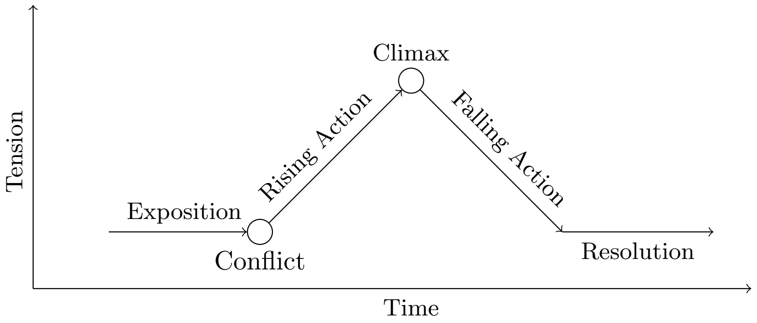


Figure 2.6: *Freytag's Pyramid* for plot structure. Lines represent sequences of events or scenes and hollow circles important single scenes. Vertical axis measures tension and horizontal axis discourse time. The plot begins with introducing the characters and environment (*Exposition*) that lead to a point of *conflict*, then events of raising tension lead to the *climax*, which is subsequently followed by events that ease the tension (*Falling action*) and finally, the conflict is *resolved*.

same story told in writing, although the same story resulted in a much longer discourse for radio drama. It is usually the case that radio drama adaptations are much more expansive in providing information compared to their source stories. Finally, for radio drama, we group events and information revealed to the listener into *scenes* which group together those events that take place at the same place and point in time [35]. In Figure 2.5(b) we notice that we have grouped discourse into two scenes: one of the fox walking by himself and revealing his desire to eat the cheese, and another talking to the crow. Scenes are further grouped into main parts, or *acts*.

A common drama structure follows the *Freytag's Pyramid* [36, p. 115] which divides the discourse structure into five parts: *Exposition*, *Rising Action*, *Falling Action*, and *Resolution*. Exposition is the part that introduces the characters and the environment, and usually leads to a *conflict*. Conflict is the point in the plot where the objectives for the story's characters are placed against each other. Rising Action is the sequence of events or scenes that build tension in the plot and lead to a point of *climax*. Finally, Climax is the highest point of tension in the story. It is usually a single event which seals the fate of the characters in the conflict. Finally, Falling Action is the part that follows the *climax*. Since the fate of the characters is sealed by the climax, tension finally eases in the *falling action*.

Tension is either communicated through narrative, or through dialogue and actions of characters. A very effective way to build tension is to use long pauses between words in dialogue or sound effects. Tension, however, is not the only emotional state the listener experiences. During the drama, the listener is expected to develop an emotional connection with the character which might be love, hate, admiration or even pity [18, p. 171]. A particular emotion can be evoked to the listener either by narration, sound effects, music, or dialogue.

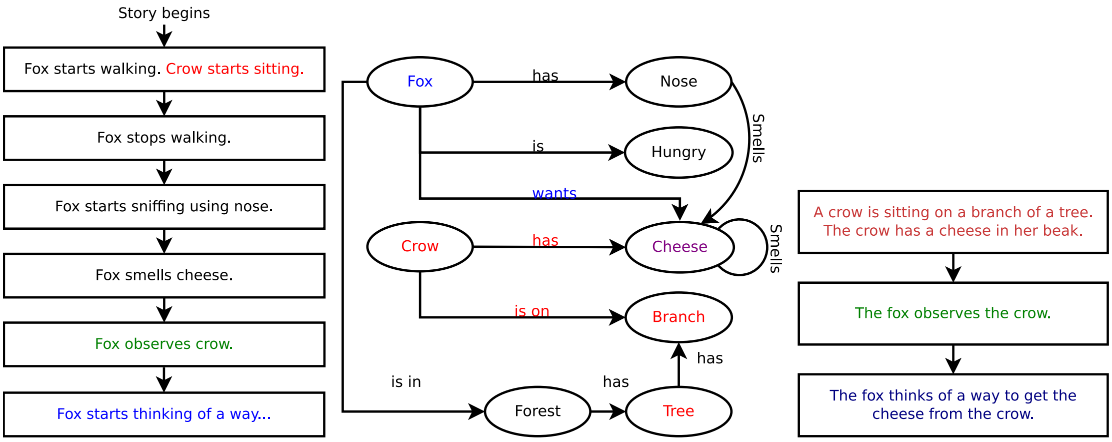
A feeling prevalent in drama which is related to tension is *suspense*. Suspense as a feeling has multiple definitions throughout the literature. Since this thesis looks at radio drama through the prism of a computational study however we will use the definition from [37] according to which, the reader feels suspense when the paths that solve the character's problem have been restricted. We adopt this definition to be consistent with previous studies of computational modelling of narrative [38, p. 12].

2.2.2 Story *events* and *setting*

In section 2.2.1 we saw that discourse reveals story information to the reader/listener. This information originates from the story *content*. The content can be decomposed to the story *events* and the story *setting* (Figure 2.3). As story *setting*, we refer to the state of the story world: objects, locations, characters and their relationships, their emotional states, etc. The elements of *story setting* get organised into the *story events* that are used to tell the story. This organisation is usually linear (events in the story world usually happen in linear time).

We visualise the differences between the *setting* and *events* in Figure 2.7. Figure 2.7(a) shows a representation of story events and Figure 2.7(b) of *story setting*. The discourse provided in Figure 2.7(c) is the same as the one in Figure 2.5(a) and is derived by picking appropriate information from the story content at each step of discourse. Note that in a similar fashion, the discourse provided in 2.5(b) is derived from the same content.

The representations chosen for discourse and content in the figures above were chosen


Story begins
Fox starts walking. Crow starts sitting.
Fox stops walking.
Fox starts sniffing using nose.
Fox smells cheese.
Fox observes crow.
Fox starts thinking of a way...
Fox
has
Nose
Smells
is
Hungry
wants
Crow
has
Cheese
Smells
is on
Branch
has
is in
has
Forest
Tree
A crow is sitting on a branch of a tree. The crow has a cheese in her beak.
The fox observes the crow.
The fox thinks of a way to get the cheese from the crow.

(a) Story events (b) Story setting (c) Discourse

Figure 2.7: A representation of story events and the story setting. Events are represented in simplified clauses. In the *setting* representation arrows represent binary relations. For example the arrow from the *crow* node to the *cheese* node labelled *has* means that the crow has the cheese. The rightmost figure represents an example discourse derived from the *story events* and *story setting*. Colours represent information used to provide each piece of discourse.

to facilitate comprehension and are not unique. There are plenty of representations to choose from for representing discourse structure and choice depends on the type of analysis needed. Examples of representation include simplified natural language [39], graph-based [2, 40, 41], structured [42] and meaning-based representations [43–46]. We discuss them further in Chapter 3.

We observe that the story setting (Figure 2.7(b)) consists of entities easily identified in the world, such as *cheese*, or *tree* or emotional states such as *hungry*. The Paris School of semiotics refers to those entities as *Figures* [47, p. 12]. Below we discuss the ones used most often in radio drama.

2.2.3 *Characters*

We refer to animate actors in the story as *characters*. Characters can be communicated to the listener by using either narration, dialogue, or sound effects. The simplest way is for them to be described by the Narrator or by them to speak in the story. They can also be hinted in the lines of other characters (e.g. by introducing them or speaking directly to them), or even by using plain sound effects (e.g. in the case of *The Revenge* the main character does not speak but is perceived by the sounds resulting from their interaction with the story world). In radio drama the characters generally are prepared using the following attributes [3, p. 184]: *Age*, *Gender*, *Sexual Orientation*, *Ethnicity*, *Upbringing*, *Political Affiliation*, *Personality Traits*, *Appearance*, *Appearance*, *Occupation*, and *Lifestyle.* Not all of them are needed or exposed in the drama. For example, in *"The fox and the crow"* the makers only use the age, gender, and personality traits. The fox is male and the crow female. The fox is young and cheeky and the crow older and vain. Such attributes have not just been introduced for the sake of making the story more interesting, for example, the crow's vanity is the reason why she finally loses the cheese to the fox. Characters can be categorised into *the main character* and *secondary characters.* As the main character, we refer to the character around whom the plot revolves. Secondary characters on the other hand are those whose role is to aid the plot to unfold through their interaction with the main character. In *"The fox and the crow"*, the main character is the *fox*, since the plot revolves around him getting the cheese. The main character usually provides the *point of listening* we will discuss in Section 2.6 and is placed above all other characters in the sound hierarchy [18, p. 163]. The role of the main character has been studied in the past in literature and is referred to as the *hero/heroine* [48]. A very important role for the main character is for the listener to be able to sympathise with them so that tension can be built through their struggle. Their aspirations must be strong and the listener must be able to develop emotions for them [18, p. 172]. In "The fox and the crow", the *crow* is a secondary character since the plot revolves around her conflict with the fox. Secondary characters are usually placed lower than the main character in the sound hierarchy (Section 2.6). They are defined by their role in interacting with the main characters as *companion*, *adversary*, etc [48]. A significant difference between radio drama and written stories is that in radio drama the number of characters tends to be kept to the bare minimum, every encounter between characters, should be significant [3, p. 109].

2.2.4 *Inanimate Objects*

As *inanimate objects* or simply *objects* we refer to those objects in the story world that may or may not produce sound, and thus be communicated using SFX, but do not have agency. They can also be used as *instruments* by characters in which case they produce sound only when the characters interact with them. In the example of *"The fox and the crow"* the *fox* uses his *nose* to smell the cheese, therefore producing a 'sniffing' sound. Examples of objects can be wall clocks ticking, weapons firing, etc.

2.2.5 *Locations*

As *location* we refer to the perceived physical space parts, or the whole of the story takes place. In our example, the location is *the forest.* Locations are communicated to the listener with the use of *sound effects*, *audio backdrop*, *emanation speech*, or referred to in *dialogue* or in *narration.* Locations can also alter the perspective in a drama, by physically restricting movement of characters in the scene (e.g. in a car). An example would be *"wind blowing, bees buzzing and the bell of a church"* when referring to a rural area.

2.3 Elements of Radio Drama Discourse

The elements available and are specific to radio drama discourse can be mostly categorised as elements of discourse manifestation. Those are the following: *Speech*, *Music*, *Sounds* and *Silence.* The only element we identified that can be classified as a structural element of discourse is *Discourse Time* which we will also discuss.

2.4 Speech

Speech is used in radio drama to communicate *information*, express *emotions*, or *inner thoughts* [3, p. 41]. It can be categorised into *Dialogue*, *Narration*, or *Emanation* speech (Figure 2.8).

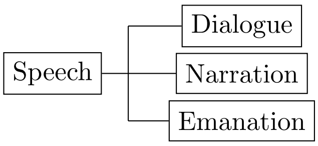


Figure 2.8: Types of speech used in radio drama

2.4.1 Dialogue

Dialogue between radio drama characters is similar to real-life conversation. They, however, must respond to what is happening in the drama [18, p. 188]. Successful dialogue uses active, direct, and emotional language as opposed to neutral communication. It also springs from the characters' background and emotional state. Dialogue is realised using the voices of actors. The voices of characters can be described using characteristics such as Volume, Pitch, Pace, Accent, etc.[3, p. 175]. Changes in those characteristics can be a sign of the characters' age, gender, or social and regional background, but also of the character's emotional or physical state [23]. As an example, male voices tend to be deeper (have lower pitch) than female, which in turn are deeper than children voices. Older characters tend to also have deeper and mellower voices. Physical or emotional state tend to also affect the characteristics of a characters' voice by e.g. increasing talking pace when the character is stressed [3, p. 174].

2.4.2 Narration

Narration is used to evoke visual imagery of parts of the drama that would appear on screen if it was on cinema [18, p. 81]. It is usual that narration happens using the voice of the narrator who has to 'guide' the listener throughout the story. An important role of the narrator is when the sound of objects and events are ambiguous: [31, p. 80] give as an example the comparison between footsteps and a methodical stacking of bricks. In those cases, it is the job of the Narrator to disambiguate the origin of the sound by giving relevant hints in their speech, for example by mentioning that the character is walking.

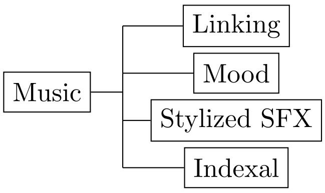


Figure 2.9: Types of music used in radio drama

2.4.3 Emanation Speech

Emanation speech is unintelligible speech possibly used in scene construction, such as the crowd speaking in the background. The presence of emanation speech can be used to signify mystery, symbolise emotional or intellectual myopia, and even be used for comic effect [18, p. 83].

2.5 Music

Music is an essential part of radio and is not unique to radio drama. It is however important for radio drama due to its ability to evoke emotions in the listener but its role does not end there[3, p. 50]. While music has many different roles in radio drama, we mainly identify *mood music*, *linking music*, *stylised sound effects*, and *indexal music*. Mood music is probably the most obvious use for music in the drama, used to evoke feelings or thoughts and usually plays in parallel with narration or lines of dialogue. Linking music is music joining scenes in a radio drama. An example of linking music can be seen in musical jingles in ads or signature tunes of radio programs. Linking music is used as the analogous of the theatre's 'curtain', signifying a change of scene. Other examples of linking music include the intro or outro of the radio drama show [49, Ch. 15]. A stylised sound effect is a short music part used in place of a sound effect. Those are usually sounds that are better at evoking the image of an event than a natural recording of the event itself. As an example, a percussion instrument can signify a thunderstorm better than a realistic recording [3, p. 50]. Other uses for stylised sound effects include e.g. communicating time changes [18, p. 167]. Finally, *indexal music* is music that occurs in the world of the drama, such as a character turning on a radio or playing an

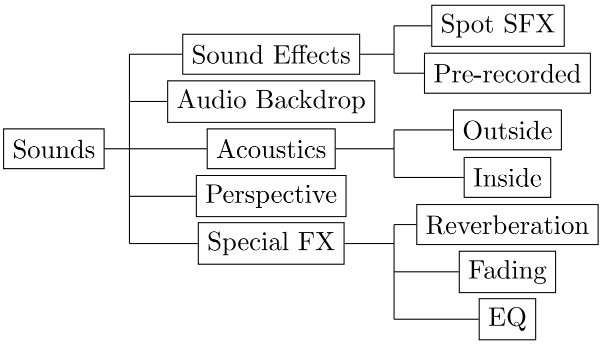


Figure 2.10: Types of sounds used in radio drama

instrument. [50, p. 51–2] elaborates further on the aforementioned functions of music.

2.6 Sounds

Sounds can be subdivided in four separate categories: *Sound effects (SFX)*, *Audio Backdrop (ATMOS)*, *acoustics*, and *perspective* (Figure 2.10) [3, p. 44]. Sound effects are distinct sounds which signify an event or location. They are usually accompanied by textual descriptions. Examples of SFX include the sound of a telephone ring, and the waves splashing on a beach shore. It is important to remind at this point that sounds need not be naturalistic: it needs to evoke the desired visual imagery and might not necessarily be a sound produced by the object it represents. According to Lance Sieveking, sound effects can be further categorised based on whether they describe a physical element in the scene and the effect they procure [3, 18]:

- *Realistic* and confirmatory: Confirming what was described. For example, the sound of a 'ship in a storm' after a character has introduced a storm.

- *Realistic* and evocative: Evoking emotions or state of mind using sounds that describe physical entities in the drama. For example, a rural, rustic atmosphere evokes a sense of 'peacefulness'.

- *Symbolic* and evocative: Evoking a character's confusion using unrealistic sounds such as abstract rhythms.

- *Conventionalised*: Spontaneously evoking stereotypical images in the mind of the listener: for example ‘the sound of a train‘.
- *Impressionistic*: Used to evoke the world of ‘dream’.

Sound effects are further categorised based on how they were sourced. We distinguish them between *spot sound effects*, and *pre-recorded sound effects.* The distinction lies on whether the SFX are created ‘live‘ in the studio during a recording of the drama, or are available in a pre-recorded form. An example of a spot SFX is the ‘live’ breaking of a frozen cabbage to give the impression of a breaking bone. Spot SFX have their origin on early cinema when music and SFX were generated ‘live’ by SFX technicians [18, p. 90]. The primary reason for this was the inability of recording to hold more than a couple of minutes of sound[3, p. 17]. The use of live SFX, however, evolved into common practice even after technology allowed for longer recordings, a reason being that some times spot SFX sounded more ‘realistic’ or were more practical to produce than their naturalistic counterpart [3, p. 145]. Sound effects, in general, tend to be used sparingly, targeting at a persuasive illusion of reality than realism, otherwise, the listener gets distracted and the image gets blurred [31, p. 79]. The producer has the role of selecting the most representative sound to signify an event or location; we refer to that sound effect as an *SFX signal.* Finally, *Audio Backdrop*, *Atmosphere*, or *ATMOS* are sounds suggesting the location of the drama [3, p. 143]. ATMOS are subtle non-intrusive sounds that ‘hint’ the location. They usually are imprecise, conveying a general impression rather than the specific location. Other elements of the radio drama, such as sound effects or textual descriptions are needed in order for the listener to pinpoint the exact location [3, p. 142].

2.6.1 Acoustics

In radio drama, *Acoustics* is the nature of sound in different locations [3, p. 150]. Acoustics are categorised to *outside acoustics* and *interior acoustics.* The former category refers to perceived ‘outdoors’ locations, such as meadows or schoolyards, and the later smaller spaces with perceived reverberation such as cathedrals, the inside of a car, etc.

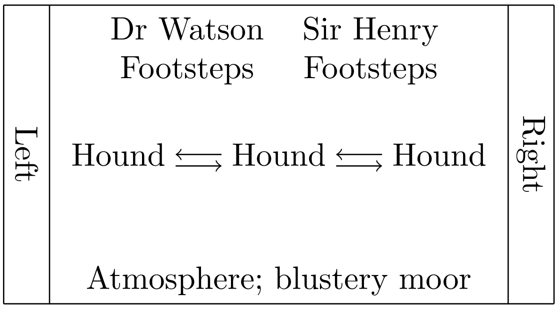


Figure 2.11: A hierarchy of sounds card for "The Hound of the Baskervilles (1998)" as shown in [3, p. 48]. Notice that Sir Henry and Dr Watson are the main characters in the scene and are higher in hierarchy followed by a hound that moves from left to right in the scene.

Change in acoustics is achieved either through careful design of the production studio or by using the audio effects of *Reverberation* and *equalisation* (EQ) at a mixing console or a DAW. As an example, acoustics of a large 'open' space like a meadow can be simulated by reducing the volume of the bass frequencies of the character voices, while a cathedral would be simulated by introducing a spacious reverberation effect. We discuss EQ and Reverberation in Section 2.7.

2.6.2 Perspective

As *Perspective* we will refer to the spatial relationships between the characters in the drama [3, p. 47]. It is important for the listener to understand how the characters interact in space. Characters can be distant or close to one another, to the left or right (in a stereo mix). A *point of listening* can be specified which can give both the relative positions of the characters as well signify their importance in the scene. This is usually achieved by establishing a hierarchy of sounds in the objects and events in a scene and can be realised by altering loudness and spatial positions of elements in the mix (e.g. with stereo or surround audio) [3, 51].

2.7 Audio Effects

We briefly mentioned the special effects of *EQ* and *Reverberation* in Section 2.6.1. While those can be used for simulation of acoustics and perspective they have other means which we explore in this section. Since there is a very large number of audio effects and discussing all of them would be impractical, we will limit our conversation to *Fading*, *Reverberation*, and *equalisation*. While *dynamic range compression* is also used in radio drama, it is typically at the mastering stage and not at the discourse level so we will not discuss it at the current stage.

Audio effects are not sounds themselves similar e.g. to speech or sound effects are, but the transformation of sounds in order to alter the perception those sounds convey in discourse. For example, by removing the treble from speech we can simulate someone speaking with their back to the listener [52]. We begin with describing *fading* which is the simpler of the three effects we will describe.

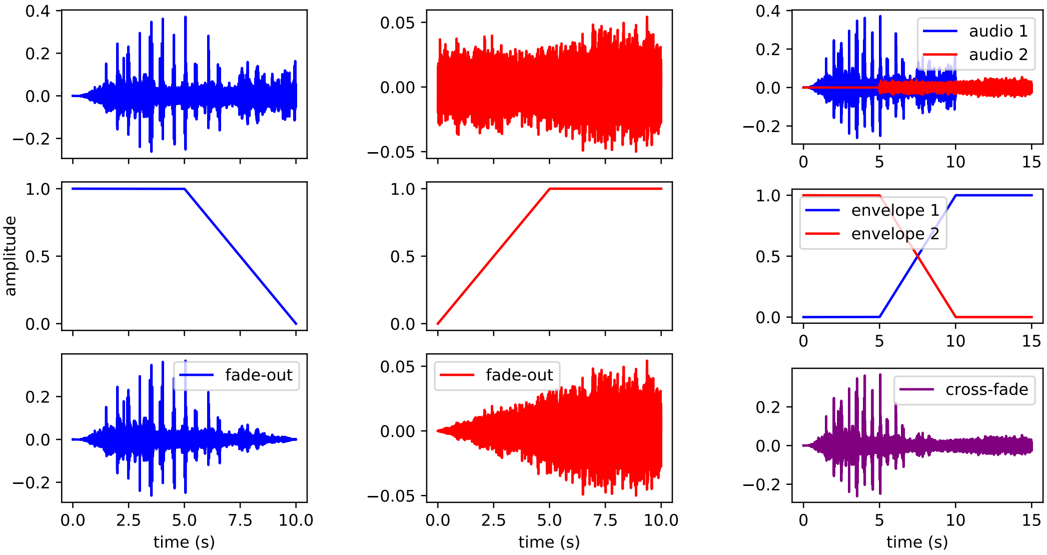
2.7.1 *Fading* and *Scene Transitions*

Fading is the gradual increase or decrease of the amplitude of a sound element of the radio drama. It is most often used on the atmosphere of a scene to signpost change of scene. It can be categorised in three types: *fade-in*, *fade-out*, and *cross-fade*:

- The *fade-out* effect gradually decreases the amplitude of a sound element (Figure 2.12(a)). It is characterised by the *fade-out time* (measure in *seconds*) which is the time it takes for the sound to dissipate completely.

- The *fade-in* effect gradually increases the amplitude of a sound element (Figure 2.12(b)). Similar to the fade-out time, *fade-in time* is the time it takes for the signal to reach from 0 amplitude to unit gain.

- The *cross-fade* effect gradually decreases the amplitude of a sound element while gradually increasing the amplitude of another, giving the perception of 'transitioning' between the two elements(Figure 2.12(c)). The *Cross-fade* time is the time it



(a) Fading out (b) Fading in (c) Fading in

Figure 2.12: *Fade-in*, *Fade-out* and *Cross-fade* applied to ATMOS. Top plots show the unprocessed sound, middle plot shows the fade envelopes and bottom plot the resulting effect.

takes for the transition to complete. Fading can be achieved either by manually modifying the volume levels of the signals we want to apply fading to, or by using an effect processor called a *Fader* [53, p. 279]. The *Fader* is a simple gain processor which gradually alters the gain of the input signals. Fader takes as inputs the signals to fade and the duration the fading effect will last f_t as well the shape of the fading envelope.

Fading is mostly used in radio drama for changes between different scenes. There are four common ways to signify a scene transition[3, p. 159]:

- A *fade-out* of the ATMOS of the first scene over roughly five seconds followed by a silence of one or two seconds and a fade-in of another five seconds of the ATMOS of the scene that follows. This is the most common transition in radio drama.
- A quick *crossfade* of two backdrops. This type of transition is used for quick scene changes e.g. when characters move from one room to the next in real-time.

- The *segue*. This is a sudden transition between the ATMOS of the scenes. A fader might not be used in this case and if used the fade-out/fade-in transition is very sharp.

- *Linking music* is used as a 'bridge' between scenes. It can often signpost changes of mood and is discussed in Section 2.5.

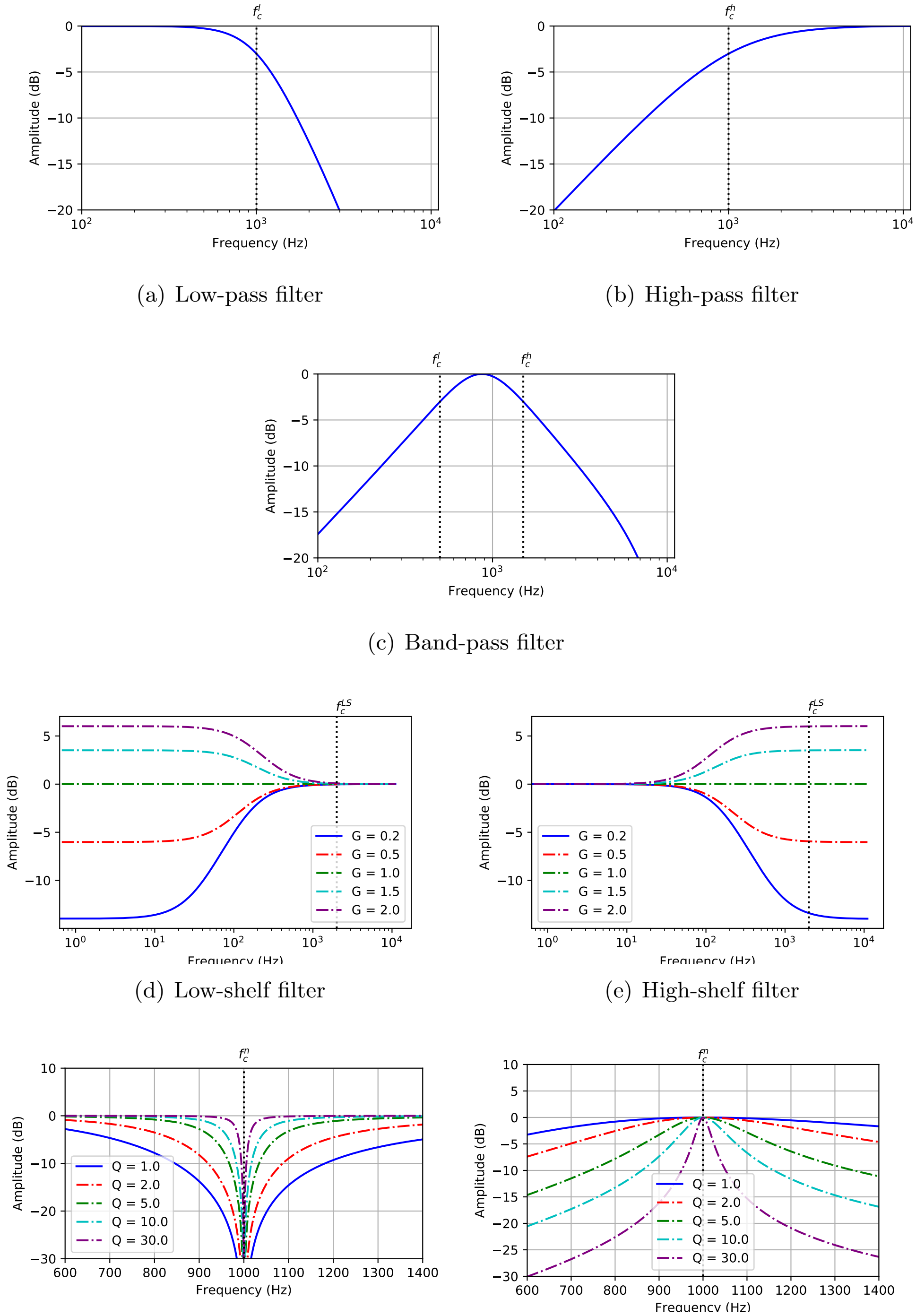
2.7.2 *Equalisation*

Equalisation is another important effect. It is used to amplify or attenuate part of the frequency content of sounds, music or speech. Like reverberation, it can be used to simulate spaces. For example, removing low frequencies from a sound can make it 'thinner', giving the impression that the source is 'outside' (e.g. in a meadow). Equalisation can be applied to a sound through the use of EQ processors. There are lots of different implementations of EQ processors but they are usually a combination of the following DSP filters [53, p. 59]:

- The *Low-pass filter* (LPF) removes frequencies of a sound *higher* than a chosen frequency f_c^l. The frequency response of an LPF can be seen in Figure 2.13(a). We can apply the effect to speech to simulate a person talking with their back on the microphone [52], simulate underwater sound [18, p. 84], and similar effects.

- The *High-pass filter* (HPF) removes frequencies of a sound *lower* than a chosen frequency f_c^h. The frequency response of a HPF can be seen in Figure 2.13(b). We already gave an example of its usage for 'thinning' the sound when e.g. simulating an outside environment. Another would be to eliminate the low frequencies of a voice to e.g. simulate a telephone call.

- The *Band-pass filters* (BPF) removes frequencies of a sound *outside* a chosen frequency band $[f_c^l, f_c^h]$. It can be implemented by applying an low-pass and then a high pass filter in sequence. The frequency response of a band-pass filter can be seen in Figure 2.13(c).



(a) Low-pass filter

(b) High-pass filter

(c) Band-pass filter

(d) Low-shelf filter

(e) High-shelf filter

(f) Notch filter

(g) Peak filter

Figure 2.13: Example frequency responses of the several building blocks for applying equalisation. Every filter is parameterised by at least a centre frequency (two for the case of the band-pass filter). The high/low-shelf filters are also parameterised by a gain factor G controlling how much the shelved part of the spectrum is enhanced/attenuated. The notch and peak filters are also controlled by a quality factor Q deciding how ‘narrow’ the notch or peak is.

- The *Notch filters* (NF) removes a narrow frequency range around a chosen centre frequency f_c^n while leaving the rest of the frequencies intact. When they instead attenuate all other frequencies except this narrow band, they are called *Peak filters* (PF). They are controlled by this centre frequency and a quality factor Q. Notch and Peak filters are used to remove of enhance specific frequencies. Their frequency responses can be seen in Figures 2.13(f) and 2.13(g) respectively.

- *Low-shelf filters* (LSF) attenuate or enhance frequencies of a sound *lower* than a chosen frequency f_c^{LS}. Aside from this frequency they are controlled by a parameter G. *High-shelf filters* (HSF) attenuate or enhance frequencies of a sound *higher* than a chosen frequency f_c^{HS}. Aside from this frequency they are controlled by a parameter G. Figure 2.13(b) shows the frequency response of an LSF and 2.13(b) of an HSF respectively. Shelving filters can be used where HP or LP filters are used in cases we do not want to remove completely but simply attenuate some of the frequencies.

2.8 Silence

As the name implies *silence* is the absence of sound. Silence is used to alter the perception by the listener of other elements in radio drama. As an example, when used within speech, it can be added to evoke 'expectancy' or introduce 'emotional overtones' [31, p. 88]. It also acts as a boundary between scenes, a lapse in time or a change in location. Other uses of silences are to e.g. introduce irony, or humour and to create dramatic tension [3, p. 57].

The role of silence has been extensively studied in speech in the form of *pauses*. Distribution of pauses can hint on the type and structure of speech and are essential for comprehension [54]. There is a variety of roles for 'pauses' in different settings such as reading or discussion, communicate emotion [55], and even across age [56]. In this thesis, we will only consider pauses in dialogue.

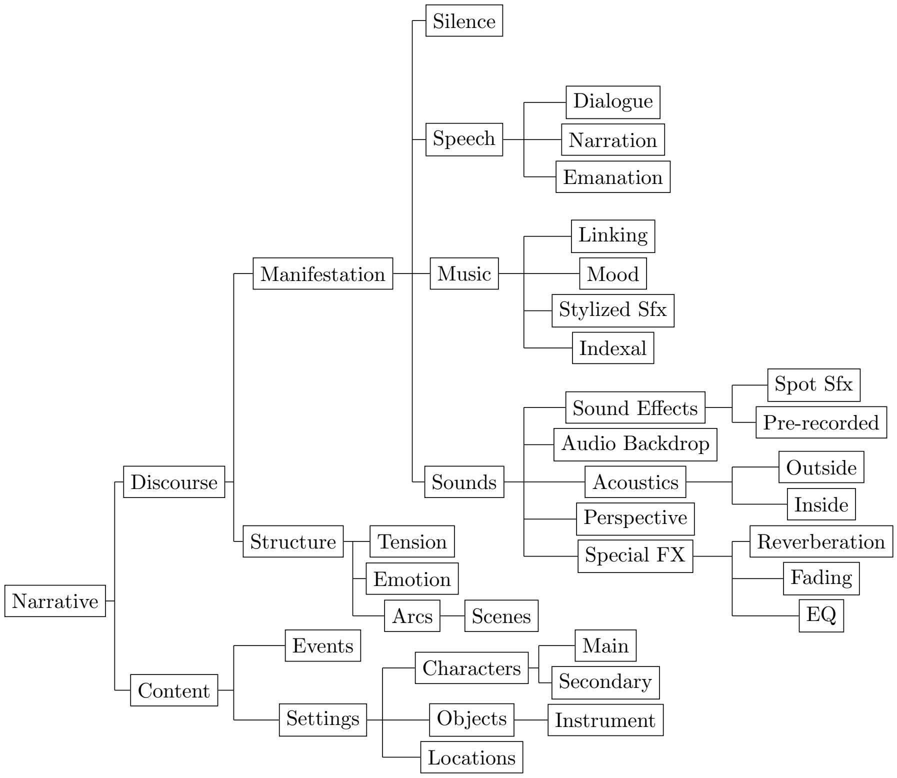


Figure 2.14: A taxonomy of narrative elements for radio drama

2.9 Summary

In this chapter, we introduced the narrative elements of radio drama. A diagram showing the overall taxonomy can be seen in Figure 2.14 and the way structural elements affect radio drama manifestation in Table 2-A.

Element	**Attributes**	**Affects**
Characters	Age Gender Orientation Ethnicity, ...	Dialogue SFX Perspective Emotion
Objects	–	SFX
Locations	–	SFX Audio Backdrop Perspective Emanation Speech Indexal Music Reverberation Equalisation
Scenes	–	Tension Stylised SFX Linking music Fading
Tension	–	Mood Music Silence
Emotion	–	Dialogue Mood Music

Table 2-A: Narrative elements, their attributes, and the elements of radio drama they affect. *Element* contains the elements we discussed about in this section. *Attributes* are properties of those (e.g. a character is of specific age). Finally, the elements in *Affects* gives the part of radio drama production that is affected by the narrative element in *Element* (e.g. *Emotion* in story can affect the choice of *Mood Music*).

Chapter 3

Computational Methods for Extracting Information from Stories

3.1 Introduction

Information extraction refers to the processes used to extract specific information from some form of free text. In this chapter, we apply information extraction techniques to extract information from stories. This information can be either names of entities, phrases relating to instances of things, relations between them, events where those participate, etc. [57, p. 4v]. There are a variety of methods used to do information extraction from text in free form. We will be using information extraction methods to analyse unstructured text in order to recognise and identify the narrative elements in Table 2-A. We will use two approaches, rule-based Open Information Extraction [58] and statistical methods entity and relation extraction [57, p. 497a,b]. An initial approach at extracting information about characters, acting lines, and story locations was presented in [59]. We begin this chapter by discussing how we can utilise knowledge about the real world when extracting information from text in Section 3.2. Information extraction about characters as well as story dialog is discussed in Section 3.3. Extracting information about story locations is discussed in Section 3.4. Section 3.5 discusses methods for discussing character, and story emotion information. Section 3.7 discusses a simple method for identifying story events, and Section 3.8 briefly discusses suspense. While most of the above sections discuss techniques already found in existing literature, the chapter introduces some original contributions as well:

1. The process for joint identification of characters, assignment of their voices and tags describing their roles in Section 3.3 in the context of adapting a written story to radio drama. While established methods are also used in that context from the NLP pipeline (Tokenization, Tagging, Dependency Parsing, Word Sense Disambiguation and Coreference Resolution) they are used in that section for the purpose of extracting characters, their roles, and their lines in order to direct actors portraying these characters.

2. The method for spatial role labelling presented in Section 3.4 which slightly improves on the state of the art on accuracy metrics on two previously established corpora.

3. In Section 3.5, the adaptation of an established technique for extracting emotions from the text of fairy tales for the purpose of directing actors as well as retrieving music for radio dramas.

Finally, the biggest contribution of this chapter is the use of established, and original methods in the context of extracting information from stories in order to direct radio drama production.

3.2 Utilising external knowledge

Before we discuss methods for extracting knowledge from text, we refer to methods of utilising real-world knowledge, an ability essential to information extraction. One way to introduce such knowledge to our methods is to use a knowledge base which was constructed specifically for this.

3.2.1 ConceptNet

One such knowledge base is ConceptNet [60], a freely available knowledge base that encodes a large set of commonsense knowledge in a single graph network. It represents real-world entities as *concepts* and their relations between them as edges that link those concepts. It encodes simple relations between concepts, such as *"love is a type of emotion"*, lexical *"love is the antonym of hate"*, and also more complex ones such as *"marriage is motivated by love"*. It is often used in the literature of Natural Language Processing to augment statistical methods with some knowledge derived from common sense. In this thesis, we use version 5.5 [61] which is the most recent. ConcentNet is provided on the web and allows querying by accessing appropriate `HTTP` requests. For example the query:

"Show me what a knife can be used for."

can be accessed with the `HTTP` request:

```
    http://api.conceptnet.io/query?rel=/r/UsedFor&start=/c/en/knife&
end=/c/en
```

where `en/knife` is the concept of a *knife* (an English word), and `r/UsedFor` returns the concepts of the things that can be done with a knife. The result is a list of ConceptNet entries such as `en/stabbing` (a *knife* can be used for *stabbing*) or `en/butter` (a *knife* can be used for *butter*). We use such relations in Section 5.4 where we align emotions as extracted from text with emotions from a dataset used for music retrieval. ConceptNet also provides *word embeddings* that map each word to a vector in a way that encodes some of the common sense relations available in the knowledge base. We discuss more about word embeddings below. Before that, however, we discuss about WordNet [62], another commonly used knowledge base.

Synonym set	**Description**
`king.n.01`	A male sovereign; ruler of a kingdom
`king.n.02`	A competitor who holds a preeminent position
`baron.n.03`	A very wealthy or powerful businessman
`king.n.04`	Preeminence in a particular category or group or field
`king.n.05`	Billie Jean King
`king.n.06`	B.B. King
`king.n.07`	Martin Luther King
`king.n.08`	The checker piece
`king.n.09`	A playing card with a picture of a king
`king.n.10`	The chess king

Table 3-A: Synonym sets for the lemma *king*. Descriptions are taken from the Princeton WordNet 3.1 web interface.

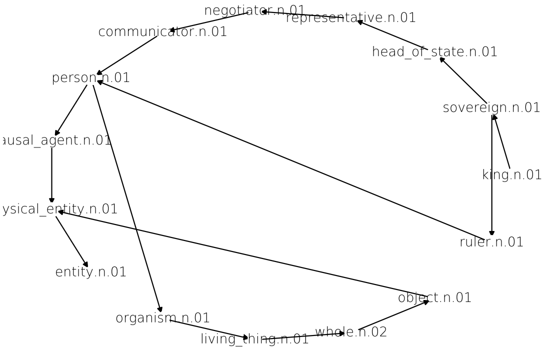


Figure 3.1: The hypernym graph for sense `king.n.01`

3.2.2 WordNet

WordNet is an electronic lexical database of nouns, verbs, and adjectives. Entities in WordNet are organised in sets of synonyms (or *senses*). Such senses might correspond to different meanings of a word, or *lemma*. For example the word *king* corresponds to the 10 senses shown in Table 3-A. We observe that the same lemma might correspond to multiple different synonym sets, a problem we discuss further in Section 3.3.5. Aside from retrieving synonyms for various words, WordNet also lets us inquire about relations between those sets and parent sets, or *hypernyms*. Hypernyms can be understood as senses that, given a sense X, correspond to Y in:

$$X \text{ is a type of } Y \tag{3.1}$$

Or otherwise Y is a generalisation of X. For example a `king.n.01` is a type of `ruler.n.01` which subsequently is a type of `person.n.01`. This means that *person.n.01* is a hypernym of `ruler.n.01` which is a hypernym of `king.n.01`. Such chain of hypernyms can be represented as a graph, as in Figure 3.1. *Hyponyms* are the opposite relation of Eq. 3.1. Y is a hyponym of X when Y is a type of X. While WordNet also contains information about other relations such as *meronyms*, *holonyms*, etc., we only utilise information hypernyms and hyponyms in this thesis.

3.2.3 Word Embeddings

In order to analyse the text of stories we will consider the story text to be a sequence of word *tokens* which is a discrete type used in natural language processing to represent text. This type of representation is appropriate when using rule-based approaches and algorithms. There will be cases when we will need to use approaches that operate and expect real numbers (or vectors) as inputs, such as the statistical approach used in Section 3.4.2. In such cases we need to assign an appropriate numerical value to tokens. Constructing a method for assigning vectors to tokens is a design decision and depends on the task. The main concerns are:

- All words should be able to be represented that way.
- Words should be uniquely represented.
- Given a collection of words, a representation should be able to represent that collection uniquely.

The last requirement comes from the need to represent structures that contain words, such as sentences or even whole documents. A straight-forward but naive way to assign a unique number n to each word and use Kronecker's delta δ_n:

$$\delta_n = \begin{cases} 1 & \text{if word is assigned unique number } n \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

where each δ_n is of the same size $|D| \times 1$ where $|D|$ is the total number of words in a lexicon D. In this way, collections of words can be represented using sums of vectors of the individual word representations:

$$\boldsymbol{t}_n = \delta_n \qquad \forall n \in 1..|D| \tag{3.3}$$

$$\boldsymbol{q} = \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{t}_i \tag{3.4}$$

This representation in Eq. 3.2 is called *one-hot* representation. The vector $\boldsymbol{q}$ in Eq. 3.4 then represents a collection of M words (each element q_i is 1 or 0 depending on whether the i-th word can be found in that collection). The factor $\frac{1}{M}$ constraints its norm to 1 for convenience of computations. This representation is sufficient for use in many approaches that expect numerical values. The word representation in Eq. 3.3 has, however, some limitations:

1. The representation vector $\boldsymbol{t_n}$ has size $|D| \times 1$ where $|D|$ is the total number of words. This means that when dealing with real-world data, we end up having to deal with very large vectors. While storing these vectors in memory is not a problem since the vectors are sparse, using them in statistical approaches might result in matrices that are not sparse (e.g. weight matrices of neural networks) and thus run into memory problems.

2. There is no correlation between the representations of individual words. Consider for example the words *large* and *larger*. One is the comparative form of the other however their representation using one-hot vectors are orthogonal (they have a cosine similarity value of 0). It is desirable to devise a representation where corre-

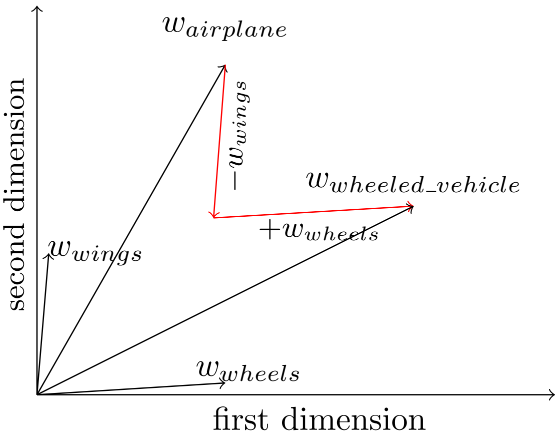


Figure 3.2: Visualised word analogies. Each vector $\boldsymbol{w}_x$ is the 300-dimensions word embedding for word x. For reasons of convenience we visualise the vectors in a 'flattened' 2D space.

lations in the vector format map to correlations in their real-world meanings.

A way to address these problems is found in *word embeddings*. These are low-dimensional dense representations (usually 300 hundred dimensions – thus addressing limitation 1 above) that can encode some of the real-world relationships we discussed above. Those relationships can be explored e.g. using analogies of the form:

x is to y like z is to w.

For example if we have the following analogy:

Wings are to aeroplane as *wheels* are to x

we can simply solve the following equation:

$$\boldsymbol{w}_{aeroplane} - \boldsymbol{w}_{wings} = \boldsymbol{x} - \boldsymbol{w}_{wheels} \Leftrightarrow \tag{3.5}$$

$$x = \boldsymbol{w}_{aeroplane} - \boldsymbol{w}_{wings} + \boldsymbol{w}_{wheels} \tag{3.6}$$

Which can be also interpreted as *that which is like an aeroplane, without wings, but with wheels*. The result is going to be a new point $\boldsymbol{x}$ which, in the case of ConceptNet, is very 'close' to $w_{wheeled\_vehicle}$. We visualise such analogies in Figure 3.2. Embedding representations are not limited to tokens but can be used for any type of discrete data, such as characters, tags, or sets of tags. In Section 5.3 we use word embeddings to devise a retrieval system for Room Impulse Responses that uses word synonyms to supplement the tags used to search a database of audio files.

3.3 Extracting information for characters

Characters in radio dramas are not very different from characters in stories. They are fictional entities with agency who think and interact with the story world. Those can be people, animals, or even inanimate objects with personality[1]. When adapting a source story to radio drama, one important task is to identify characters in order for actors to impersonate them. Extracting characters for radio drama adaptation is not very different from extracting information for characters in stories. There have been prior efforts to recognise characters in stories by assign roles to them using both rule-based approaches [63–65] as well as statistical methods [66].

In [63], the authors assign roles with the use of genetic algorithms. They try to construct adjacency matrices between noun phrases and roles which are similar to matrices from their corpus. This approach presupposes a library of pre-existing roles which are suitable only for tales that can be analysed with Proppian roles [48]. In [64, 65] the authors also begin by identifying noun phrases which they filter to characters using hand-crafted rules and a folk-tale ontology, based on the Proppian morphology as well which is subject to the same limitation. A more recent, rule-based approach that does not depend directly on an ontology for identifying characters is taken in [67] where the authors use abstractive summarisation to identify thematic roles characters might participate in. While the authors subsequently use a database of Proppian annotations to assign the roles, this is not needed for character annotation.

A different approach is taken in [66] where the authors use a large corpus of annotated Dutch novels and statistical approaches for named entity recognition to extract charac-

[1] Lewis Carroll's "Alice in Wonderland" had both animals and objects as characters.

ters. While this approach does not suffer from the limitation of identifying characters that adhere to the Proppian morphology, its ability to identify characters depends on the annotation and curation of a large amount of text data which is a long and arduous process. In [68], the authors apply a hybrid approach that combines Machine Learning with hand-designed rules to detect animacy in 142 narratives which also included stories where characters were inanimate objects. This approach also requires an annotated corpus. In [69] the same authors provide a simple supervised learning-based approach on classifying characters in an annotated corpus of 46 Russian folktales.

3.3.1 A process for joint character identification and voice assignment

Most of the methods described above require an annotated corpus. They also do full character extraction and analysis for every character which is not necessary for radio drama. In radio drama, the main element of expression for characters is the Actor, via their speech. Information shown in Table 2-A such as age, sex, status and emotional and physiological states can be expressed directly through the character's voice [3, p. 184]. Events, where characters participate but do not speak, are expressed mainly through sound effects and exact identification of characters results in little extra information. For this reason, we develop a character identification approach for characters that use their voice in a story.

The process we developed achieves jointly three tasks:

1. Identify Characters

2. Identify their acting lines

3. Assigns them a set of tags that correspond to properties in Table 2-A

A summary can be seen in Figure 3.3. The input to our process is the raw story text. Below we explain step by step each box in Figure 3.3. We will give an example for each step in the context of the short fable *"Belling the cat"*

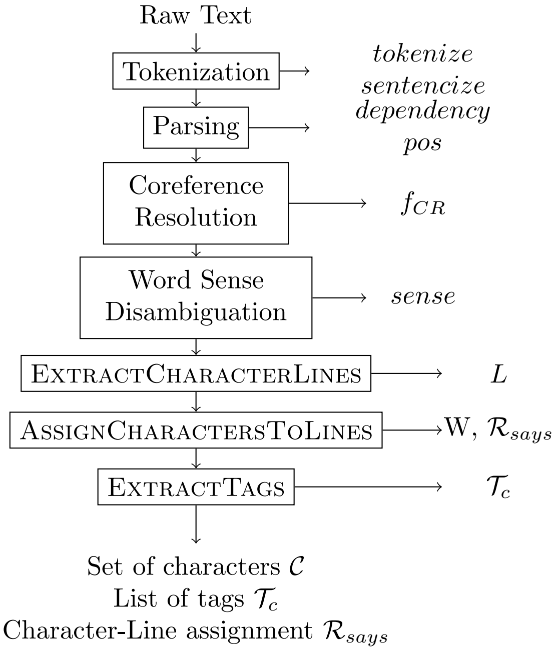


Figure 3.3: The Character extraction and assignment to speech acts process

> Long ago, the mice had a general council to consider what measures they could take to outwit their common enemy, the Cat. Some said this, and some said that; but at last **a *young* mouse** got up and said **he** had a proposal to make, which **he** thought would meet the case. *”You will all agree,”* said **he** *”that our chief danger consists in the sly and treacherous manner in which the enemy approaches us. [...] could easily retire while she was in the neighbourhood.”* This proposal met with general applause, until **an *old* mouse** got up and said: *”That is all very well, but who is to bell the Cat?”* The mice looked at one another and nobody spoke.

The process’ goal is to assign characters in **bold** to dialogue lines in *italics* and also assign to the characters the properties in ***bold italics***.

3.3.2 Tokenization and Sentence Segmentation

Tokenization is the pre-processing step of splitting natural text to basic units, called *tokens* and is part of almost every NLP process. Here we use Spacy [70] which tokenizes

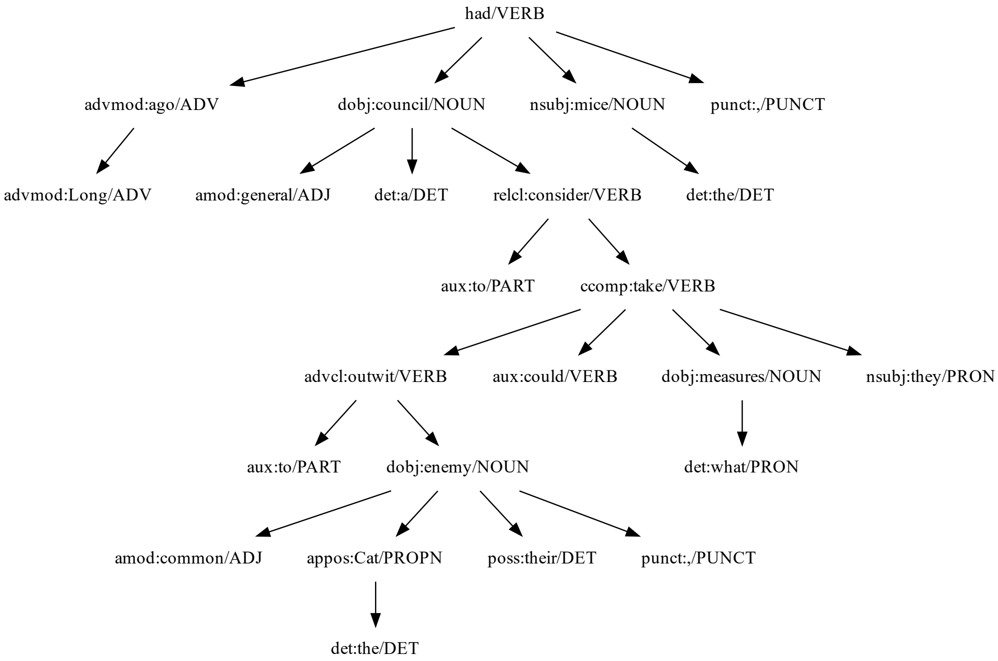


Figure 3.4: The parsing tree of the first sentence in *"Belling the cat"*. Each node represents a word token and is labelled using the universal dependency of the token, the token word, and the part-of-speech of that token. The list of universal dependencies and their meanings can be found in [4].

to word tokens according to the Penn TreeBank [71]. We also split the text into sentences. The story becomes segmented as:

$$\underbrace{\overbrace{\text{Long}}^{\text{Token}_1}\ \overbrace{\text{ago}}^{\text{Token}_2}\ \overbrace{,}^{\text{Token}_3}\ \dots\ \overbrace{\text{the}}^{\text{Token}_{23}}\ \overbrace{\text{Cat}}^{\text{Token}_{24}}\ \overbrace{.}^{\text{Token}_{25}}}_{\text{Sentence}_1}\ \underbrace{\overbrace{\text{Some}}^{\text{Token}_{26}}\ \dots\dots\ \text{the case}}_{\text{Sentence}_2}\dots$$

3.3.3 Parsing

Parsing is another step omnipresent in NLP. Dependencies and part-of-speech tags are assigned by SPACY [70]. This process transforms each sentence into a tree similar to the one in Figure 3.4. We define the functions *dependency* and *pos* for extracting the dependency and part-of-speech of a token accordingly:

$$dependency(token) = \qquad dep \tag{3.7}$$

$$pos(token) = \qquad pos \tag{3.8}$$

As an example, for the word-token 'Long' in "Long ago, the . . . ", we have:

$$dependency(\underbrace{\text{Long}}_{\text{Token}_1}) = \qquad \text{advmod} \tag{3.9}$$

$$pos(\underbrace{\text{Long}}_{\text{Token}_1}) = \qquad \text{ADV} \tag{3.10}$$

3.3.4 Coreference Resolution

Coreference Resolution(CR) [57, p. 61] is another common task in text-based information retrieval for stories [63–65, 67, 72]. Its role is to resolve referents, such as pronouns or *this/that* to the entities they refer to. As an example consider the sentence:

> *...Some said this and some said that, but at last* ***a young mouse*** *got up and said* **he** *had a proposal to make, which* **he** *thought would meet the case...*

In the above example, the pronoun **he** refers to **a young mouse**. Knowing this reference comes handy when we need to answer questions such as *Who thought this proposal would meet the case?* The answer comes by pointing to the subject of the verb *thought* which is *he* which subsequently refers to *a young mouse.* In our case we use co-reference resolution in this way, in order to gather more information about the characters in the story.

While automatic co-reference resolution still suffers and is not recommended for question answering tasks for stories [73], our use is limited into referencing subjects of the speech act triggers mentioned above and assigning the 'sex' property of a character, when such property cannot be identified otherwise. There are plenty of automatic coreference resolution algorithms to choose from, we make use of the deep reinforcement

Referent	**Entity**	**Sentence**
they	the mice	Long ago **the mice** had a . . . measures **they** could take . . .
their	the mice	Long ago **the mice** had a . . . outwit **their** common . . .
he	a young mouse	at last **a young mouse** got up and said **he** had . . .
he	a young mouse	at last **a young mouse** got up . . . which **he** thought . . .
he	a young mouse	at last **a young mouse** . . . " said **he** . . .
this	proposal	**a proposal . . . the case** ". . . " **this** proposal . . .
her	the Cat	enemy, **the Cat** . . . signal of **her** approach . . .
she	the Cat	enemy, **the Cat** . . . **she** was about . . .

Table 3-B: The contents of the list of subject-trigger pairs W for the short fable. For each show we show its index i and the sentence that corresponds to the pair.

learning-based algorithm in [74], mainly due to the ease of use of the NeuralCoref[2] implementation. We construct a mapping function for every pronoun P that refers to character C:

$$f_{CR}(\text{pronoun}) = \text{character} \tag{3.11}$$

We will then use this mapping we come across sentences such as:

"You will all agree," said **he**

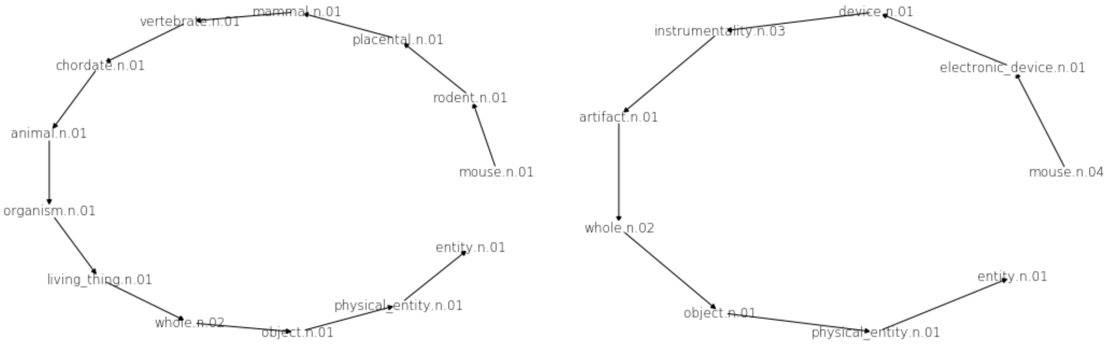
To refer to the original character (the young mouse) and gather more information about it (that it was young). For the whole fable text we have the following mappings:

$$f_{CR} = \{\langle \underbrace{\text{they}}_{\text{Token}_{15}}, \underbrace{\text{the}}_{\text{Token}_4} \underbrace{\textit{mice}}_{\text{Token}_5}\rangle, \langle \underbrace{\text{their}}_{\text{Token}_{20}}, \underbrace{\text{the}}_{\text{Token}_4} \underbrace{\text{mice}}_{\text{Token}_5}\rangle, \langle \underbrace{\text{he}}_{\text{Token}_{48}}, \underbrace{\text{a}}_{\text{Token}_{40}} \underbrace{\text{young}}_{\text{Token}_{41}} \underbrace{\text{mouse}}_{\text{Token}_{42}}\rangle, \ldots\} \tag{3.12}$$

3.3.5 Word Sense Disambiguation

A pre-processing step is taken to clarify the meaning of the words. To do that, we used the neural WSD model introduced in [75]. As an example we repeat the first sentence

[2] https://github.com/huggingface/neuralcoref



(a) Mouse (the rodent)

(b) Computer Mouse (the instrument)

Figure 3.5: Hypernym graphs of `mouse.n.01` and `mouse.n.04`

of the story given above together with the meanings of each word:

$$\overbrace{\text{Long}}^{\text{long.a.01}}\ \text{ago, the}\ \overbrace{\text{mice}}^{\text{mouse.n.01}}\ \overbrace{\text{had}}^{\text{experience.v.03}}\ \text{a}\ \overbrace{\text{general}}^{\text{general.a.01}}\ \overbrace{\text{council}}^{\text{council.n.03}}\ \text{to}\ \overbrace{\text{consider}}^{\text{study.v.03}}$$

$$\text{what}\ \underbrace{\text{measures}}_{\text{measure.n.01}}\ \text{they could}\ \underbrace{\text{take}}_{\text{take.v.01}}\ \text{to}\ \underbrace{\text{outwit}}_{\text{outwit.v.01}}\ \text{their}\ \underbrace{\text{common}}_{\text{common.a.01}}\ \underbrace{\text{enemy}}_{\text{enemy.n.04}}\ \text{, the}$$

$$\underbrace{\text{cat}}_{\text{cat.n.01}}\ .$$

Where the identifiers above the words in the text are WordNet senses [62]. Using word sense disambiguation in the above sentence establishes that the word *mice* refers to the rodents and not the computer interface apparatus. This helps later to construct the appropriate hypernym graphs for each word and assign properties to the identified characters. As an example we would like to assign the *old mouse* character in the story the property of it being a living mouse and not a computer mouse; we can distinguish between the two by looking at their hypernym graphs (Figure 3.5). For our algorithms, we regard word sense disambiguation as a function *sense* which returns the sense of a token t:

$$sense(token) = wordnet\_sense \tag{3.13}$$

Or, for the case of "Belling the cat", we can give it in the following relation-set form:

$$sense = \{\langle \underbrace{\text{long}}_{\text{Token}_1}, \texttt{long.a.01}\rangle, \langle \underbrace{\text{mouse}}_{\text{Token}_{42}}, \texttt{mouse.n.01}\rangle, \dots\} \tag{3.14}$$

3.3.6 Character and dialogue line extraction

Entity Recognition is part of many information retrieval tasks. In our case the entities we need to extract are *dialogue lines* and *trigger words* for speech (marked with *italics* in the story presented above). Characters are extracted indirectly afterwards using this information. Assigning lines to characters is part of the task of *speaker identification*, or *quotation attribution*, and it has been studied before for extracting direct and indirect speech from newspapers [76, 77], as well as fictional narrative [78–80]. We approach this task using a mix of simple grammar rules and linear programming. The method can be summarised to extracting a word-trigger that signposts a speech act and then using linear programming to assign that word to the character who speaks, as well as the dialogue line of the speech act. Something similar can be found in [80] where the authors assign speakers to speech verbs. The main difference to that work is that while its authors use machine learning and an annotated corpus, we use a simple linear programming algorithm and do not require labelled data.

Like earlier works on quotation attribution, we assume dialogue lines come as direct speech in quotation marks, and are easily captured using the pattern:

$$\texttt{<CLINE>} := \text{``}\ \texttt{<TOKEN>} * \text{''} \tag{3.15}$$

where `<CLINE>` represents a dialogue line and `<TOKEN>` a token extracted from the *tagging* step. A simple constraint is enforced to not allow overlapping dialogue lines. The use of the star operator ($*$) denotes that any number of `<TOKEN>` can be inside the quotation marks (" and "). We store those dialogue lines in a list L for later use. The algorithm

Algorithm 1 Extracting dialogue lines. The extraction uses a pattern matching algorithm (line 3) and keeps the matches that do not overlap with other patterns in the list. $match(P, T, M)$ is interpreted as: "Token sequence M is a match for pattern P in token sequence T"

```
1: procedure ExtractDialogueLines($t_{1...|T|}$)
2:     $L \leftarrow []$                                                    ▷ An empty list
3:     $M \leftarrow [t_{k...m} : match(\text{"}\texttt{<TOKEN>} * \text{"}, t_{1...|T|}, t_{k...m})]$     ▷ Matches pattern in Eq 3.15
4:     for $i \in 1 \ldots |M|$ do
5:         if $\nexists l \in L : M_i$ overlaps $l$ then
6:             $L \leftarrow L.M_i$
       return $L$
```

i	**Dialogue line** l_i
1	"You will all agree,"
2	"that our chief ... in the neighbourhood"
3	"That is all very well ... the Cat?"

Table 3-C: The contents of the list of dialogue lines L. i is the index of the dialogue line l_i in the list.

is given in Algorithm 1. In the case of *"Belling the cat"*, the contents of L are given in Table 3-C. The aforementioned algorithm assumes that all speech is given as direct speech. In general, however, this assumption does not hold, nearly 50% of speech found in text is written as *indirect speech.* An example of such speech from *"Belling the cat"* is the following:

A young mouse [...] said *he had a proposal to make.*

Where in italics is what was said by the young mouse. This is clearly not captured by simply matching spans in quotation marks as above. It has been shown however that similar direct quotation extraction approaches can be easily adapted to work with indirect speech as well [81, 82].

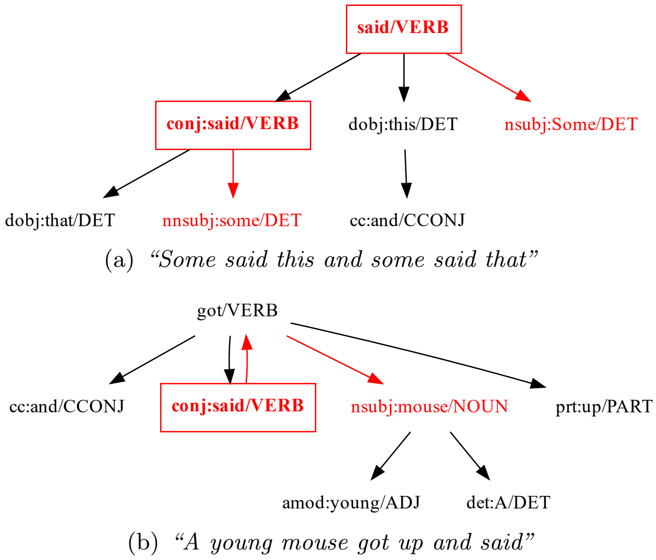
Since we are looking to extract characters that use their voice, the other entity we need to extract is *speech act triggers.* Those are words that signpost a dialogue line, such as *said*, *questioned*, *exclaimed*, or even animal sounds such as *roar* or *squeak* when the characters are animals. Speech act triggers might also reveal information about the way of the line is spoken, such as *grumbled*, *complained* or *murmured.* We identify triggers by

Lemma	Example Hyponyms	Example sentence

`communicate.v.01`	`say.v.05`
	`ask.v.01`
	`question.v.01`
	`continue.v.02`
		A young mouse said: "…"
		The young mouse asked: "…"
		"Is this true?", Mary questioned.
		"As I was saying", the judge continued.

`express.v.02`	`proclaim.v.02` `state.v.01`	The president proclaimed: "…" The scientist stated: "…"
`utter.v.02`	`grumble.v.03`	The old pirate grumbled: "…"
`complain.v.01`	`murmur.v.02`	"…", murmured the young child.
`interrupt.v.01`	`interrupt.v.03` `chime_in.v.01`	"This is wrong", Nicholas interrupted

Table 3-D: Example speech act trigger senses



(a) *"Some said this and some said that"*

(b) *"A young mouse got up and said"*

Figure 3.6: Capturing subjects for trigger verbs. Triggers are in square nodes with red colour and captured subjects with red text. The red arrows show the path the algorithm is taking in order to capture the subject of each trigger. In (a) the subjects are captured by simply examining the children of the *said* instances. In (b) the algorithm needs to go a step 'up' and look the subject in the children of the root node. This process is repeated as long as the root node is of dependency type 'conj'.

examining each token's hypernym graph and selecting those that are hyponyms of some trigger sense given in Table 3-D.

Together with identifying trigger words we also keep their subjects from the dependency tree using a simple criterion:

i	**Pair** $(\mathbf{s_i}, \mathbf{v_i})$	**Sentence**
1	(Some, said)	. . . some said this . . .
2	(some, said)	. . . and some said that . . .
3	(mouse, said)	. . . a young mouse got up and said. . .
4	(he, said)	. . . ” said he “ . . .
5	(mouse, said)	. . . until a young mouse got up and said . . .

Table 3-E: The contents of the list of subject-trigger pairs W for the short fable. For each pair we show its index i and the sentence that it is found in.

1. **If** the trigger verb has an `nsubj` dependency in its leaves, **then** match that as the subject.
2. **Else if** the trigger does not have an `nsubj` dependency in its children and is part of a coordinate conjunction (`conj`) **then** search its parent for an `nsubj` dependency and match that as the subject.

Two examples of subject captures for triggers in two small phrases can be seen in Figure 3.6. For our purpose, we store the pairs (s_i, v_i) where v_i are the captured trigger verbs and s_i their associated subjects, to a list W. The triggers and the subjects stored from the fable above are:

$$W = \left[(\underbrace{\text{Some}}_{\text{Token}_{28}}, \underbrace{\text{said}}_{\text{Token}_{29}}), (\underbrace{\text{some}}_{\text{Token}_{33}}, \underbrace{\text{said}}_{\text{Token}_{34}}), (\underbrace{\text{he}}_{\text{Token}_{73}}, \underbrace{\text{said}}_{\text{Token}_{72}}) \right] \tag{3.16}$$

The sentences they belong to can be see in Table 3-E.

3.3.7 Character to lines assignment

After we have identified candidate characters in the text, either via extracting subject-trigger pairs $(s_i, v_i) \in W$ or coreference resolution, we need to assign them to the dialogue lines $l_i \in L$. We approach the problem as an assignment problem: that is, to find a symmetric relation such that:

$$\mathcal{R}_{\text{says}} = \{\langle (s_i, v_i), c_j \rangle \quad \forall (s_i, v_i) \in W', \forall c_j \in L : \text{“ } s_i \text{ says line } c_j \text{ ”}\} \tag{3.17}$$

Where $W' \subseteq W$ the subject-trigger pairs that correspond to a character speaking (for example "some **said** this" is not a legal subject-trigger pair in our case since there is no dialogue line associated with it). Normally, characters s_i speak more than one line l_j but this will be dealt with later. We rephrase the problem of finding subscripts i, j as finding the adjacency matrix $\mathbf{X}_{i,j}$ such as that a cost:

$$cost = \sum_i \sum_j \mathbf{C}_{i,j} \mathbf{X}_{i,j} \tag{3.18}$$

is minimised. This is a well-studied problem and a solution can found with linear programming using the Hungarian algorithm [83]. The main problem is appropriately constructing the matrix $\mathbf{C_{i,j}}$. From the fables in our dataset, we observe that the "closer" the dialogue lines are to a trigger word, the higher the probability of it being assigned to a trigger word. We exploit this information while constructing $\mathbf{C}$:

$$\mathbf{C}_{i,j} = \begin{cases} v_i^{\text{begin}} - l_i^{\text{end}} & \text{if} \quad v_i^{\text{begin}} > l_i^{\text{end}} \\ l_i^{\text{begin}} - v_i^{\text{end}} & \text{if} \quad l_i^{\text{begin}} > v_i^{\text{end}} \end{cases} \tag{3.19}$$

where s^{begin} and s^{end} is the beginning and ending token position of text segment. Minimising cost *cost* gives a solution where each possible subject-trigger pair $(s_i, v_i) \in W'$ is assigned to a dialogue line $l_j \in L$ while extraneous subject-trigger pairs (that do not correspond to a dialogue line) are eliminated. This would solve the assignment problem exactly when $|W'| = |L|$. When $|W'| < |L|$ (as is usually the case in stories) there are $|L_u| = |L| - |W'|$ unassigned dialogue lines. To overcome this we assign them to the subject-trigger pair of the previous dialogue line:

$$\exists c_j \in L_u, \exists \langle (s_i, v_i), c_{j-1} \rangle \in \mathcal{R}_{\text{says}} \rightarrow \langle (s_i, v_i), c_j \rangle \in \mathcal{R}_{\text{says}} \tag{3.20}$$

where L_u is the set of unassigned dialogue lines. An overview of the procedure can be seen in Algorithm 2.

Algorithm 2 An algorithm for assigning characters to dialogue lines. W_D is the gazette with senses denoting speech act shown in Table 3-D.

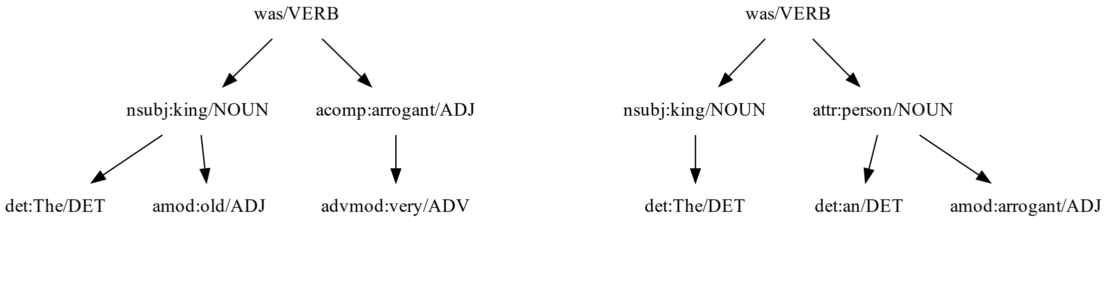
```
procedure AssignCharactersToLines($t_{1...|T|}, W_D$)
    $L \leftarrow$ ExtractDialogueLines($t_{1...|T|}$)
    $W \leftarrow []$                                              ▷ Empty list of tuples
    $\mathcal{T}_r \leftarrow \{t : t \in t_{1...|T|}, \exists w \in W_D, sense(t) = hyponymOf(w)\}$    ▷ All ‘trigger’ tokens
    for $t \in \mathcal{T}_r$ do
        $vp = verbPhraseOf(t)$                                     ▷ The ‘parent’ verb phrase of $t$
        if $\exists d \in descendants(t) : dependency(d) = nsubj$ then
            $np = nounPhraseOf(d)$                                 ▷ The ‘parent’ noun phrase of $d$
            if $\exists c : (np, c) \in f_{CR}$ then               ▷ If it refers to another noun phrase
                $W \leftarrow W.(c, vp)$
            else
                $W \leftarrow W.(np, vp)$
        else
            if $dependency(t) = cconj$ then
                $\rho = parent(t)$                                 ▷ Parent node in the parse tree
                if $\exists dindescendants(\rho) : t \neq d, dependency(d) = nsubj$ then
                    $np = nounPhraseOf(d)$
                    if $\exists c : (np, c) \in f_{CR}$ then       ▷ If it refers to another noun phrase
                        $W \leftarrow W.(c, vp)$
                    else
                        $W \leftarrow W.(np, vp)$
    $\mathbf{C} \leftarrow \underset{|W|\times|L|}{\mathbf{0}}$                     ▷ An $|W| \times |L|$ matrix full of 0s
    for $i \in 1 \ldots |W|$ do
        $(np, vp) \leftarrow W_i$
        for $j \in 1 \ldots |L|$ do
            $l \leftarrow L_j$
            $\mathbf{C}_{i,j} = \begin{cases} vp^{\text{begin}} - l^{\text{end}} & \text{if } vp^{\text{begin}} > l^{\text{end}} \\ l^{\text{begin}} - vp^{\text{end}} & \text{if } l^{\text{begin}} > vp^{\text{end}} \end{cases}$
    $I, J \leftarrow Hungarian(\mathbf{C})$                        ▷ Get assignment indices list $I$ and $J$
    for $i \in I$ do
        $(np, vp) \leftarrow W_i$
        for $j \in J$ do
            $\mathcal{R}_{\text{says}} \leftarrow \mathcal{R}_{\text{says}} \cup \{\langle (np, vp), L_i \rangle\}$
    return $\mathcal{R}_{\text{says}}$
```

3.3.8 Character property assignment

This part of the procedure assigns a series of tags:

$$\mathcal{T}_{\text{character}} = \{tag_1, tag_2, ...\} \tag{3.21}$$

to each of the characters extracted with the methods above. Those tags are any tag that can signify a character property listed in Table 2.2.3, such as *sex*, *age*, *title*, etc. A



(a) *"The old king was very arrogant"* (b) *"The king was an arrogant person"*

Figure 3.7: Different ways of expressing that the king was *old* and *arrogant*.

straight-forward way to infer such information is by observing the noun phrases associated with the characters or their referents (i.e. pronouns). As an example, in "Belling the cat" we have two characters: *young mouse*, and an *old mouse*. Let us revisit the first speech act:

"...**a young mouse** got up and said..."'

Its parse tree can be seen again in Figure 3.6(b). We observe that we can derive attributes for the character by following the `nsubj:mouse/NOUN` subtree by observing the `amod` leaf dependencies. In Figure 3.6(b) we observe we have a single `amod` leaf which attributes the tag *young* to the mouse. We can have more than one `amod` dependency for each noun phrase. Another obvious source of tags is the noun itself. By taking the hypernym graphs of the *mouse* seen in 3.5(a) we see that it is a hyponym of an *animal*. Thus we can add *animal* to the list of tags.

"The old king was very arrogant"

"The king was an arrogant person"

Their parse trees are seen in Figures 3.7(a) and 3.7(b) respectively. In those figures we observe we can extract information in two ways:

1. By using the adjectival complement *arrogant* (`acomp`) of the verb *was* (Figure

3.7(a)). In this case we insert the text of the `acomp` dependencies to the character tags.

2. By using the `attr` dependency (Figure 3.7(a)). In this case we insert both the text of the `attr` dependency, as well as the text of its `amod` children dependencies.

We have to note that information about a character is not available in a single place but spread throughout the story. To compensate for that we gather information for each character by not only observing the attributes associated with its main noun phrase but for each of the character's pronouns as well. An example is given below:

"‘**The king** was very *old*. **He** was also *arrogant*"’

In this case, we have two sentences each giving a piece of information about the king. In the first sentence we understand that the king is very old, and in the second that he was very arrogant. An extra piece of information we get from the second sentence is that the king was ‘male’, as suggested from the pronoun *he*. We have the following tags for *king*:

$$\mathcal{T}_{\text{king}} = \{old, arrogant, male, king\} \tag{3.22}$$

After we construct the set of tags for each character, we check each of the tags whether it is a hyponym of a word that expresses a character from Table 2.2.3. An example list of such words and the properties they affect can be seen in Table 3-F. Those tags can aid us to look for actors to portray those characters. For example, the tag *animal* can help us focus our search to actors doing animal voices. We describe the algorithm in Algorithm 3. An example of the characters, tags, and associated properties in "Belling the cat" can be seen in Table 3-G.

3.3.9 Assigning Character Roles

Apart from character properties, we assign one of two roles to the characters in the story. A character can be a *main* character and the rest are *secondary* characters. Who

Tag	**Property**	**Example sentence**
`young.a.01` `old.a.01`	Age	An **old** mouse got up and said:
`male.n.02` `female.n.02`	Sex	The young **woman** was wise.
`aristocrat.n.01`	Upbringing	The **queen** was sitting on her throne.
`professional.n.01`	Occupation	Mary was a medical **doctor**.
`disputant.n.01`	Lifestyle	Mike is a **hippy**.
`person.n.01` `animal.n.01`	Species	The young **mouse** said.
`arrogant.a.01`	Other traits	The old king was **arrogant**.

Table 3-F: Example tag lemmas, character properties they affect, and sentences they appear in.

.

Character	**Tag/Pronoun**	**Property**
A young mouse	`young.a.01`	Young age
	`mouse.n.01`	Animal
	`he/PRON`	Male
An old mouse	`old.a.01`	Old age
	`mouse.n.01`	Animal

Table 3-G: Characters in "Belling the cat" and their properties.

is a main and who a secondary character is decided according to a character importance heuristic first described in [67]. According to [67], the assigned importance is:

$$I_c = \frac{|\mathcal{R}_c| \times (r_c^{(|\mathcal{R}_c|)} - r_c^{(1)})}{r_c^{(1)}} \tag{3.23}$$

where $|\mathcal{R}_c|$ is the number of referents of character c in the text, and $r^{(i)}$ the position of the i-th referent in the text.

3.3.10 Evaluation

Evaluation of the character extraction method was done on the task of extracting characters with voices, their age, sex, type of character (whether animal or person), and their

Algorithm 3 An algorithm for assigning tags to characters. Function f_{CR} is the coreference resolution function shown in Eq. 3.12. Function *root*(*sent*) gives the root node of the parse tree of sentence *sent*. Function *pos*(t) returns the part-of-speech of token t, *sense*(t) its sense identified with word sense disambiguation [75], *dependency*(t) its dependency label, *parent*(t) its parent token and *descendants*(t) the sub-tree of its children.

```
procedure ExtractTags($c$)                                    ▷ Extract tags for character $c$
    $\mathcal{T}_c \leftarrow \emptyset$
    $\mathcal{R}_c \leftarrow \{c\} \cup \{c' : f_{CR}(c') = c\}$              ▷ Referents of character $c$
    for $r \in \mathcal{R}_c$ do
        $\rho \leftarrow root(r)$                                 ▷ The root of the dependency tree of $r$
        $D \leftarrow descendants(\rho)$                          ▷ Children of $\rho$ in the dependency tree
        if $pos(\rho) \neq PROPN$ then                            ▷ Part-of-speech of $\rho$
            $\mathcal{T}_c \leftarrow \mathcal{T}_c \cup \{sense(\rho)\}$          ▷ Append sense of $\rho$ to tags
        for $d \in D$ do
            if $dependency(d) = amod$ then                        ▷ Dependency type of word $d$
                $\mathcal{T}_c \leftarrow \mathcal{T}_c \cup \{sense(d)\}$
        $p \leftarrow parent(\rho)$                               ▷ Parent of $\rho$ in the dependency tree
        if $pos(p) = VERB$ then
            $D' \leftarrow descendants(p) - \{r\}$
            for $d' \in D'$ do
                if $dependency(d') \in \{acomp, attr\}$ then
                    $\mathcal{T}_c \leftarrow \mathcal{T}_c \cup \{sense(d')\}$
                    if $dependency(d') = attr$ then
                        $D'' \leftarrow descendants(d')$
                        for $d'' \in D''$ do
                            if $dependency(d'') = amod$ then
                                $\mathcal{T}_c \leftarrow \mathcal{T}_c \cup \{sense(d'')\}$
    return $\mathcal{T}_c$
```

role in the story (main or secondary). Additionally, evaluation included the performance on assigning dialogue lines to characters as well as the verbs that signpost these dialogue lines.

From 249 Aesop fables scraped from the web[3], 166 were chosen that had at least one detected dialogue line (and thus can be associated with the character). Those were annotated automatically using the character extraction method described above and these annotations were manually corrected according to the following criteria:

1. Annotate correctly the dialogue lines if incorrectly assigned.

2. Annotate those words or phrases that signpost those dialogue lines (*speech act*

[3] `http://www.aesopfables.com`

Extraction Type	p	r	f_1	**support**
Character	0.85	0.88	0.86	100
Character [sex=male]	0.84	0.57	0.68	37
Character [sex=female]	0.92	0.48	0.63	12
Character [type=person]	0.82	0.73	0.77	50
Character [type=animal]	0.93	0.66	0.77	44
Character [age=young]	0.75	0.38	0.50	4
Character [age=old]	1.00	0.31	0.47	4
Character [role=main]	0.79	0.88	0.83	72
Character [role=secondary]	0.82	0.81	0.81	56
Speech act trigger	0.97	0.93	0.95	14
Dialogue line	0.99	0.89	0.94	310
Says what	0.88	0.81	0.84	314
Who speaks	0.77	0.74	0.75	257
Says	0.67	0.64	0.65	267

Table 3-H: Precision p, Recall r, f_1 scores and number of retrieved elements *support* in 166 short Aesop Fables. The first 11 rows refer to entities in the text and their attributes and the 3 last to relations. *Characters*, *speech act triggers*, and *dialogue lines* correspond to spans in the text denoting story characters, words signifying that someone speaks, and dialogue lines respectively. Attributes in brackets correspond to properties of the story characters (e.g. *characters [sex=male]* corresponds to male characters in the text). *Who speaks* links *characters* with *speech act triggers. Says what* links *speech act triggers* with *dialogue lines.* Those two binary relations are used to construct the ternary relation *says* which links *characters*, *speech act triggers*, and *dialogue lines*, and denotes speech acts in the text.

triggers). Leave blank if no such words or phrases exist.

3. If the system has already annotated a mention of a character saying a line, leave it as is. Otherwise, annotate as character the first instance of the character in the text.

4. Draw arcs from *characters* to *speech act triggers* and from speech act triggers to the dialogue lines to signify that a character *speaks* that line as signified by *speech act triggers.*

5. For each character, select the correct attributes:

   - For age, sex and type (animal or person), as derived from the context of the story.

- For its role (main or secondary character), as derived from the story or if in doubt, use the title of the story (e.g. in *"The Wily lion"* the main character is the lion. If the main character does not speak, assign every character as secondary.

The above corrections served as a 'gold' standard against which to evaluate the automatically annotated one. We report precision p, recall r and f_1 scores:

- *Precision*, usually denoted as p, is the ratio of the number of elements that are retrieved and are also found in ground truth over the number of retrieved elements.

$$p = \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{retrieved}|} \tag{3.24}$$

- *Recall*, usually denoted as r, is the ratio of the number of elements retrieved and found in ground truth, over the number of elements found in ground truth.

$$r = \frac{|\text{relevant} \cap \text{retrieved}|}{|\text{relevant}|} \tag{3.25}$$

item f_1 *score*, usually denoted simply as f_1, is the harmonic mean of p, and r and shows a "combination" of the two metrics:

$$f_1 = 2\frac{p \cdot r}{p + r} \tag{3.26}$$

Where 'relevant' are the items found in our gold annotations and 'retrieved' are the ones resulting from automatic annotation using our method. The values reported for the aforementioned setting for each extraction task can be seen in Table 3-H. We observe that tasks that depend on previous tasks report lower values. For example, since recognition of character attributes depend on the correct recognition of characters themselves, errors in character recognition will result in lower p, r and f_1 values for attribute extraction as well. In general we observed that our method suffered from the following errors:

1. Incorrect annotations in conjunctions: In some cases phrases such as:

   ... the bull and the lion ...

   are considered a single entity which needed to be corrected and thus resulting in annotation errors on behalf of our method. There are other cases however that such conjunctions are correct when followed by the pronoun *they* and a speech act.

2. The same character annotated with different types of pronouns: There are cases where a single character might be referred to with a neutral pronoun (it) and subsequently with a gendered one (he or she). This leads the link of character mentions to either break or misses assigning the sex attribute.

3. Capitalised mentions: There are cases where the stories are written with capitalised characters (e.g. *The Lion* which lead the tagging system in the NLP pipeline to tag them as proper names, and thus leads some characters to be assigned as *people* instead of animals.

4. Incorrect disambiguation: There are cases where the algorithm for word sense disambiguation assigned incorrect senses, e.g. `king.n.01` (as the head of state) instead of `king.n.04` for the preeminent member of a group. This led to cases where the word *king* such as in *the king of beasts* were assigned the former sense, and thus be regarded as a person, instead of an animal. In other cases the *Ass* (the animal) was either assigned as *American Samoa* or *Arsenic* which led to a failure of assigning any type attribute at all.

5. Speech act triggers are not always verbs: There are cases where a dialogue line is signified with the verb 'to be'. E.g.: *"The reply was"*.

6. Speech act triggers are not always communication verbs: Similarly to above, the triggers can be verbs that do not confer communication directly, e.g.: *"They made*

reply".

7. Speech acts for inner voices: We omitted to capture trigger verbs that signify inner voice, e.g. *think*.

8. Implied coreference: There are cases where two different nouns refer to the same entity, however, the link is implicit. E.g.:

   The woodcutter was [...] the man said: "...

Part of the mistakes reside in insufficient design on part of our algorithm, and part on the imperfections of NLP algorithms used. The main sources of NLP errors come from incorrect tagging (e.g. as proper names instead of capitalised nouns), incorrect word sense disambiguation (e.g. *the king of beasts* being a ruler of state) and incorrect coreference resolution as seen in the examples of errors above. From these, only the latter has been formally examined in the context of extracting information from stories [84].

3.4 Locations

An important element in a story that gets communicated in radio drama is the spatial locations of the story's constituents. Those can be locations of objects relative to other objects, or characters relative to other characters or environments as well as identification of the environments themselves. Consider the following excerpt[4]:

A fat Bull was feeding *in* **a meadow** when **a lion approached him**.

The elements given in bold in the excerpt above signify some part of a spatial relation between two entities. Those elements signify in a straightforward way that there *is* a bull in *a meadow* and that *a lion approaches* the bull. Prior knowledge also lets us understand that *a meadow* is a kind of outside environment and the verb *approach* means that the lion is getting closed to the bull. Understanding this kind of information when

[4] The excerpt is taken from the first sentences of "The Wily Lion" Aesop's fable, rewritten for brevity.

trying to adapt story text to radio drama is important. Knowing which parts constitute environments lets us choose relevant sound effects, EQ and reverberation settings to accompany the characters in the story. Furthermore, understanding relative positions helps us establish the acoustic hierarchy described in Section 2.6.1:

> The man tore his hair, and raised such an outcry that **all the neighbours** came *around* **him**.

In the above excerpt[5], a radio drama interpretation would have the sound hierarchy with the main character in the highest amplitude, panned to the centre, and emanation speech with lower amplitudes panned left and right. Therefore, it is useful to be able to recognise such spatial configurations

3.4.1 Spatial Role Labelling

Identification of the above elements and the spatial configurations they constitute is a challenging problem in information extraction from natural language. The task of *Spatial Role Labelling* pertains to extracting *spatial relations* in natural language sentences, each of which includes a *spatial indicator* and its *arguments* [7, 85]. Spatial Relations are relations that define the spatial configuration between two entities or objects. They can be categorised between *Static*, and *Dynamic* spatial relations. A static spatial relation defines a spatial configuration between two entities (the arguments) where both of the entities are fixed in space (they do not move). On the other hand, a dynamic spatial relation defines a spatial configuration between two entities where the first is moving in relation to the second. To demonstrate the difference between static and dynamic spatial relations, consider again the sentence:

> [**A fat Bull** was feeding *in* **a meadow**]$_1$ when [**a lion approached him**.]$_2$

The relation described in the part of text in $[\ldots]_1$ implies a *static* spatial relation since

[5] Adapted from the Aesop's Fable "The Miser".

the fat bull does not move relative to the meadow (it is always in it). In the one between $[\ldots]_2$ the lion is moving towards the bull and the spatial relation implied is therefore *dynamic*.

Both the static and dynamic spatial relations are indicated by a single word in the sentence. The preposition *in* indicates the static relation between *the bull* and *the meadow* and the verb *approaches* indicates the dynamic spatial relation between *the lion*, and *the bull*. Such indicators of spatial relations are called *Spatial Indicators*. For dynamic spatial relations, those are are called *Motion Indicators* instead. The arguments of a spatial relation may be placed before, after, or around the indicators. The names of these arguments vary depending on their role and the type of spatial relations they are part of. In this thesis, we use the nomenclature from [7]:

- The *Trajector* is the entity that actively participates in a spatial relation.
- The *Landmark* is the location or object the *trajector* is spatially associated with.

In the spatial relation found in the quotation above, the *trajector* refers to *the bull*, and the *landmark* refers to *the meadow*. Other roles for the arguments include the following:

- The *Path* contains the intermittent 'locations' that a trajector starts from, traverses or ends up during their motion.

An example of a path would be [6]:

> A [wagoner $]_{trajector}$ was once [driving $]_{motion\_indicator}$ a heavy load along [a very muddy way$]_{path}$

In the segment above, the *wagoner* is passing through the *very muddy way* in order to arrive to his destination.

- The *Distance* is an explicit or implicit distance given in the text.

[6] Aesop's Fable: "Hercules and the Wagoner"

i	**Argument 1**	**Indicator**	**Argument 2**	**Label**
1	a fat Bull	in	a meadow	Region
2	a lion	approached	him	Direction

Table 3-I: Spatial relations in "The Wily Lion"

An example of a distance relation would be:

$$\text{There was [a lake]}_{trajector}\text{ [not far away]}_{distance}$$

Distances such as 'near', 'far', '20 meters from' are very helpful in establishing the placement of actors and objects in a radio drama scene. For the rest of the text, we will use the shortcuts $tr, si, mi, lm, p, di, dr$ for *trajector*, *spatial indicator*, *motion indicator*, *landmark*, *path*, *distance*, and *direction.* We label a segment of text with one of the aforementioned labels by putting it into brackets with the label as a subscript as per the following example:

$$\text{[A fat bull]}_{tr}\text{ was feeding [in]}_{si}\text{ [a meadow]}_{lm}.$$

To refer to the spatial relations themselves we use a triplet notation:

$$\langle \text{A fat bull}_{tr}, \text{in}_{si}, \text{a meadow}_{lm} \rangle\text{\_Region} \tag{3.27}$$

An example of the spatial relations found in the first sentence of *"The Wily Lion"* can be seen in Table 3-I. The spatial relations are further labelled as *Region*, *Direction*, or *Distance*:

- The Region label is applied to spatial relations where the trajector is defined relative to the interior or exterior of the landmark, e.g. "The flower is *in* the vase".

- The Direction label is applied to spatial relations where the trajector is defined relative to the landmark as an external frame of reference, e.g. "The flag was *on top of* the building" or "The wagoner was *moving along* a path".

Relation Label	**Argument 1**	**Indicator**	**Argument 2**	**Type**
Region	Trajector	Spatial Indicator	Landmark	Static
Direction	Trajector	Spatial Indicator	Landmark	Static
Direction	Trajector	Motion Indicator	Landmark or Path	Dynamic
Distance	Trajector	Distance	Landmark	Static

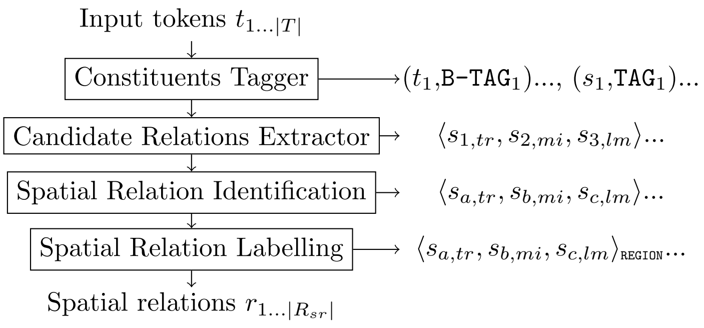
Table 3-J: Types of spatial relations identified in natural text.

- The Distance label is applied to spatial relations where the relation between the trajector and the landmark is given via a quantitative expression such as *near*, *12km*, etc. E.g.: “The kids are *close* to the blackboard”.

Static relations can be labelled as either *region*, *direction*, or *distance* while dynamic relations can only be labelled as *direction.* A summary of types of spatial relations, their labels and their arguments can be seen in Table 3-J. In the next section, we describe how we extract those arguments and relations automatically.

3.4.2 Automatic Spatial Role Labelling

Automatic labelling of spatial semantics has been the goal of past competitions for semantic parsing from natural text (SemEval) which led to the development of techniques to solve the problem [7–10, 86]. A notable such study [87] uses high recall heuristics to mine candidate static relation constituents (the *trajector*, *landmark*, and *spatial indicator* described above) and train a binary support vector machine classifier to identify such relations. [8] use a Hidden Markov Model (HMM) with SVM emissions together with shallow grammatical features and distributional semantics to identify both static and dynamic relation constituents, and an SVM classifier to assign them one of the labels described previously in Section 3.4.1. In [9], use of explicit features is avoided and instead the features for the spatial indicator and its arguments are learned using a combination of convolutional and simple MLP neural networks together with the Viterbi decoding algorithm. Finally, in [10] the authors use Bi-directional long short-term memory networks (BiLSTM) to label spatial relation constituents, together with a rule-based system to assign labels to spatial relations.



Input tokens $t_{1\ldots|T|}$
Constituents Tagger
$(t_1,\texttt{B-TAG}_1)\ldots, (s_1,\texttt{TAG}_1)\ldots$
Candidate Relations Extractor
$\langle s_{1,tr}, s_{2,mi}, s_{3,lm}\rangle\cdots$
Spatial Relation Identification
$\langle s_{a,tr}, s_{b,mi}, s_{c,lm}\rangle\cdots$
Spatial Relation Labelling
$\langle s_{a,tr}, s_{b,mi}, s_{c,lm}\rangle_{\texttt{REGION}}\cdots$
Spatial relations $r_{1\ldots|R_{sr}|}$

Figure 3.8: An overview of assigning SpRL labels to spans and relations

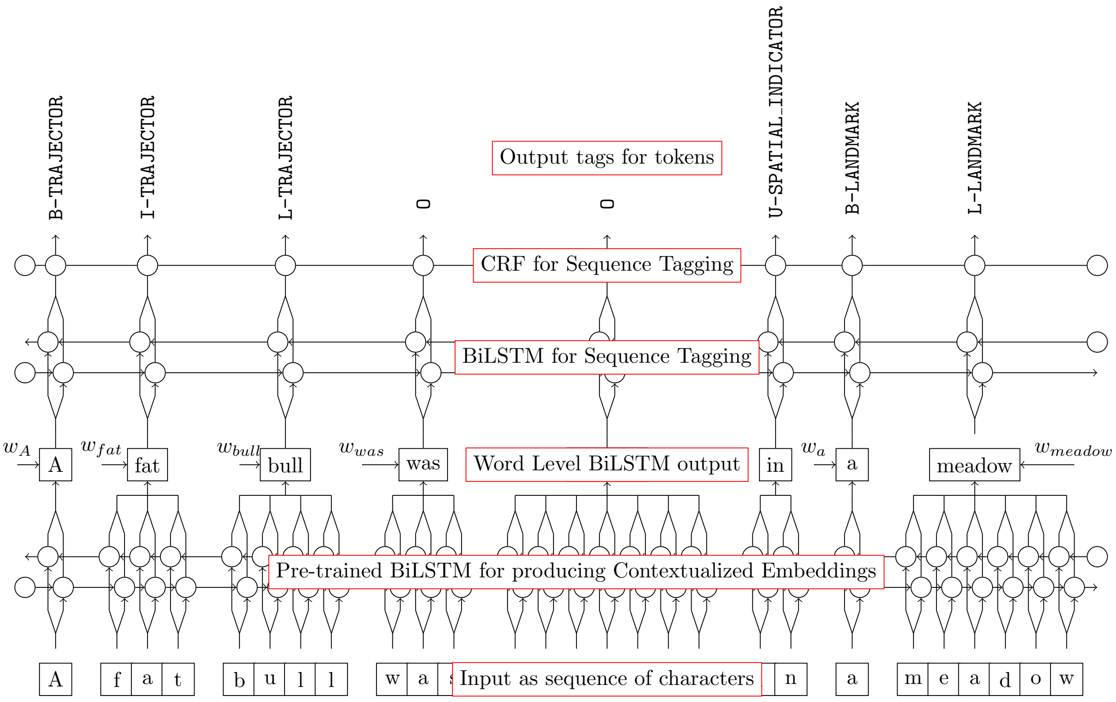


Figure 3.9: An overview of the model used for sequence tagging.

Our approach is similar to [10] in that it uses a BiLSTM to assign spatial constituent roles to spans of text, combines those constituents in each sentence in order to create candidate spatial relations and then uses a cascade of two decision tree classifiers: the first to classify whether a relation is spatial or not, and one to assign a label. An overview of the method can be seen in Figure 3.8. Below we explain each step in more detail:

1. **Input** As input, we provide the unmodified text we want to extract spatial relations from, as a sequence of ASCII characters.

Label	**Description**
Features relating to the trigger	
trigger.type	Spatial label of trigger (e.g. *si*, *si*, *di*...).
trigger.length	Number of tokens in trigger.
trigger[n].text	Raw text for the $n-$th token in the trigger.
trigger[n].lemma	Lemma for the $n-$th token in the trigger.
trigger[n].pos	Part-of-speech for the $n-$th token in the trigger.
trigger[n].dep	Dependency with the token head for the $n-$th token.
trigger[n].entity_type	Entity type for the $n-$th token (e.g. PERSON, GPE).
trigger.text.bigrams[n]	The $n-$th raw text bigram in the trigger span.
[trigger.text == W]	Bag of words for token W in the trigger.
[trigger.dep == D]	Bag of words for dependency type D in the trigger.
Features relating to each of the two arguments	
argN.length	Number of tokens of the N-th spatial argument.
argN[n].text	Raw text for the $n-$th token in the N-th spatial argument.
argN[n].lemma	Lemma for the $n-$th token in the N-th spatial argument.
argN[n].dep	Dependency to its head for the $n-$th token.
argN[n].entity_type	Entity type for the $n-$th token.
argN[n].null?	True if argument N is empty, else False.
argN-trigger[n].path[i]	Step *i* of the dependency path in the tree in order to reach the trigger (e.g. ↑ nsubj)
argN-trigger.distance	Distance (in number of tokens) to the trigger
argN-trigger.position	Left or Right depending on the position of the N-th argument relative to the trigger.
argN-trigger.#SE	Number of other spatial elements between the argument and the trigger.
arg1-arg2.path[i]	Step *i* of the dependency path in the tree in order to reach arg2 from arg1
arg1-arg2.distance	Distance (in number of tokens) between arg1 and arg2

Table 3-K: Grammar features for a candidate relation $r_c = \langle \texttt{arg1}, \texttt{trigger}, \texttt{arg2} \rangle$. There is a large overlap with the features used in [6] which were used as a starting point.

2. **Constituents Tagger** Spatial relation constituents are sequences of word-tokens. To identify a span in an entity recognition task, each individual token of the span must be identified correctly together with its position in the span. For example, in:

   A bull feeding in a meadow.

   the tokens A and bull both belong to the span A bull. If we want to tag this span as e.g. a Trajector we tag both *A* and *bull* with the label *Trajector* in addition to a label signifying their role in the span (e.g if they begin the span, are inside of it, or are outside the span). These latter labels are used as prefixes to the overall span label. In our case, we use the BILOU scheme which has been shown to perform well in entity recognition tasks [88]. The BILOU scheme prefixes the span label with the following single letter prefixes:

- `B` – Signifies that the word-token starts a new span.
- `I` – Prefixes token tags that belong to the same span that have been previously begun using the `B` prefix.
- `L` – Signifies that the token ends the span.
- `U` – Signifies that span comprises of a single token.
- `O` – Signifies that the token does not belong to any span.

An example of such annotation can be seen in Table 3-L. The actual tagging procedure uses a Bi-directional long short term memory (BiLSTM) followed by a conditional random field (CRF) [89] with contextualised word embeddings based on the sequence of characters in each word [90]. The word embeddings used were part of the pre-trained embedding models available with the FLAIR NLP framework [91]. We encourage the reader to read in the aforementioned papers for details on how contextualised embeddings and sequence tagging with BiLSTM-CRF models work. A quick summary of the steps is given below:

(a) The sentence to be parsed is given to the contextualised embeddings BiLSTM as a sequence of characters. The output is a vector for each word in the sentence that represents that word based on its context in the sentence. In our case this process happens two times, using the pre-trained embedding models `news-forward` and `news-backward` available in the FLAIR database of pre-trained models[7].

[7] The models `news-forward` and `news-backward` are trained on the same dataset of 1 billion words,

(b) The outputs from the two pre-trained BiLSTMs for each word as well as their GloVe [92] vector representation are stacked together to create a very ‘tall’ word embedding vector.

(c) This resulting embedding vector is passed through the BiLSTM-CRF model which assigns a spatial tag (e.g. `B-TRAJECTOR`) for each word. The parameters of this BiLSTM-CRF model are learned from examples taken from a corpus of annotated word-tokens.

A visual overview of this tagging process is presented in Figure 3.9. To “learn” the parameters of the top BiLSTM-CRF model we use the unmodified dataset originally given as part of the competition in [7].

3. **Candidate relation extraction**

   For each combination of spatial elements found above we construct a candidate relation $r_c = \langle tr, t, a_2 \rangle$ where tr is an identified trajector, t is a trigger (motion indicator mi, spatial indicator si or distance di) and a_2 is the second argument of the relation, which can be a landmark lm, a path p, or a a direction dr. To limit the number of candidate relations, we consider only combinations of elements which exist in the same sentence.

4. **Spatial Relation Identification**

   In this step, features are extracted from every candidate relation r_c according to the features seen in Table 3-K. This step creates a feature vector $\mathbf{f}_c$ for every relation r_c. A simple binary decision tree classifier then labels every relation r_c into *valid* or *invalid* according to whether the candidate relation is a valid spatial relation or not.

5. **Spatial Relation Labelling**

   In this step, the set of valid spatial relations identified in the previous step, is

with the only difference being that during training, `news-forward` observes the text sequentially ‘left to right’, while `news-backward` ‘right to left’.

Word	Tags	Span
`A`	`B-TRAJECTOR`	[A fat bull]$_{tr}$
`fat`	`I-TRAJECTOR`	
`bull`	`L-TRAJECTOR`	
`was`	`O`	
`feeding`	`O`	
`in`	`U-SPATIAL_INDICATOR`	[in]$_{si}$
`a`	`B-LANDMARK`	[a meadow]$_{lm}$
`meadow`	`L-LANDMARK`	
...	...	

Table 3-L: Example of automatic annotations for “A fat bull was feeding in a meadow”. The raw text can be seen in the left column. In the middle column, there are the associated tags for each word using the BILOU tagging scheme. The last column shows the spans each of the tagged word belongs to.

	IAPR TC-12		**Confluence**	
Statistic	**Training**	**Testing**	**Training**	**Testing**
Files	1	1	95	22
Sentences	600	613	1422	367
Trajectors	716	872	1701	497
Landmarks	661	743	1037	316
Sp. Indicators	670	796	879	247
M. Indicators	–	–	1039	305
Paths	–	–	945	240
Directions	–	–	223	37
Distances	–	–	307	87
Relations	765	940	2105	598

Table 3-M: Statistics for the corpora available in the SemEval 2013 Task 3 (SpRL) competition. The statistics are taken from [7, Table 1].

labelled using a decision tree classifier using one of the three labels: `Region`, *Direction*, or `Distance`, each corresponding to a spatial relation type discussed in Section 3.4.1.

3.4.3 Training the models

In the steps given in Figure 3.8 we included the use of three learned models: a Constituents tagger based on a BiLSTM-CRF, a Decision Tree classifier for identifying spatial relations, and a second Decision Tree Classifier for assigning labels to those relations.

The parameters for these models are learned from examples provided in the training data of the original SpRL competition [7]. There were two corpora provided in the competition: a subset of the IAPR TC-12 image description dataset [93], and descriptions of locations form the Confluence[8] project. The first corpus contains simple sentence descriptions of spatial configurations of objects found in images such as:

"About 20 kids in traditional clothing and hats waiting on stairs."

The corpus based on the Confluence project contains annotated paragraphs taken from a recollection of journeys to several locations around the world such as:

> "Travelling without an atlas, I relied solely on my receiver and wound my way through the Florida countryside, past Ebro Greyhound Park, and finally arrived within 1/4 mile of my goal."

Coincidentally, the formats of the two corpora, while dissimilar to one another, can be found complementing each other in text stories. Story sentences such as:

A fat bull was feeding in a meadow.

imply (mental) image descriptions similar to the ones found in IAPR TC-12, while story sentences such as[9]:

"A carter was driving a wagon along a country lane, when the wheels
sank down deep into a rut."

are more similar to the event-recollection style text contained in the Confluence project corpus. A model therefore trained on both will be able to extract almost the entirety of spatial relations found in story text. The BiLSTM-CRF model for sequence tagging and the two decision tree classifiers are therefore trained to the joint IAPR TC-12–Confluence corpus. Training was done as follows:

[8] Degree Confluence: `http://confluence.org/`

[9] Another version of Aesop's Fable: "Hercules and the Wagoner"

Parameter	**Value**
LSTM hidden size	128
# hidden LSTM layers	1
Dropout	0.259
Learning rate (starting)	0.1
Epochs	67

Table 3-N: Training parameters for the BiLSTM-CRF model training. The model was trained with an adaptive learning rate which halved every 5 epochs with non-improving loss and early stopping when the learning rate fell to negligible levels.

Dataset	**Task**	p	r	f_1	f_1 [8]	f_1 [9]	f_1 [10]	**Class**
IAPR TC-12	A	0.936	0.950	**0.943**	0.926	–	0.901	Sp. Indicator
		0.861	0.915	**0.887**	0.785	–	0.814	Landmark
		0.576	0.828	0.679	0.682	–	**0.823**	Trajector
	B	0.607	0.522	0.561	0.458	**0.702**	0.562	Relation
	E	0.561	0.350	0.431	–	–	–	Direction
		0.612	0.661	0.636	–	–	–	Region
		0.167	0.125	0.143	–	–	–	Distance
Confluence	A	0.755	0.481	**0.587**	0.538	–	–	Sp. Indicator
		0.729	0.443	0.551	**0.554**	–	–	Landmark
		0.753	0.472	**0.580**	0.406	–	–	Trajector
		0.536	0.353	0.426	–	–	–	Relation
	C	0.755	0.481	**0.587**	0.536	–	–	Sp. Indicator
		0.729	0.443	0.551	**0.554**	–	–	Landmark
		0.753	0.472	**0.580**	0.406	–	–	Trajector
		0.665	0.766	**0.712**	0.427	–	–	Path
		0.832	0.645	**0.727**	0.443	–	–	M. Indicator
		0.528	0.411	**0.462**	0.264	–	–	Direction
		0.887	0.607	**0.721**	0.490	–	–	Distance
	D	0.151	0.088	0.111	–	**0.463**	–	Relation
	E	0.004	0.003	0.004	–	–	–	Direction
		0.484	0.278	0.353	–	–	–	Region
		0.000	0.000	0.000	–	–	–	Distance

Table 3-O: Results for our method of Automatic Spatial Role Labelling compared to [8–10]. It is worth noting that [9] reported finding errors and fixing them in the original dataset and therefore it is not appropriate to directly compare with the other methods. [8] submitted a different model for each of the training/testing corpora while our method has been trained to both the corpora. We could not find information about which part of testing corpus is being reported in [10] but inferred that it is the IAPR TC-12 based on the comparison the authors report with [8] and their lack of reporting on dynamic spatial relation constituents such as `MOTION_INDICATOR`.

1. Convert all of the corpus `.xml` files to the Co-NLL 2002 shared task format [94] for named entity recognition. The only difference is the use of the BILOU tagging scheme instead of the proposed BIO [88].

2. Train a sequence tagging model using the FLAIR framework for NLP [91]. Contextualised embeddings are extracted for each word in the sentence and are stacked with GloVe vectors as described in Section 3.4.2. Pairs of embeddings and BILOU tags are used to train the sequence tagging BiLSTM-CRF model. Training statistics and hyperparameters chosen were found using the hyper-parameter optimisation package Hyperopt [95] and are found in Table 3-N.

3. Sets of candidate spatial relations were created for every sentence in the training corpus by combining the spatial constituents relevant to that sentence. The relations that were found in the corpus to be valid were then labelled using their respective label (e.g. `REGION`, `DIRECTION`, ...). This set was used to train two Decision tree classifiers: a binary one that classified the candidate relations as `VALID` or `INVALID` and one that classified the relations classified as `VALID` to one of the spatial labels.

Training was done explicitly on the joint training set of the two corpora and testing to each of the two corpora separately. Evaluation was done using the official competition's java evaluation scripts in the "relaxed" mode. We report metrics from the test corpora in Table 3-O. For the sake of completion, we also report f_1 scores for three more works reported on the same datasets.

3.4.4 Conclusion

We observe that the method we used, despite using learned data from both the corpora in the SemEval competition outperforms, or at least performs similarly with the state of the art, at least when comparing f_1 scores. While the reported f_1 scores of relations seem low and are almost zero in the case of `DISTANCE` relations, we have empirically observed that our method adequately captures spatial relations in the Aesop's Fables dataset we used for extracting characters. Furthermore, the results in the work of [9] suggest that improving those metrics also depend on improving the corpora used for training. Finally, we developed our method in a free to use python library based on Spacy and the Flair framework[10] and we encourage the reader to try it for themselves.

3.5 Emotions

Emotions are inherent and omnipresent in human activity and as something generally understood intuitively hard to quantify and give a formal description. A common definition, however, describes emotion as “a response to events that are important to us” [96]. In studies pertaining to analysing text, or even sound media, *Emotion* is usually used interchangeably with *Mood*, which is a longer, less powerful affective phenomenon that is described in a two-dimensional energy-tension coordinate axis. On the other hand, *Sentiment* is a personal belief that is not founded on proof or certainty [97]. Understanding and detecting emotion in story texts serves two main purposes:

1. Understanding the mental and emotional states of characters in the story.

2. Understanding the overall emotional theme the author intended to convey to the reader.

Both the above functions are important when adapting story texts to radio dramas. The two functions identified above directly contribute to the following:

1. Direct the speech of actors portraying the characters by using suggestions such as ‘with contempt’, ‘in an angry voice’, etc when writing the radio drama script.

2. Direct the speech of the narrator according to the implied emotion of the text passage they are narrating.

3. Choose appropriate intro-outro music according to the emotion at the beginning or ending of the story, as well as linking music (Section 2.5). This can be done by

[10] SPRL-SPACY: `https://github.com/mmxgn/sprl-spacy`

extracting a set of emotional tags to search for music, or by using *mood* dimensions that correspond to *arousal* and *valence.*

3.6 Extracting emotions from text

Extracting emotions from text is a thoroughly studied subject and as a task it can be thought of as a subset of Sentiment Analysis [98]. Usually the task consists of identifying conveyed emotion in text passages of varying length and granularity (from the emotion communicated using simple words, to the emotion underlying the whole story text). The most popular form of the task assigns one of 7 basic emotions: *Anger*, *Disgust*, *Fear*, *Guilt*, *Joy*, *Sadness*, or *Shame* to a passage of text with granularity usually of a sentence although there are exceptions to this rule.

There has been a large amount of studies focused on automating emotion extraction using a variety of techniques from simple words-emotions association lexicons to using databases of common sense knowledge and statistical methods. Before we refer to them we provide some examples to better understand the challenges of the task [99]. The authors in that study provide some examples for the challenges in emotion extraction from text. Consider the following dialogue line:

"I am happy"

This directly communicates the emotion of *joy.* An emotion extraction system could easily assign this emotion to said character by simply consulting an emotional lexicon that associates the word *happy* with the emotion *joy.* Such lexicon-based methods can be found, for example, in [100] and are quire popular in literature [98]. The statement, however, can be negated:

"I am *not* happy"

Negation automatically complicates things for lexicon-based approaches [101], since now the appropriate assigned emotion is the opposite of 'joy': 'sadness'. Understanding and extracting such emotions needs a system to understand such cases. A simple rule based system to augment a lexicon with identifying word negations would suffice. Consider, however, the following phrase:

“I am going to the party”

The sentence above would be also normally marked as ‘joy’ (unless the speaker does not like parties). However, a lexicon based approach, even augmented with a rule-based system to detect opposites would not work because there is no word to convey emotion in this case. Understanding and conveying emotion now requires commonsense knowledge. The above example can also be part of a bigger context:

“I am going to the party”, she said, “although I need to study”

which immediately transforms the underlying emotion from ‘joy’ to ‘guilt’. It is apparent that extraction of emotions needs to also take into consideration the context of the phrase that is expressed. A different challenge described in [101] is that emotion recognition systems must be sensitive to the type of the text they perform their analysis to. As an example, when examining modern electronic communications in social media (e.g. in discussion fora) we observe use of emoticons, emojis, usage of capitalisation, abbreviations and other not grammatically relevant cues that are usually absent from story text. In short, we have gathered the following challenges that pertain to emotion recognition from text [98]:

1. Emotion extraction is a context-sensitive problem [102].
2. Emotions are not always expressed with affective words (such as happy, joyful) but by describing real-world situations. [99].
3. Analysing text for emotion extraction depends on the type of the text that is analysed [101].

We referred to three main approaches to emotion recognition: *Lexicon-based*, *Rule-based*

Reference	**Type**	**Approach**	**Data used**	**Emotions**
[103]	ED	Statistical	Children stories	7
[104]	ED	Rules	Fairy Tales	7
[105]	ED, AD	Lexicon, KB, Rules	Fairy Tales	95 (+3 continuous)
[106]	ED	Lexicon, Rules	Personal experiences	7
[100]	ED, AD	Lexicon	Novels, Fairy Tales	8
[107]	AD	Lexicon, KB	Fairy Tales	95 (+3 continuous)
[108]	ED	Statistical, KB	Fairy Tales	5

Table 3-P: Relevant studies for extracting emotion from stories. Type refers to the task they contributed. ED stands for *Emotion Detection* and AD for *Annotated Dataset*. KB stands for *Knowledge base*. The column *Emotions* refers to the number of emotions examined in the study.

and *Commonsense knowledge-based*. Studies have also used *Statistical* approaches, such as Machine Learning, by trying to automatically extract emotional associations in passages of text in an annotated corpus [97]. In general, it has been shown that a combination of the aforementioned approaches tackles the problem the best. The reader can find a comprehensive analysis of the relevant studies in [98, 107]. In Table 3-P we present a summary of studies relevant to story text.

3.6.1 Extracting Emotions for Radio Drama Production

Before we attempt emotion extraction we need to choose an appropriate method from the ones in Table 3-P as well as an appropriate dataset. We focus only on methods which provide emotional categories (as opposed to continuous emotional values), and methods that have been used in children's tales. The focus on emotional categories can be used as tags for retrieving appropriate music for intro/outro and scene changes. The focus on children's stories is due to a better match with the dataset of Aesop's Fables. We use the method described in [100]. This method assigns emotions $e \in E$ in text T using the following function:

$$f_{emotions}(T, e) = \frac{1}{|T|} \sum_{w \in T} D(w, e) \tag{3.28}$$

$$D_{emotions}(w, e) = \begin{cases} 1 & \text{if word } w \text{ is associated with emotion } e \\ 0 & \text{otherwise} \end{cases} \tag{3.29}$$

where $D_{emotions}$ is a dictionary provided by [100] that associates a word w to emotion e. We will refer to Eq. 3.29 as the text's *emotional word density* for emotion e. The overall *emotion* of text is then given by:

$$f_{emotion}(T) = \arg\max_{e \in E} \{f_{emotions}(T, e)\} \tag{3.30}$$

In our dataset of Aesop Fables, we consider extraction of emotional tags from text in four different granularities: Word level, Sentence level, Dialogue Line level, and whole text level. We mentioned above some of the functions that emotion extraction serves, such as choosing appropriate intro, outro, or linking music as well as directing the speech of actors. For these purposes, we use the aforementioned emotion extraction method to extract three different kinds of emotion values:

1. Emotions across the whole story text.
2. Emotions of the first and last sentences.
3. Emotions of each dialogue line.
4. Emotions outside the dialogue lines.

In Tables 3-Q, 3-R, and 3-S we observe the first emotion tags for each of the above cases in the first 10 fables in the dataset. We can use the associated emotions as tags. For example, in the following dialogue line:

"I feel depressed", said Mary.

Name	**ang.**	**ant.**	**disg.**	**fear**	**joy**	**sad.**	**surp.**	**trust**	**emot.**
The Dogs and the Fox	0.00	0.20	0.00	0.40	0.20	0.00	0.00	0.20	fear
The Man and the Lion	0.12	0.12	0.00	0.50	0.12	0.00	0.00	0.12	fear
The Cat and the Birds	0.10	0.14	0.05	0.19	0.10	0.14	0.10	0.19	fear
The Cobbler Turned Doctor	0.14	0.14	0.09	0.11	0.05	0.18	0.05	0.25	trust
The Farmer and the Stork	0.12	0.08	0.00	0.17	0.21	0.21	0.04	0.17	joy
The Cat and the Rooster	0.40	0.20	0.00	0.20	0.00	0.00	0.00	0.20	anger
The Boy and the Filberts	0.14	0.00	0.29	0.07	0.07	0.36	0.07	0.00	sad.
The Gnat and the Lion	0.13	0.13	0.13	0.37	0.05	0.11	0.00	0.08	fear
The Heifer and the Ox	0.06	0.24	0.06	0.00	0.24	0.06	0.18	0.18	ant.
The One Eyed Doe	0.00	0.19	0.11	0.26	0.11	0.22	0.04	0.07	fear

Table 3-Q: Emotion association percentages for the first 10 fables in our dataset.

Name	**ang.**	**ant.**	**disg.**	**fear**	**joy**	**sad.**	**surp.**	**trust**	**emot.**
The Dogs and the Fox	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	fear
The Man and the Lion	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	fear
The Cat and the Birds	0.17	0.00	0.00	0.50	0.00	0.17	0.00	0.17	fear
The Cobbler Turned Doctor	0.17	0.00	0.17	0.17	0.00	0.33	0.00	0.17	sad.
The Farmer and the Stork	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	ang.
The Cat and the Rooster	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	ang.
The Boy and the Filberts	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	disg.
The Gnat and the Lion	0.33	0.00	0.00	0.67	0.00	0.00	0.00	0.00	fear
The Heifer and the Ox	0.12	0.25	0.12	0.00	0.12	0.12	0.12	0.12	ant.
The One Eyed Doe	0.00	0.20	0.00	0.20	0.20	0.00	0.20	0.20	ant.

Table 3-R: Emotion association percentages for the first sentence for the first 10 fables in our dataset.

Name	**ang.**	**ant.**	**disg.**	**fear**	**joy**	**sad.**	**surp.**	**trust**	**emot.**
The Dogs and the Fox	0.00	0.25	0.00	0.25	0.25	0.00	0.00	0.25	ant.
The Man and the Lion	0.00	0.33	0.00	0.33	0.33	0.00	0.00	0.00	ant.
The Cat and the Birds	0.00	0.25	0.00	0.00	0.12	0.12	0.25	0.25	ant.
The Cobbler Turned Doctor	0.25	0.00	0.00	0.00	0.00	0.25	0.00	0.50	trust
The Farmer and the Stork	0.12	0.08	0.00	0.17	0.21	0.21	0.04	0.17	joy
The Cat and the Rooster	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	ang.
The Boy and the Filberts	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	joy
The Gnat and the Lion	0.16	0.05	0.11	0.37	0.05	0.11	0.00	0.16	fear
The Heifer and the Ox	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	ang.
The One Eyed Doe	0.00	0.14	0.29	0.29	0.00	0.29	0.00	0.00	disg.

Table 3-S: Emotion association percentages for the dialogue lines for the first 10 fables in our dataset.

The feeling of `sadness` is prevalent, so we add it to the list of tags of the dialogue line:

$$\mathcal{T}_{\text{I feel depressed}} \leftarrow \mathcal{T}_{\text{I feel depressed}} \cup \{sad\} \tag{3.31}$$

In the dialogue line above, the feeling of sadness is dominant in the content. There are cases, however, where the content of the dialogue lines cannot be associated with emotion, but the emotion is given outside the dialogue line itself. Consider the following quotations:

> "You cannot be serious", Mary said with contempt. "You are not serious", Mary said angrily.

In such cases, the emotion is given by the modifier of the speech act trigger (e.g. *said*). We add this modifier as a tag as follows:

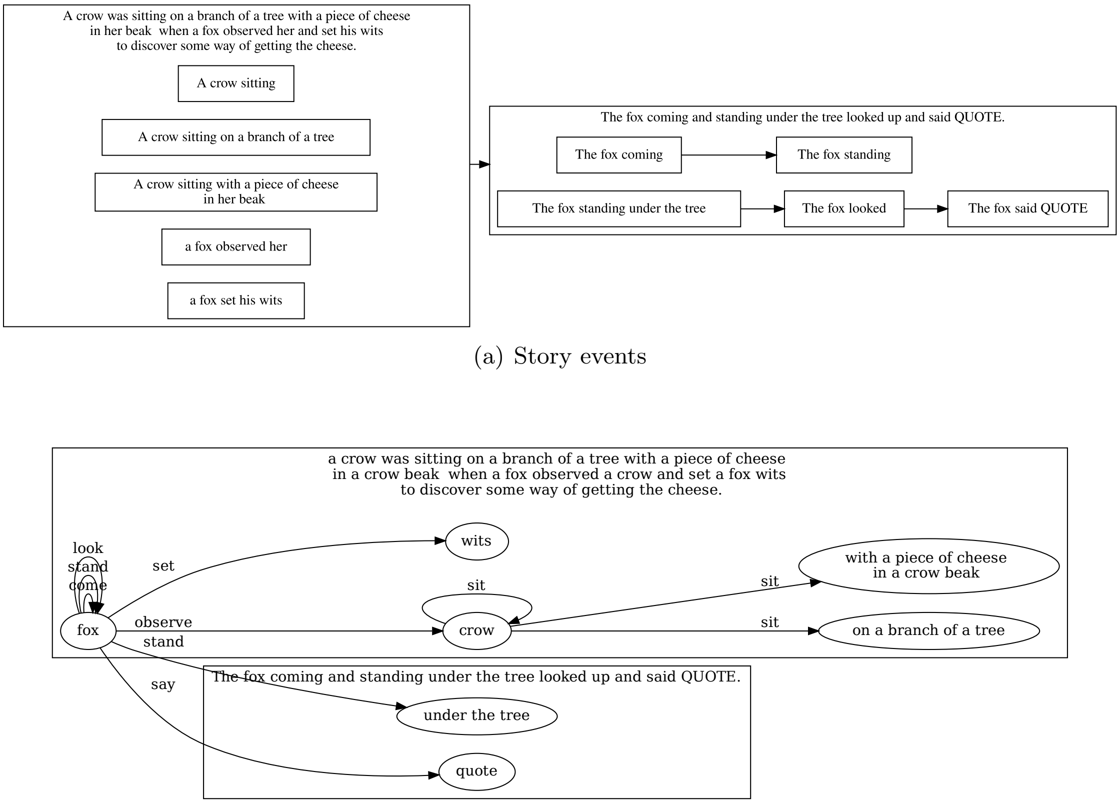
1. We analyse the sentences to their dependency tree structure similar to Section 3.3.6.

2. If the speech act trigger (said) is followed by a preposition (with) then we check whether the following noun is a hyponym of `emotional_arousal.n.01`, `emotion.n.01`, or `feeling.n.01`. In that case, and if the preposition is negated (e.g. *not with disgust*, or *without disgust*) we add it as a tag to the dialogue line.

3. If the speech act trigger is modified by an adverbial (e.g. *angrily*) we convert this adverbial to its corresponding noun (e.g. *anger*) using WordNet [62] and we proceed as in step 2.

3.6.2 Conclusion

Emotion extraction from text is a well-studied problem. For this reason, we did not put effort in developing new methods but we decided to depend on the already established portfolio of methods and showcase how emotion extraction can be used in the context of producing in a radio drama from story text. While we did not do an evaluation


A crow was sitting on a branch of a tree with a piece of cheese in her beak when a fox observed her and set his wits to discover some way of getting the cheese.
A crow sitting
A crow sitting on a branch of a tree
A crow sitting with a piece of cheese in her beak
a fox observed her
a fox set his wits
The fox coming and standing under the tree looked up and said QUOTE.
The fox coming
The fox standing
The fox standing under the tree
The fox looked
The fox said QUOTE
a crow was sitting on a branch of a tree with a piece of cheese in a crow beak when a fox observed a crow and set a fox wits to discover some way of getting the cheese.
look
stand
come
set
wits
sit
sit
with a piece of cheese in a crow beak
fox
observe
stand
crow
sit
on a branch of a tree
say
The fox coming and standing under the tree looked up and said QUOTE.
under the tree
quote

(a) Story events

(b) Story setting

Figure 3.10: Story events and setting approximation for the first two sentences of "The Crow and the Fox". In the top figure, smaller boxes represent simple events and arrows progression of events. Events that are grouped into larger boxes take place at the same time. In the bottom figure, arrows represent predicate directions similar to Figure 2.7(b). In both figures, the larger group boxes are labelled based on the sentence they represent.

of emotion extraction in our test set, evaluation of emotion extraction methods in the context of fairy tales can be found in past literature [105]. We revisit emotions in Section 5.4 where we discuss how we retrieve music that adheres to the emotional theme of the story.

3.7 Event extraction via Text Simplification

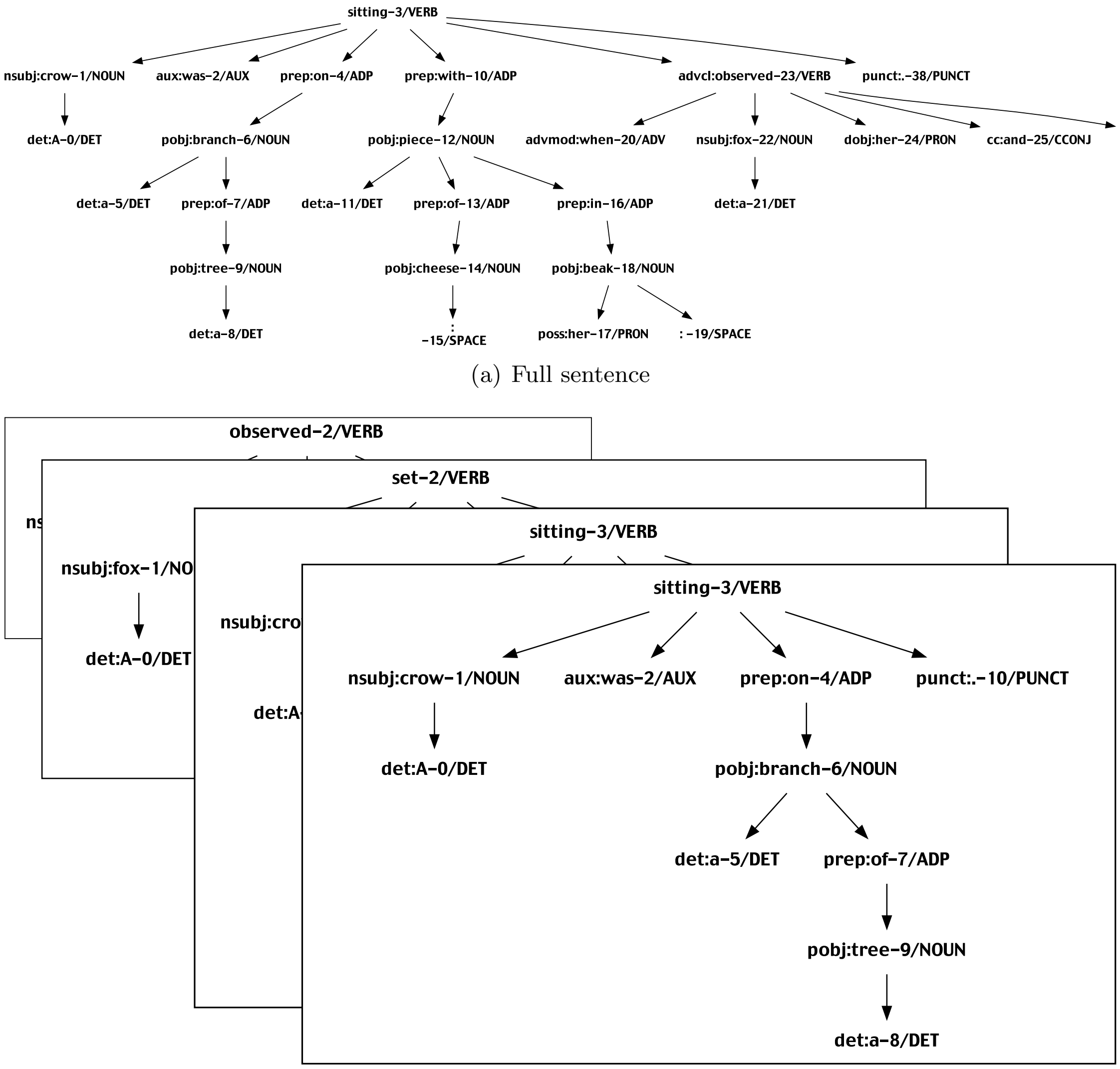
In the above sections, we presented elements that can be simply extracted by looking at a 'surface' level of the discourse text. There are elements, however, that cannot be extracted simply by looking at the discourse text and need a slightly deeper inspection of the story. An example is recognising when a character is performing an action that might lead to a sound being played (Section 2.6). The aforementioned tasks require at least some knowledge of the story world (see Section 2.2.2). Efforts that attempt at extracting the narrative structure (the *events* and *setting* part of the storyworld) can be found in [67], where the authors derive *Abstract Meaning Representations* [43] of story sentences and try to extract knowledge that is unable to be discerned by a shallow analysis of text, such as character roles, motivations, etc. While following a similar approach would help us find methods that take into consideration the happenings in the story world, AMR loses morphology information about the text, something that is useful when retrieving sound effects from story sentences (Chapter 5). Instead, we will be using what is called automatic *text simplification* (TS) for reducing the story text into simple sentences that can be easily processed using simple rules. TS is the process of modifying natural language to reduce its complexity and improve both readability and understandability [109]. In order to comprehend what this means consider the following example from Section 2.2.2:

> A Crow was sitting on a branch of a tree with a piece of cheese in her beak when a Fox observed her and set his wits to work to discover some way of getting the cheese.

The sentence above has a complicated dependency parse tree (Figure 3.11(a)) which makes it difficult to derive simple rules that extract e.g. the actions of the fox and the crow. By using automatic text simplification, however, we can simplify it to the following sentences:

> A crow was sitting on a branch of a tree. A crow was sitting with a piece of cheese in her beak. A fox observed the crow. A fox set his wits to work to discover some way of getting the cheese.

These sentences have much simpler dependency parse trees. Additionally, the main verb is at the root of each sentence (e.g. *sitting* or *observed*) is at the root of each parse



(a) Full sentence

(b) Simplified sentence

Figure 3.11: Parse tree of the first sentence in "The Crow and the Fox". There is a subtree of 11 more leaves on the right of the full sentence parse tree that was omitted due space constraints.

tree and each tree describes an action performed by someone in the story (denoted with `nsubj`).

Automatic text simplification has been used in the past of simplifying text for people with aphasia, knowledge elicitation from natural language sources, simple-wiki construction for foreign speakers, drug-discovery, etc [109]. For our purpose we use ClausIE [58] which jointly achieves text simplification and information extraction. ClausIE relies on a simple algorithm that processes the dependency parse tree of a more complicated sentence and extracts simpler clauses with a single verb and a single subject. It then proceeds to categorise each clause (e.g. as a *di-transitive* or *mono-transitive*). This categorisation process adds information about which parts of the clause are important and which can be omitted when simplifying the text.

The steps to simplify a sentence as a list of word-tokens are given as a flowchart in [58, Fig. 2] which is repeated in Figure 3.12 for convenience. Applying those to "The Crow and The Fox" results in the following seven clauses:

$$\langle \text{A crow}, \text{sitting}, \{\text{on a branch of a tree}, \text{with a piece of cheese in her beak}\}\rangle_{\text{SVA}} \tag{3.32}$$

$$\langle \text{a fox}, \text{observed}, \text{her}\rangle_{\text{SVO}} \tag{3.33}$$

$$\langle \text{a fox}, \text{set}, \text{his wits}\rangle_{\text{SVO}} \tag{3.34}$$

We observe that each clause might have multiple adverbials. While not shown, this is also the case for subjects, objects, and complements. It can, however, have a single verb. We can use Algorithm 4 to construct simple facts such as:

A crow sitting.

A crow sitting on a branch of a tree.

A crow sitting with a piece of cheese in her beak.

A fox observed her.

A fox set his wits.

Each sentence above has a single verb and subject. This will come handy in Section 5.2 when we discuss about a grammar-informed sound effects retrieval system.

The above methods allow us to analyse a single sentence in simple facts but do not tell us about their temporal progression (e.g. when these happen). Such temporal analysis has been the subject of previous SemEval Competitions [110, 111], however, doing a thorough analysis for event progression is outside of the scope of this thesis. Instead,

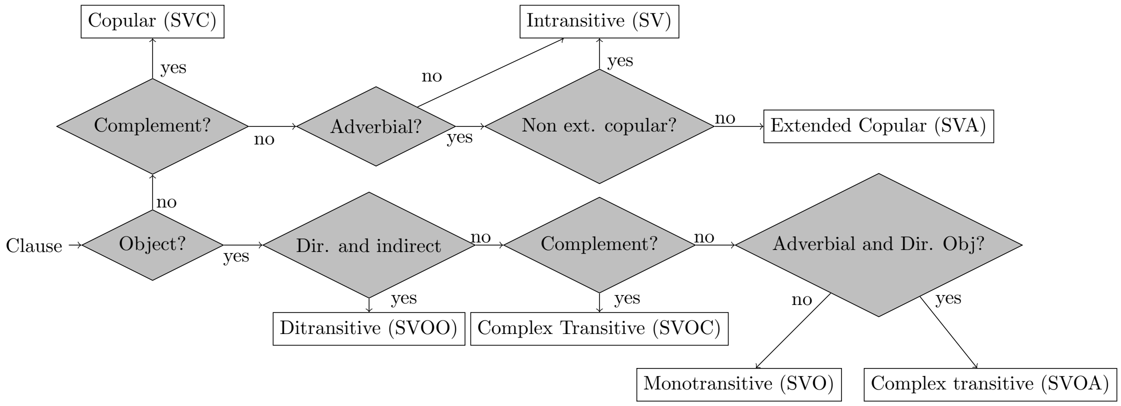


Figure 3.12: Flowchart of the process labelling extracted clauses in ClausIE.

we make the following assumption: *"Events happen in time in the same order they are written in the text"*. While this assumption does not hold in the general case, we found that it is usually the case in our dataset of Aesop Fables. Using the methods described above we can visualise an approximation of the story events and setting we spoke of in Chapter 2. In Figure 3.10 we can see the story events and settings extracted using the above methods for the first two sentences of "The Crow and the Fox". We provide a Python implementation of the aforementioned text simplification methods using the Spacy library at `https://github.com/mmxgn/spacy-clausie`.

3.8 Suspense

In the last section of this chapter we will briefly touch *Suspense* which we introduced in Chapter 2. Suspense is one of the most important elements of stories, radio drama included, and is what keeps the reader, or listener, engaged to the story. It is especially important in radio dramas due to the nature of the medium - listeners that are not captured from the beginning of the play will abandon it. Understanding suspense in the level of story text would aid us to convey it in radio drama when adapting the text, for example, by introducing relevant 'suspenseful' music or pauses at important steps in the drama.

Extracting suspense requires a deeper understanding of the narrative of the source

Algorithm 4 Proposition extraction using clauses extracted using the ClausIE extraction algorithm. *expandcc* is a function that expands conjunctions to sets of simpler terms, e.g. the phrase *Cat and Mouse* to $\{Cat, Mouse\}$

```
procedure PropositionExtraction($\langle S, V, O_i, O_d, C, A \rangle_{label}$)
    $P \leftarrow []$                                        ▷ An empty list
    $S' \leftarrow expandcc(S)$                              ▷ Expand conjunctions
    $O_i \leftarrow expandcc(O_i)$
    $O_d \leftarrow expandcc(O_d)$
    $C \leftarrow expandcc(C)$
    for $s \in S'$ do
        if $label \in$ {Intransitive (SV), Copular (SVC), Ext. Copular (SVA)} then
            $P \leftarrow P \cup \{s.V\}$                    ▷ Subject $s$ concatenated with verb $V$
        if $label =$ Ext. Copular (SVA) then
            for $a \in A$ do
                $P \leftarrow P \cup \{s.V.a\}$
        if $label =$ Ditransitive (SVOO) then
            for $o_d \in O_d$ do
                for $o_i \in O_i$ do
                    $P \leftarrow P \cup \{s.V.o_i.o_d\}$
        if $label =$ Monotransitive (SVO) then
            $O \leftarrow O_d \cup O_i$
            for $o_d \in O_d$ do
                $P \leftarrow P \cup \{s.V.o\}$
        if $label =$ Complex Transitive (SVOA) then
            for $o_d \in O_d$ do
                for $a \in A$ do
                    $P \leftarrow P \cup \{s.V.o\}$
                    $P \leftarrow P \cup \{s.V.o.a\}$
        if $label =$ Complex Transitive (SVOC) then
            $O \leftarrow O_d \cup O_i$
            for $o \in O$ do
                for $c \in C$ do
                    $P \leftarrow P \cup \{s.V.o.c\}$
        if $label =$ Copular (SVC) then
            for $c \in C$ do
                $P \leftarrow P \cup \{s.V.c\}$
    return $P$
```

story text and is not something that can be done with a shallow parsing method similar to the ones previously presented in this chapter. Suspense arises from the worry of the reader or listener towards the fate of a character and thus requires recognising who is the intended important character, what actions have they taken as well as understanding where these actions might lead [112]. Suspense cannot be analysed on the *discourse level* but requires a modelling on the *story events* and *story settings* level (Section 2.2.2). Furthermore, it has been shown that perceiving suspense is dependent on the knowledge of the reader about similar works of art. For example, suspense in spy stories come partly from the familiarity of the reader with the genre [38, p. 35].

Computational modelling of suspense in narrative, as far as the author of this thesis is aware of, has only been studied in the context of narrative generation. Despite this, we believe it is important to refer to the most well-known methods for suspense modelling for narrative generation, in the hopes that it leads to future endeavours on approximating it at the discourse level. One of the earlier computational methods for introducing suspense can be found in MEXICA [113], a system for narrative generation which describes a series of *tensions* described as patterns called *primitive actions*. Primitive actions describe suspenseful situations for characters. They comprise of a set of events that trigger these tensions as well as their consequences for the characters in the story setting. The author recognises a list of tensions such as *Life at risk* which is the tension caused when the life of an important character is at risk. In the context of Aesop Fables, the following would be an example of *Life at risk* [11]:

> "Once when a lion was asleep a little Mouse began running up and down upon him; this soon wakened the lion, who placed his huge paw upon him, and opened his big jaws to swallow him."

In this scenario the mouse is an important character and its capture by the lion and the uncertainty about his life causes suspense to the audience that worries about his fate. A system that models this suspense must be able to derive the possible outcomes. The less positive outcomes there are for the life of the mouse the higher the suspense [37].

A system that exploits the observation above is called SUSPENCER [114] which generates all the possible outcomes and models suspense based on the ratio of the number of failed outcomes (e.g. the mouse dies) versus the successful outcomes (e.g. the mouse lives). The higher this ratio is, the bigger the perceived suspense is. This approach, however, is computationally intractable for sufficiently complicated stories. An attempt

[11] Aesop's Fable: "The Lion and the Mouse"

at better modelling of suspense which also solves the problem of intractable outcome generation is Dramatis [115]. In this work, the authors identify the negative outcome of a story and try to devise an *escape plan* that the character can follow to avoid this fate (e.g. for the mouse to convince the lion not to eat him). Suspense is then modelled based on the probability of the escape plan succeeding. The three approaches we described above are based on explicit modelling of the story protagonist (or at least an important character). A work that disentangles character-specific modelling from suspense generation is found in [116, 117] therefore disentangling the emotional affinity towards the character from modelling of suspense.

3.9 Summary

In this section, we presented methods for extracting information from stories with the goal of adapting it into radio drama. We presented a procedure for automatically identifying characters that need to be communicated using actors in a drama as well as their dialogue lines. In addition, we discussed how we can extract events from the story and locations where these events happen. Furthermore, methods for recognising emotions underlying the text, as well as emotion of characters were presented. Finally, we briefly discussed suspense.

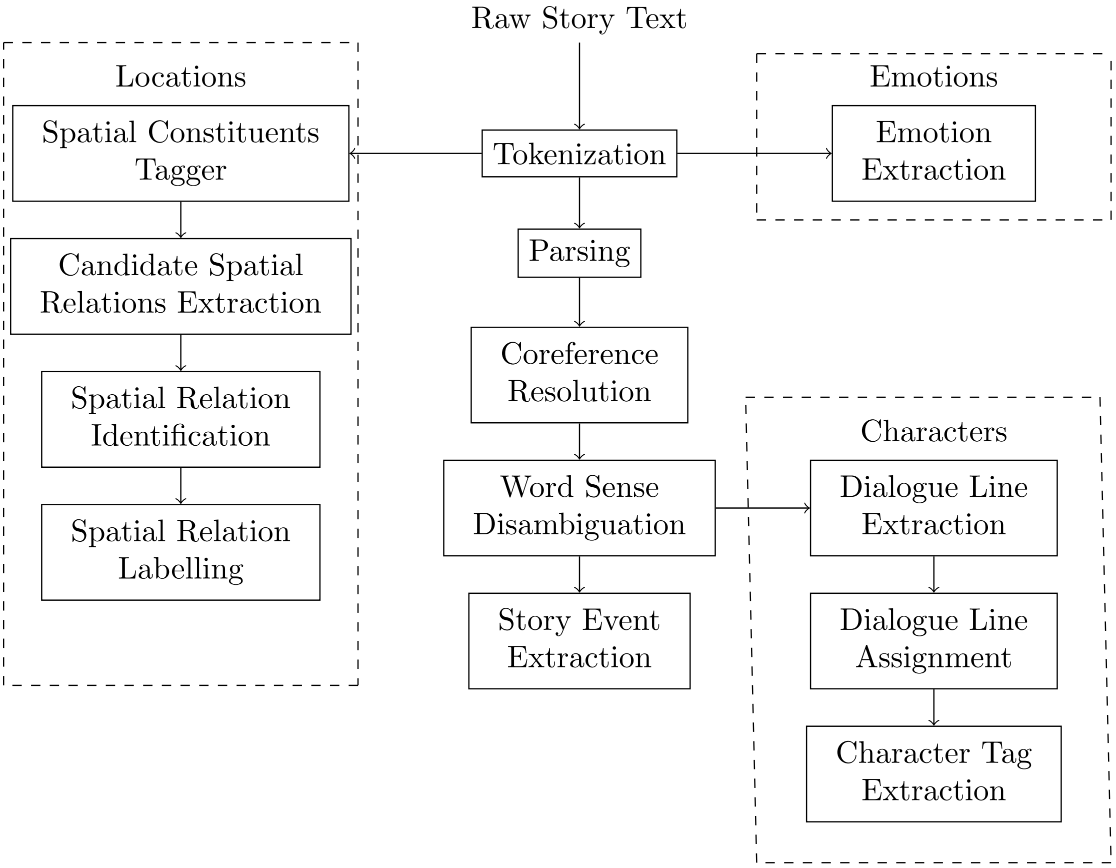
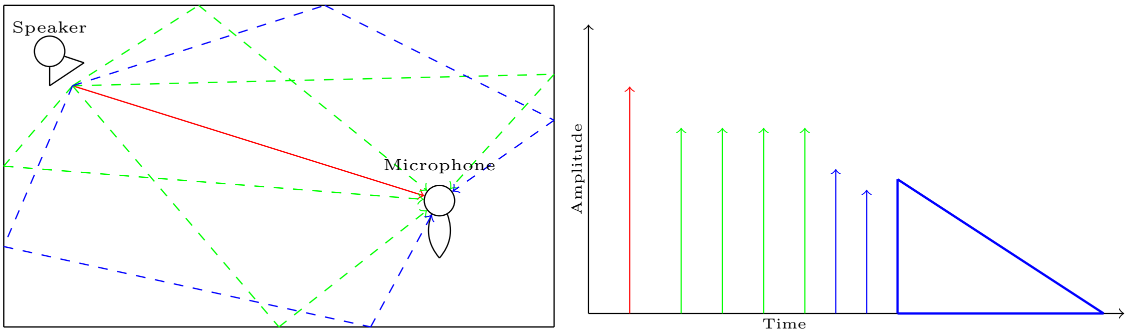


Figure 3.13: The complete framework consisting of the various computational approaches we described in this chapter. Boxes describe the various algorithms described throughout the chapter. Dashed boxes represent the four main narrative elements we try to extract: Characters and their lines (Section 3.3), Locations (Section 3.4), Emotions (Section 3.5), and Story events (Section 3.7).

Chapter 4

Reverberation

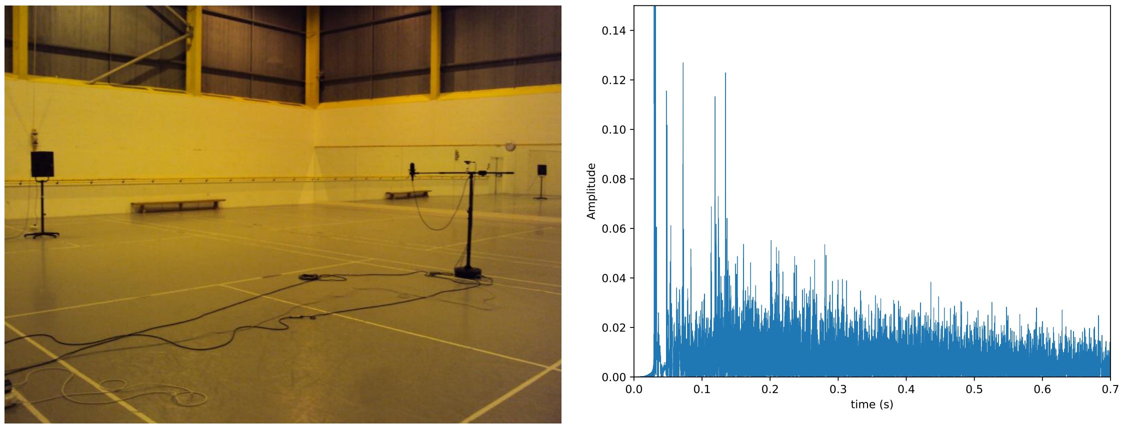
4.1 Introduction



(a) Diagram of sound reflections

(b) Impulse Response

Figure 4.1: Diagram for understanding reverberation and the corresponding impulse response plot. In this setting, we have a sound source (loudspeaker), a capture device (microphone) and four walls. Here we represent sound transmission using piece-wise line segments. Red colour is used for sound that is transmitted directly by the speaker, green for sound that gets reflected on the walls once, and blue twice. Here, sound gets transmitted by the speaker in all directions. Part of this sound is captured by the microphone (solid red line) and part hits the four walls (dashed lines). Part of those reflections are captured by the microphone (green lines), other parts get subsequently reflected (blue lines), etc. To avoid clutter, only two of the latter reflections is shown here. Each part of the sound that is captured by the microphone in Figure 4.1(a) is represented by a vertical arrow in Figure 4.1(b) at the time it was captured by the microphone. Further reflections captured that are not shown in Figure 4.1(a) are represented as a continuous triangle.



(a) Sports Centre, University of York

(b) Impulse Response

Figure 4.2: The Sports Centre at the University of York and the plot of the amplitude of its Room Impulse Response retrieved from the OpenAIR impulse response library [5]. The RIR shown was captured by Aglaia Foteinou and Simon Shelley using the sine-sweep method. Note the direct sound after a short delay of silence and the distinct peaks corresponding the first-order reflections.

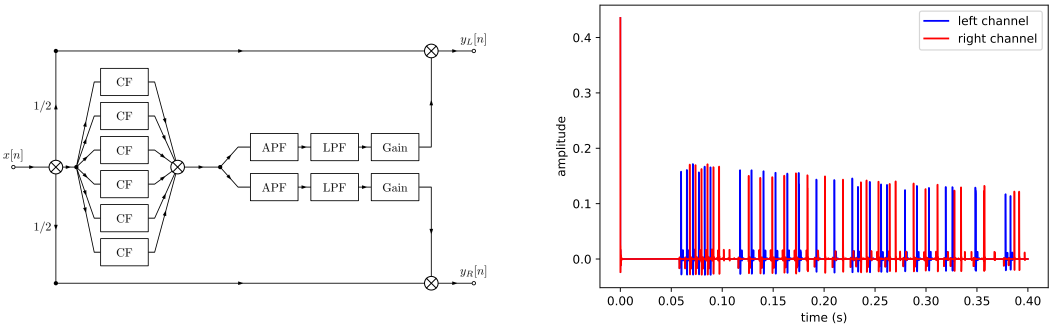
While part of audio effects (Section 2.7) we give specific emphasis to Reverberation since Chapters 5 and 6 relate to it. In Section 4.2, we discuss what Reverberation and Room Impulse Responses are. In Sections 4.3 and 4.4 we discuss what are the main methods to capture and replicate the effect using digital means. In Section 4.5 we discuss the DSP effect configuration in [118] which allows us to replicate reverberation artificially and which is controlled by simple parameters that can get mapped to the perceptual characteristics of reverberation discussed in Section 4.6. Finally, in Section 4.7 we introduce a mapping from those characteristics to the aforementioned parameters which allows a user to simply choose the desired reverberation parameters and control the effect in such way.

4.2 Reverberation and Room Impulse Responses

Reverberation (or Reverb for short) is the name given to the perceived effect of a sound reflected on the surfaces of walls, ceilings, and other objects in a space. Suppose we have the configuration of Figure 4.1(a) with a loudspeaker on one side of the room and a microphone on the other. When the loudspeaker emits a sound wave, it does so towards every direction. The part of the sound wave emitted directly towards the capture device will be the first to arrive, after a delay of $\frac{d}{c}$ seconds, where d is the distance between speaker and microphone and $c = 343m/s$ the speed of sound in air. The part of the wave that does not reach the microphone directly will get reflected on the surfaces of the room, and those reflections will reach the microphone at some time after the direct sound. Subsequently, those reflections that are not captured will be reflected back, etc. When travelling through the air or reflected by the surfaces, the sound wave gets partially attenuated. If we could produce a sound of infinitesimal duration at the source's position and record it using the configuration seen in Figure 4.1(a), the recorded signal would look like the one in Figure 4.1(b) which is called a *Room Impulse Response* (RIR). Producing such a sound in practice is impossible, however, since the speakers cannot reproduce signals of arbitrarily high energy. Instead, can approximate such a sound by clapping hands, shooting guns or popping air balloons. Each of those techniques has a particular set of limitations, the most common being the lack of energy at low frequencies that results to low Signal-to-Noise ratio (SNR) [119]. A different technique called the *Maximum Length Sequence* (MLS) method uses, instead of an impulse, pseudo-random noise as excitation signal and a deconvolution algorithm to derive the impulse response [120]. While this technique achieves much higher SNR compared to the aforementioned impulse techniques, it is still not sufficient for modern needs. Instead, the standard method for measuring RIRs is to use a continuous sinusoidal signal with exponentially increasing frequency (sweep) as the excitation signal [119, 121]. A realistic configuration to capture the RIR of a space using the sweep method and the resulting RIR can be seen in Figure 4.2.

4.3 Convolution Reverb

We could use an impulse response such as the one in Figure 4.2(b) to re-create the impression the specific environment by convolving a dry (with no reverberation already)



(a) DSP Block Diagram

(b) $400ms$ of the Impulse response

Figure 4.3: A *Moorer* algorithmic reverberator with the corresponding impulse response. $x[n]$ is the input (mono) signal and $y_R[n]$, $y_L[n]$ the right and left channels of the output. The reverberator is controlled by directly affecting the parameters of the digital filter in boxes.

sound source with the reverberation impulse response of that environment:

$$y[n] = \sum_{k=0}^{L} x[k]h[n-k] \tag{4.1}$$

where L is the length of the impulse response. This way of introducing reverberation to a dry sound source is called *convolutional reverberation*. There are various methods of recording or generating RIRs but are out of the scope of this thesis. The important characteristic of applying reverb with RIRs is that since RIRs are essentially sound files, those can be labelled and stored in a library for a radio drama producer to retrieve and use at a later stage.

4.4 Algorithmic Reverb

Another way to apply reverberation to a dry sound source is to process it using a systems block configuration called an *algorithmic reverberator*. Systems for algorithmic reverberation rely on networks of delay-inducing filters. They are older than their convolutional counterparts [122] but can be more expressive: while in convolutional reverberation effects choice of reverb is limited to the choice of a RIR file, filter parameters in algorithmic reverberators can produce a large range of perceived reverberation types, even types that do not exist in nature [123].

4.5 The Moorer Reverberator

An example of algorithmic reverberation is the *Moorer* reverberator [118]. It is a simple effect controlled by five simple parameters which can be mapped to perceived characteristics of space [124, 125]. The reverberator consists of a configuration of *Comb filters* (CF), *allpass filters* (APF), *low-pass filters* (LPF) and *gain* stages and can be seen in Figure 4.3 with its corresponding IR can be seen in 4.3(b). The Moorer reverberator is composed of simpler components seen in Figure 4.4. Each of the components in Figure 4.4 is controlled by two parameters: a delay d and a gain g. Below we provide the difference functions for each and we give a short description of the component's function:

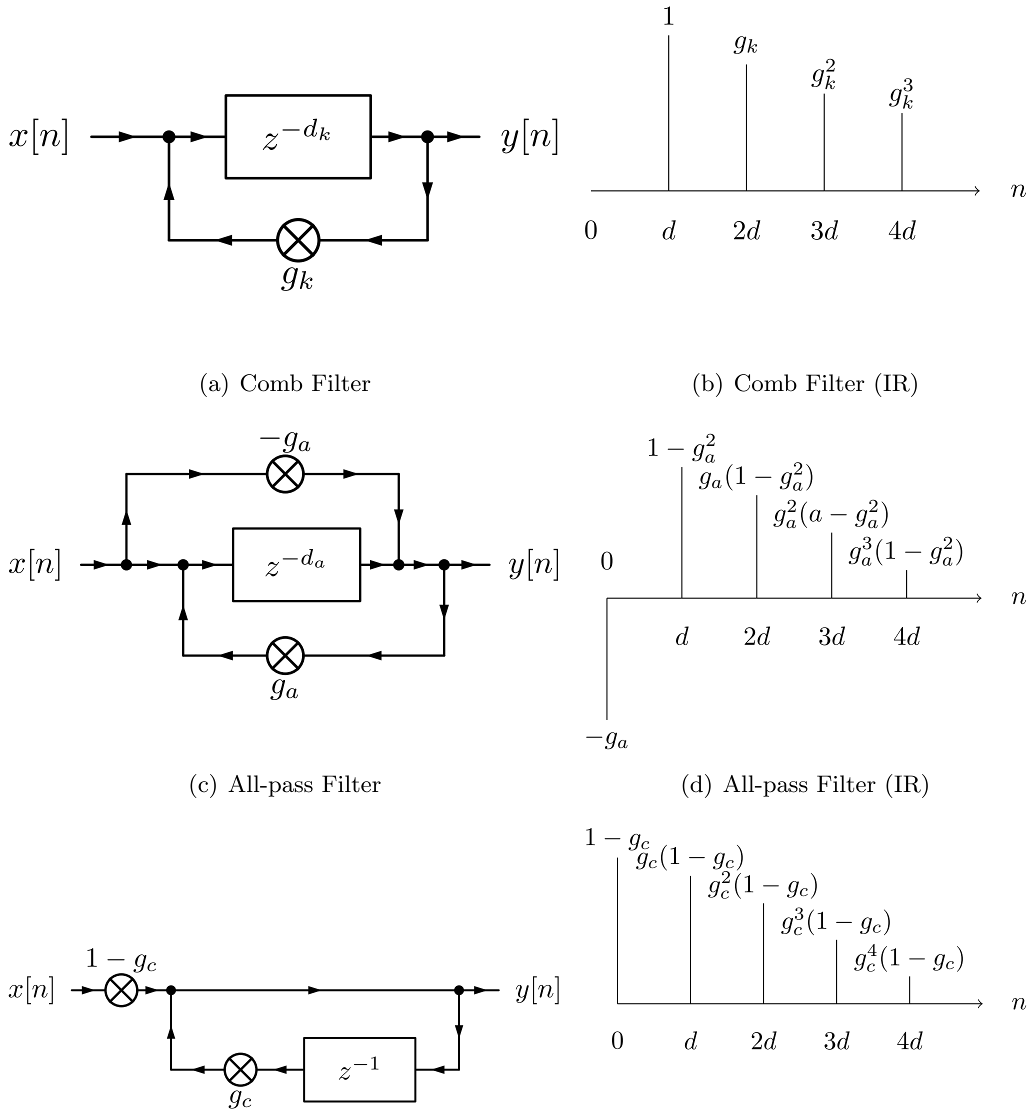
4.5.1 Comb Filter CF

– The comb filter's function is to introduce 'copies' of the signal of diminishing gain (g_k) at multiples of the filter's delay parameter d_k (Figure 4.4(b)) where k is a number from 1 to 6 corresponding to a comb filter in Figure 4.3. Each comb filter is used to 'simulate' a surface of a closed space. In the Moorer reverberator, a comb filter 'simulate' each of the six surfaces of a shoebox-shaped room. The block diagram and impulse response of a comb filter can be seen in Figure 4.4(a) and 4.4(b) respectively. The difference equation of a comb filter is:

$$y[n] = x[n - d_k] + g_k y[n - d_k] \tag{4.2}$$

When used in cascade each comb filter is labelled using a number $k = 1..6$. The delays $d_{1...k}$ take values between 10 and 100ms and have a constant ratio:

$$\frac{d_{i+1}}{d_i} = \frac{1}{1.5} \tag{4.3}$$



(a) Comb Filter

(b) Comb Filter (IR)

(c) All-pass Filter

(d) All-pass Filter (IR)

(e) Low-pass Filter

(f) Low-pass Filter (IR)

Figure 4.4: Block diagrams of the components of the Moorer reverberator (left) with their corresponding impulse response (right).

with d_1 having the largest value. The values d_k and g_k are further set to satisfy the following constraint:

$$\frac{d_l}{\log g_l} = \frac{d_m}{\log g_m} \tag{4.4}$$

which ensures the comb filters have the same *reverberation time*. Distributing delays with such a constant ratio will result in overlapping echoes. To solve this problem, we use the solution given in [124, 125], which is to make sure the delays are rounded to a co-prime set of integers when converted to samples, i.e. impulses from the six different comb filters should all fall on different times. We discuss about reverberation time in Section 4.6. From Eq. 4.3 and 4.4 we observe that by controlling the gain and delay of the first comb filter g_1 and d_1 we control the filter parameters for the rest of the comb filters as well.

4.5.2 All-pass Filter APF

The all-pass is used in a similar way as the comb filter. It introduces attenuated 'copies' of the input sound at intervals controlled by the filters' delay parameter d_a. The two parallel all-pass filters in Figure 4.3 are set with a slightly different delay so that a slightly different reverberation is introduced to the left and right channels. This gives the impression of a 'stereo' reverberation effect. We observe this in Figure 4.3(b) where the left channel (blue colour) is slightly different to the right channel (red colour). The block diagram of the all-pass filter can be seen in Figures 4.4(c) and 4.4(d) respectively. Its difference equation is:

$$y[n] = x[n - d_a] - g_a x[n] + g_a y[n - d_a] \tag{4.5}$$

The delays of the two allpass filters $d_{a,1}$, $d_{a,2}$ and the gains $g_{a,1}$, $g_{a,2}$ [124]:

$$d_{a,1} = 0.006 + \frac{m}{2} \tag{4.6}$$

$$d_{a,2} = 0.006 - \frac{m}{2} \tag{4.7}$$

$$g_{a,1} = g_{a,2} = \frac{\sqrt{2}}{2} = 0.707 \tag{4.8}$$

where the delays are expressed in seconds (s). Since the gains are constant and the delays of the two all-pass filters are both dependent on m, it is used as a parameter to control the all-pass filter stage of the used Moorer reverberator.

4.5.3 Low-pass Filter LPF

We briefly introduced the low-pass filter when we discussed about EQ in Section 2.7.2. We use a low-pass filter after the comb and all-pass filter stages to simulate the air and wall absorption present in natural reverberation. The filter used in this case is a very simple single-pole lowpass filter controlled by a single gain parameter g_c and a difference equation:

$$y[n] = (1 - g_c)x[n] + g_c y[n-1] \tag{4.9}$$

Its block diagram and impulse response can be seen in Figures 4.4(e) and 4.4(f) respectively. Finally, the relation between the low-pass cutoff function f_c and the gain g_c is given by:

$$g_c = 2 - \cos\left(2\pi\frac{f_c}{f_s}\right) - \sqrt{\left(cos\left(2\pi\frac{f_c}{f_s}\right) - 2\right) - 1} \tag{4.10}$$

4.5.4 Dry/Wet Mix Gain

There is a last ‘Gain’ stage in Figure 4.3. This simply multiplies the signal via a parameter $G \in [0, 1]$:

$$y[n] = Gx[n] \tag{4.11}$$

The parameters controlling the Moorer reverberator can be seen in Table 4-A.

4.6 Perceptual Measures of the Impulse Response

In Figure 4.3(b) we see the first $400ms$ of the impulse response of a reverberation effect. By plotting the impulse response, one might get a rough idea on some of its characteristics. For example, by observing the time it takes for the impulses to fall under a

Parameter	**Values**	**Description**
d_1	$[0.01, 0.1]$	Delay of the first comb filter
g_1	$(0, 1)$	Gain of the first comb filter
m	$(0, 0.006)$	Difference of delays between the all-pass filters.
g_c	$(0, 1)$	Gain of the low-pass filter
G	$(0, 1)$	Dry/Wet mix gain

Table 4-A: Parameters controlling the Moorer reverberator effect of Figure 4.3

perception threshold (e.g. an amplitude of 0.001) we can estimate that the reverberation might give a perception of a 'spacious' or 'damp' room depending on whether this time is large or small.

Observing the impulse response, however, is not practical: we would like to characterise the impulse response using simple measures (such as the time we mentioned above). We derive five such measures: *Reverberation Time* T_{60}, *Echo Density* D, *Central Time* T_c, *Spectral Centroid* S_c, and *Direct-to-Reverberant Ratio* D_{rr}. For the architecture in this section, we describe how we derive these measures from an impulse response of both an algorithmic (e.g. Moorer) reverberation effect as well as realistic impulse responses recorded for use with a convolutional reverb. We then discuss how we can synthesize impulse responses that adhere to these measures using the Moorer reverberator shown in Section 4.5.

4.6.1 Reverberation Time T_{60}

Reverberation time is the time it takes for the power of the impulse response to drop to imperceptible levels, usually below $60dB$ for T_{60}. For RIRs synthesised by the Moorer reverberator, this is usually just a matter of measuring the levels of the impulse response dropping below a relative 10^{-3} compared to the direct sound (Figure 4.5). The reverberation time of the Moorer reverb's IR can also be calculated as a function of the delay and gain parameters d_1, g_1 of the first comb filter, the gain of the low-pass filter g_c and

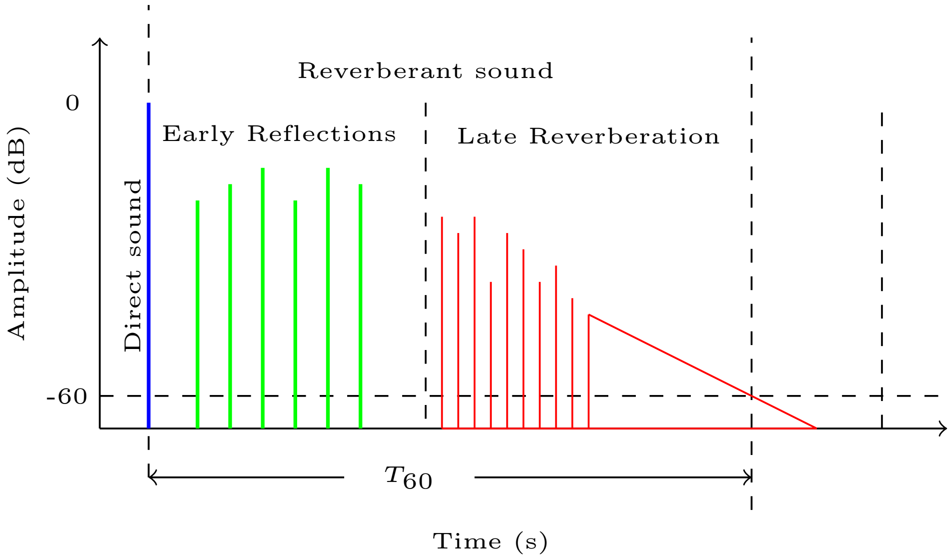


Figure 4.5: A Moorer reverberator's Impulse Response. Estimating T_{60} requires measuring the time it takes until the amplitude of the impulse response falls under a value of $0.001(60dB)$.

the dry/wet mix gain G of the 'Gain' stage in Figure 4.3 [124]:

$$T_{60} = \frac{d_1}{\log g_1} \log \left(\frac{0.001\sqrt{2}}{(1 - g_c)G} \right) \tag{4.12}$$

Equation 4.12 allows us to predict the reverberation time when using a Moorer reverberator and thus introduce reverb with desired reverberation time (e.g. a long reverb time gives the impression of a 'spacious' room).

For recorded RIRs used in convolutional reverb, the presence of noise makes measuring T_{60} directly from the IR unreliable since the noise level will probably be higher than $60dB$. In order to measure reverberation time at $60db$, we use PYTHON-ACOUSTICS[1] which implements a method based on energy decay curves [126] and is shown in more detail in Appendix A.1. T_{60} is not measured at the full spectrum, but at 8 octaves centred at $C_f = \{63, 125, 250, 500, 1000, 2000, 4000, 8000\}Hz$ and the average is taken as the chosen reverberation time T_{60}:

[1] `https://github.com/python-acoustics`

$$T_{60} = \frac{1}{8} \sum_{f \in C_f} T_{60}^{f} \tag{4.13}$$

where each superscript f is the centre frequency of the respective band.

4.6.2 Echo Density E_d

Echo density is the number of distinct reflections of the direct sound that can be heard. It is usually measured from the impulse response during the early phase of the reverberation. Again, as with reverberation time, echo density can be directly measured in the case of an algorithmically synthesised RIR by measuring the number of echoes (peaks) over a small period of time $\tau_D = 0.1s$ at the early reverberation phase (Figure 4.5). Similar to the reverberation time, echo density can be described as a function of Moorer reverberator's parameters:

$$E_d = \frac{\tau_D}{d_a} \sum_{1}^{6} \frac{1}{d_k} \tag{4.14}$$

where $\tau_D = 0.1$, d_a the delay of an all-pass filter (either the left or the right channel) from the all-pass filter stage in Figure 4.3 and d_k the delay of the k-th comb filter.

As with reverberation time, Echo Density is not as easy to measure in the case of recorded RIRs. To do that, we use the method described in [127] which does not measure the echo density exactly, but a metric that highly correlates with it. The method assumes that distinct echoes are outliers of a Gaussian bell which models the distribution of samples around points of time when the reverberation is considered fully mixed. In our case, we measure this metric at 8 different time instances $T = \{5, 10, 15, 20, 30, 50, 90, 190\}ms$ of the RIR and take the average.

$$E_d = \frac{1}{8} \sum_{\tau \in T} E_d^{\tau} \tag{4.15}$$

4.6.3 Direct-to-Reverberant Ratio D_{rr}

In [124, 128, 129], the authors base their work on a measure they call Clarity and denote by C. The way, however, it is defined in those works it is more similar to the Direct-to-Reverberant ratio D_{rr} when measured in recorded RIRs. This describes the ratio of energies of the direct sound, to the rest of the reverberation, expressed in dB.

$$D_{rr} = \frac{E_{direct}}{E_{reverberant}} \tag{4.16}$$

In the algorithmically synthesised case, it is a simple matter of taking the ratio of the RIR value at time $t = 0$ to the sum of squares of the rest of the values. It can also be predicted from the Moorer reverb's parameters [124]:

$$D_{rr} = -10 \log_{10} \left(G^2 \frac{1 - g_c}{1 + g_c} \sum_{k=1}^{6} \frac{g_k^2}{1 - g_k^2} \right) \tag{4.17}$$

In the recorded RIR case again this is not the case since we are uncertain of the location and duration of the direct sound. To estimate D_{rr} we use the method described in [130] which calculates the ratio of energies around a $5ms$ window around the highest peak of the signal and divides it by the rest of the RIR signal.

$$D_{rr} = \frac{\int_{t_0 - 2.5ms}^{t_0 + 2.5ms} y^2(\tau) d\tau}{\int_{t_0 + 2.5ms}^{T} y^2(\tau) d\tau} \tag{4.18}$$

where $y(t)$ is the signal of the RIR, t_0 the location of the highest peak, and T the total duration of the impulse response.

4.6.4 Central Time T_c

Central time T_c is the "centre of mass" of the energy in the impulse response [131] and in previous works has been found to correlate to perceptual descriptors for reverberation such as *boomy*, or *church-like* [124, Table 1–3]. It can be calculated the same way for algorithmic and recorded RIRs and is given by:

$$T_c = \frac{\int_0^T \tau y^2(\tau) d\tau}{\int_0^T y^2(\tau) d\tau} \tag{4.19}$$

where again, y is the signal of the RIR, and T its total duration. In the Moorer reverberator it can be expressed using the parameters (gains and delays) of the architecture's components [124]:

$$T_c = \sum_{k=1}^{6} \frac{d_k g_k^2}{(1 - g_k^2)^2} / \sum_{k=1}^{6} \frac{g_k^2}{1 - g_k^2} + d_a \tag{4.20}$$

4.6.5 Spectral Centroid S_c

Similar to central time, spectral centroid is the frequency at the centre of gravity of the spectrum of the RIR and is correlated to the *brightness* of the impulse response [124, 132]. In our case it is calculated using Librosa [133]. The signal is split into frames, its Short Time Fourier Transform (STFT) is calculated and each frame n of the STFT is normalised. The spectral centroid of frame n is then given by:

$$S_c^n = \frac{\sum_{k=1}^{K} k \cdot STFT[n, k]}{\sum_{k=1}^{K} STFT[n, k]} \tag{4.21}$$

Where K is the number of bins of the STFT. We calculate the spectral centroid S_C^n for frames at $T = \{5, 10, 15, 20, 30, 50, 90, 190\} ms$ and take their average to compute the final spectral centroid:

$$S_c = \frac{1}{8} \sum_{\tau \in T} S_C^{\tau f_s} \tag{4.22}$$

Where f_s is the sampling rate of the recorded signal. Its relation to the Moorer's reverb effect parameters is given by:

$$S_c = \sum_{n=0}^{f_s/2} \frac{n}{1 + g_c^2 - 2g_c \cos(2\pi n/f_s)} \Big/ \sum_{n=0}^{f_s/2} \frac{1}{1 + g_c^2 - 2g_c \cos(2\pi n/f_s)} \tag{4.23}$$

4.7 Controlling the Moorer Reverberator using measures of reverberation

In Section 4.6, we introduced five simple measures of the impulse response that relate to how we perceive reverberation. Moreover, we found that we can predict those measures from five control parameters of a Moorer Reverberation effect. In this section, we derive the inverse mapping, from the impulse response measurements to the Moorer reverb parameters. This enables us to introduce reverb based on desired characteristics (e.g. long reverberation time or high clarity). We formulate the problem as such:

> "Given a set of the five impulse response characteristics introduced in Section 4.6 (T_{60}, E_d, T_c, D_{rr}, and S_c), return a set of values for the five reverb parameters in Table 4-A (d_1, g_1, d_a, g_c, G)."

Spectral Centroid S_c depends only on g_c and thus we can easily estimate a value for g_c given a value for S_c via numeric optimisation. The rest of the parameter-measure pairs, however, form a 4×4 non-convex system of equations with constraints given by the allowed value ranges shown in Table 4-A. Finding the feasibility space of such system is non-trivial, and in some cases there exist no exact solutions: e.g. it is not possible to have an arbitrarily low echo density E_g as well as reverb time T_{60}. Instead, we approximate a set of values that minimises an objective function that gives a non-exact, but hopefully good enough, solution. Suppose we have a vector of actual reverberation measurements (remember we can directly estimate g_c from S_c):

$$\boldsymbol{v} = \begin{bmatrix} T_{60} & E_d & D_{rr} & T_c \end{bmatrix}^T \tag{4.24}$$

and the desired reverberation measurements:

$$\boldsymbol{v'} = \begin{bmatrix} T'_{60} & E'_d & D'_{rr} & T'_c \end{bmatrix}^T \tag{4.25}$$

we need to find a set of parameters that minimises the Euclidean distance of the target measurements from the actual measurements, given the constraints of our parameters. Furthermore, we add the extra constraint of a uniform error distribution. What we do is find the optimal solution for the problem below (all variables are normalised to 0-1):

$$\begin{aligned}
&\underset{\boldsymbol{x}=[g_1\ d_1\ d_a\ G]^T}{\text{minimize:}} f_0(\boldsymbol{x}) = \sqrt{\boldsymbol{e}^T\boldsymbol{e}} + \mathrm{Var}[\boldsymbol{e}]^2 \\
&\quad \text{subject to:} \\
&\qquad\qquad 0 < g_1 < 1, 0 < G < 1, \\
&\qquad\qquad d_{1,min} \leq d_1 \leq d_{1,max}, \\
&\qquad\qquad d_{a,min} \leq d_a \leq d_{a,max} \\
&\qquad \text{where:} \\
&\qquad\qquad \boldsymbol{e} = \boldsymbol{v} - \boldsymbol{v'}
\end{aligned} \tag{4.26}$$

Instead of trying to exactly solve the system of equations in 4.26, we derive an approximation that depends on optimising a single variable (Appendix A.2):

$$
\begin{aligned}
&\underset{g_1}{\text{minimise:}}\ f_0(g_1) = \sqrt{\boldsymbol{e}^T\boldsymbol{e}} + \text{Var}[\boldsymbol{e}]^2 \\
&\text{subject to:} \\
&\qquad\qquad 0 < g_1 < 1, \\
&\qquad \text{where:} \\
&\qquad\qquad G = f_1(C', g_1) \\
&\qquad\qquad d_1 = f_2(C', T'_{60}, g_1) \\
&\qquad\qquad d_a = f_3(C', T'_{60}, D', g_1)
\end{aligned}
\tag{4.27}
$$

Solving the system in Eq.4.27 gives us a sub-optimal solution when there is no exact solution in our feasible space, and the exact solution when there is one. If we are deriving those measurements directly from a Moorer reverberator impulse response, we are expecting the problem to have an exact solution, we only expect non-exact solutions when the reverberation measurements are chosen arbitrarily.

4.8 Conclusion

In this chapter, we discussed the effect of Reverberation and how it can be introduced used recorded signals that capture the characteristics of a space (Room Impulse Responses) as well as using algorithmic means. Furthermore, we introduced simple measures that can perceptually describe such impulse responses. Finally, we showed how we can introduce reverberation algorithmically based on desired values of those perceptual measures. In Chapter 5 we discuss how we can use those measures together with the algorithmic reverberation effect we discussed and recorded RIRs to introduce appropriate reverberation to a radio drama using information extracted from story text.

Chapter 5

Organising Assets for Radio Drama Production

5.1 Introduction

In Chapter 3 we discussed methods for extracting information from a source story that is used when adapting the story to radio drama. In this chapter, we first discuss methods for retrieving music, sound effects, and room impulse responses for reverberation. In Section 5.2, we show we can use the methods for Text Simplification and Coreference Resolution in Chapter 3 to extract tags from natural story sentences with the goal of retrieving spot sound effects from an online library of sound effects. In Section 5.3, we discuss how Room Impulse Responses can be labelled and retrieved using such tags. Finally in Section 5.4, we discuss retrieval of music.

5.2 Tag-based retrieval of Sound Effects

In this section, we discuss how we can automatically retrieve sound effects relevant to a story we are adapting to radio drama, in order to assist the sound managers in their job. Similar to Sections 5.3 and 5.4, we assume that the team has access to a library of sound effects where sounds can be searched and retrieved using a collection of tags. To retrieve relevant sound effects for the story therefore we need to find ways to extract relevant tags from the stories, which in our case are written in natural language.

We search for such methods in previous attempts that try to automate the task of *soundscape composition.* In soundscape composition, a sound designer has the goal of producing a sound 'scene' able to elicit a mental image of a specific environment to the listener [134]. Such scenes can be described in natural language, e.g.:

> "On hot summer days, down by the river, we would listen to the hum of insects." [1].

There is a number of previous methods that try to assist in or automate the soundscape composition process. In [136], the authors use tag-based search and a knowledge base based on WordNet [62] to augment a user's search query with the corresponding tag 'concepts' and thus retrieve sound effects that do not correspond simply to the objects they have been queried about, but also to the concept they relate to (e.g. a 'city' tag relates also to 'cars', 'pedestrians', etc). The SoDa project [137] uses a sound effects library richly annotated with various kinds of metadata to allow a creator to 'explore' and choose sound effects for use in a soundscape. In the Audio Metaphor soundscape synthesis engine [134, 135], the authors extract tags from natural sentences using a tag slicing algorithm called SLICE which uses simple word features and a sliding window, to automatically construct queries for searching and retrieving sounds from the FreeSound online sound effects library [138]. Furthermore, they use social media posts to augment such queries with new tags. This last technique seems appropriate for our task of extracting tags for retrieval of sound effects from story sentences. Consider the following sentence[2]

> One day the countrymen noticed that the mountains were in labour; smoke came out of their summits, the earth was quaking at their feet, trees

[1] Quote taken from [135]

[2] Taken from the Aesop's Fable: "The Mountains in Labour"

were crashing, and huge rocks were tumbling .

In the above sentence, there are references to previous text (e.g. 'their' → 'the mountainś'), as well as coordinating conjunctions ("and, taking his cane ..."). SLICE [135] extracts tags by considering all the nouns phrases in a sentence, as such:

day countrymen mountains labour smoke summits earth feet trees huge rocks

as well as all the possible sub-sequences of decreasing length as queries:

day countrymen mountains labour smoke summits earth feet trees huge

countrymen mountains labour smoke summits earth feet trees huge rocks

⋮

day

countrymen

⋮

rocks

Overall, 66 queries were constructed from the above sentence, out of which only 8 will retrieve sound effects[3] when querying FreeSound. Story sentences like the above tend to be more complex than descriptions of soundscapes. They tend to contain references to previous text (e.g. pronouns such as 'himself', 'his', 'them'), embedded clauses, coordinated conjunctions, and other causes of sentence complexity [139]. When querying SFX libraries, this complexity leads to the construction of a very large amount of queries for stories since they contain a large number of complex sentences. This is especially strainful for online libraries with limited bandwidth, as is the case for FreeSound. Furthermore, when taking sub-sequences of tags, those that are related to each other in the story are not going to be used together in a query without including all the tags that exist in the span between them. For example, in the sentence above there is no query

[3] As of 6 Oct. 2020

that contains both *mountain* and *summits* without including the tags *labour*, and *smoke*. We observe therefore that simply using the algorithm in [135] without appropriate pre-processing leads to two main issues: a large number of queries, and therefore requests to an online library, as well as not entirely relevant queries. To overcome the issues mentioned above, we test coreference resolution and text simplification as a pre-processing step before retrieval. This lets us do two things:

1. Replace text referents with the text they refer to, for example: "*their* summits" → "*the mountains* summits".

2. Extract the coordinating conjunctions (identified by semicolons ';', commas ',', and words such as 'and' in the sentence above) from the complex sentence into multiple simpler sentences:

   The countrymen noticed.
   The mountains were in labour.
   Smoke came out of their summits.
   The earth was quaking at their feet.
   Trees were crashing.
   Huge rocks were tumbling.

Using SLICE to extract tags from the above sentences lead to the construction of only 16 queries, which is a significant decrease from the 66 of the original sentence. While in this case the successful queries remained the same, the successful-to-total queries ratio increased. To test more systematically the effect of coreference resolution and text simplification to sound effects retrieval, we used the first sentence of 44 Aesop's Fables chosen from the dataset of 249 Aesop's Fables we introduced in Chapter 3 to construct tags and query the FreeSound library for sound effects. The first sentence was chosen deliberately for two reasons:

1. We observed that in Aesop's Fables they tend to introduce the characters and the

locations of the story, and therefore are of interest when composing, e.g. atmospheric effects

2. It is easier to apply coreference resolution since the pronouns refer to entities in the same sentence.

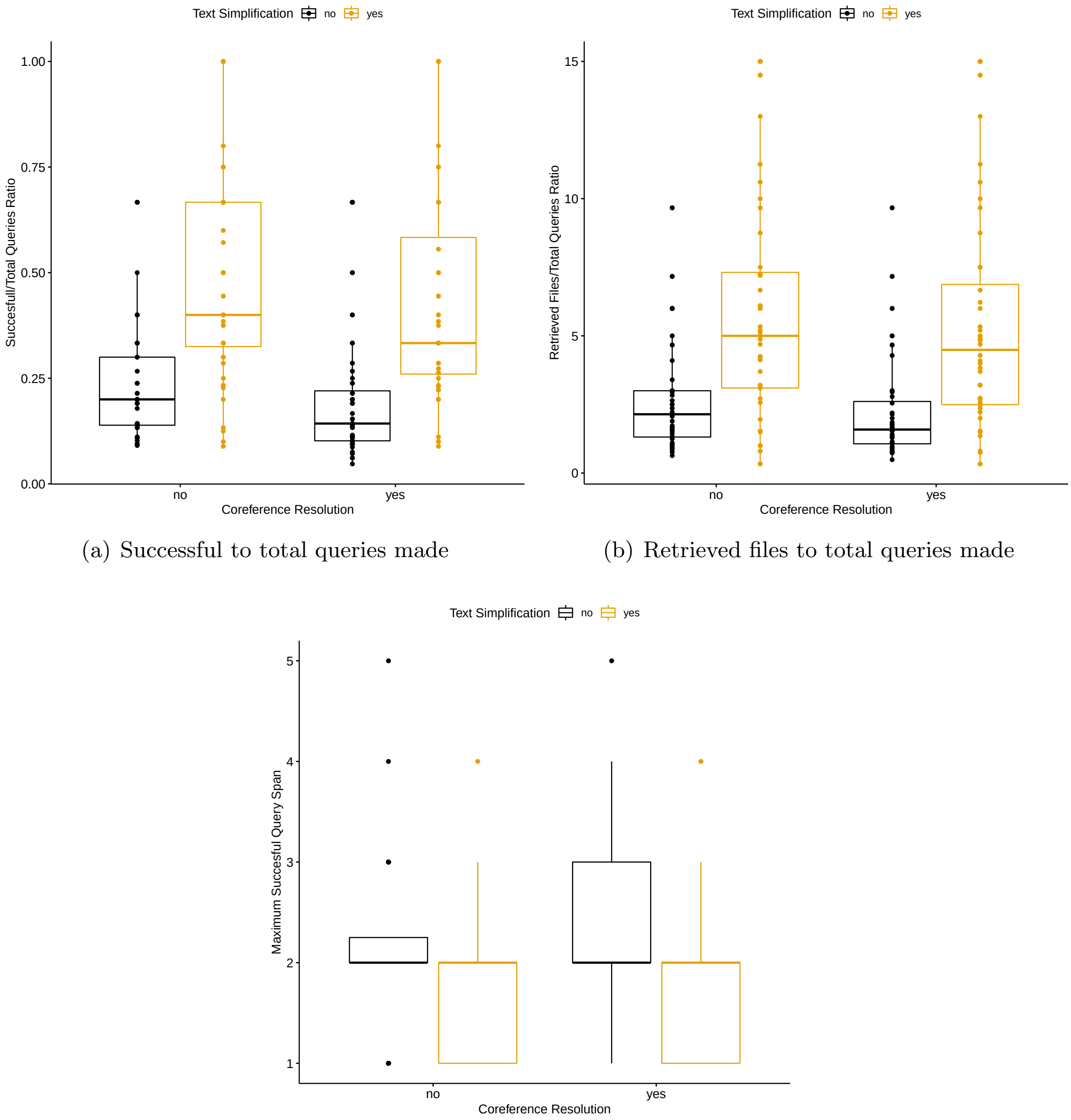
The small number of Aesop's fables was chosen simply to not overwhelm the FreeSound library with queries. We examined how much text simplification and coreference resolution affect the following:

1. The ratio of successful queries (queries with at least a sound effect retrieved) to the number of all queries made.
2. The ratio of the number of retrieved files to the number of all queries made.
3. The span length of the successful queries.

The first two will tell us how much the ability of the method to retrieve relevant files is affected since a higher number will mean either a greater number of successful queries or a smaller number of queries, both of which are desired when querying a library. The last number will provide us with an idea of how relevant the sound effects retrieved are since we expect queries that contain more words from a sentence to retrieve sound effects more specific to it.

5.3 Tag-based retrieval of Recorded RIRs

In Chapter 4 we discussed about artificial reverberation. We saw that there exist two distinct ways to produce reverberation with computational means: by imitating the reverberation process using algorithmic approaches, or by convolving with a *Room Impulse Response* which tries to capture the behaviour of a space across time and frequency [123]. While algorithmic approaches predate convolutional historically, the latter became prevalent in cases when sound engineers wanted to assign a label (e.g. forest, underground park) to the acoustics of an environment. Those approaches rely on using a convolutional



(a) Successful to total queries made

(b) Retrieved files to total queries made

(c) Maximum Successful Query Span

Figure 5.1: Boxplots that visualise the effect of coreference resolution and text simplification for the three tasks. Note that there is an increase in the ratios of successful queries and retrieves files associated with text simplification and a decrease in the maximum successful query span. Coreference resolution on the other hand seems to not affect the results at a significant degree.

reverberation plug-in in the sound engineer's digital audio workstation and a library of RIRs. These RIRs are stored as audio files and up to now needed to be manually annotated (usually in their filename), and retrieving them relied on simply searching the text

Task	N	**CR**	**TS**	μ	σ
Successful-to-total Queries	44	no	no	0.233	0.136
	44	no	yes	0.483	0.256
	44	yes	no	0.193	0.141
	44	yes	yes	0.435	0.256
Retrieved files-to-total Queries	44	no	no	2.663	1.949
	44	no	yes	5.874	4.123
	44	yes	no	2.216	1.890
	44	yes	yes	5.317	3.978
Maximum Successful Query Span	44	no	no	2.159	0.805
	44	no	yes	1.773	0.677
	44	yes	no	2.432	0.925
	44	yes	yes	1.886	0.754

Table 5-A: Mean μ and standard deviation σ for each sample in our analysis. N is the number of data points in each sample.

Task	**Factor**	$\Delta\mu$	p
Successful-to-total Queries	TS	0.246	2.72×10^{-13}
	CR	0.044	0.157
Retrieved files-to-total Queries	TS	3.156	4.81×10^{-10}
	CR	0.502	0.295
Maximum Successful Query Span	TS	0.465	1.46×10^{-4}
	CR	0.193	0.109

Table 5-B: Results of two-way ANOVA. We observe that for the three tasks there are significant differences in the mean when text simplification is present ($p < 0.05$).

for these annotations. This may raise issues when the files are labelled incorrectly or not according to their perceived characteristics (e.g. `ir.wav` instead of `forest.wav`). In this section, we present a way to alleviate those issues, by automatically tagging and retrieving unlabelled RIRs based on the perceptual effect they have on sound. For example, we can search RIRs that make a sound *loud* and *dark* based on their content. We believe that our method can lead to assistive tools for sound engineers, allowing them to browse a library of RIRs easily; aid in field recording scenarios [140] by quickly organising recorded RIRs; and organising and retrieving the numerous freely available RIRs available on the net.

Since RIRs are stored as regular audio files, an approach would be to use content-based audio retrieval tools to label and retrieve them [141]. Those approaches, however, make only minor assumptions about the features of the sound, and try to learn the parts that are useful in retrieval by analysing each separate frame of the signal and using sophisticated machine learning tools to assign labels to audio files. However, RIRs have been studied extensively in the past and their perceptually relevant characteristics are well known. We exploit these characteristics to provide a retrieval system for RIRs.

In [124] the authors described an algorithmic reverberation effect that can be controlled by perceptually relevant measurements of the reverberation impulse response, such as reverberation time and echo density, to apply reverberation based on specific terms (e.g. *boomy* or *not boomy at all*). The work continued in [128] which created a map from those terms and applied reverberation either by searching for a specific term or by exploring the descriptor map. The authors of [142] presented an effect plugin architecture for algorithmic reverberation that allows crowdsourcing of semantic descriptors from the users of the effect. The work presented in the following sections is similar to the works above in that it tries to apply reverberation using crowd-sourced semantic descriptors but differs in that it allows applying reverb using multiple descriptors (for example, *dark* and *muffled* instead of just *dark* or *muffled*). A novelty introduced is that we do not limit those descriptors to a preexisting set of tags, as in the works above, but allow searching the RIRs using words with similar meanings. Additionally, it uses convolutional reverberation and recorded room impulse responses instead of an algorithmic reverb effect.

5.3.1 Retrieval based on similarity

We approach the problem of retrieving RIRs from text queries in a similar fashion to [141]. A content-based retrieval system that can retrieve RIRs from queries has the goal of taking a set of RIRs M, and a query q and ranking them so that a RIR that is more relevant to q than m' gets ranked earlier:

$$r(m, q) < r(m', q) \tag{5.1}$$

Similarly, a system that assigns tags t in T to RIRs m in M should rank tag t that is more relevant to m than t' ahead of it:

$$r(m, t) < r(m, t') \tag{5.2}$$

In order to construct such a system, we can construct functions F such that:

$$F(m, q) > F(m', q) \tag{5.3}$$

$$F(m, t) > F(m, t') \tag{5.4}$$

A simple but effective method is to count occurrences of pairs (m, t) in a set, where m is an impulse response and t a tag, and use those occurrences in a matrix as our scores (normalised so that each row has length 1 for convenience). Suppose we have a query q consisting of tags $t_1 \dots t_N$ and a matrix of occurrences $\boldsymbol{W}$, we can define the score as:

$$F(m, q) = w_{m,t_1} + w_{m,t_2} + \cdots + w_{m,t_N} \tag{5.5}$$

If we represent tags as a set of column vectors $\boldsymbol{t} \in [0, 1]^N$, we can write the above equation as:

$$F(m, q) = \boldsymbol{w}_m \boldsymbol{q} =< \boldsymbol{w}_m^T, \boldsymbol{q} > \tag{5.6}$$

where $\boldsymbol{w}_m$ is the row of the occurrence matrix $\boldsymbol{W}$ corresponding to impulse response m, $< \cdot, \cdot >$ represents the vector inner product, and $\boldsymbol{q}$ is the sum of tags $\boldsymbol{t}_{1 \dots N}$. It is worth noting that the inner product is a measure of *similarity*, the more similar $\boldsymbol{w^T}$ is to $\boldsymbol{t}$, the

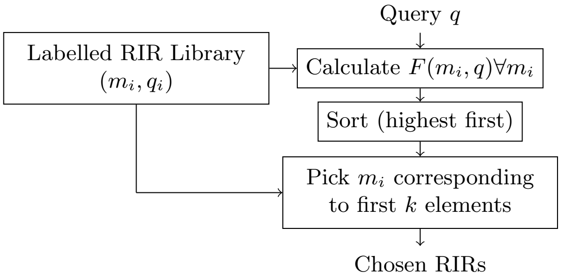

Query q
Labelled RIR Library
(m_i, q_i)
Calculate $F(m_i, q)\forall m_i$
Sort (highest first)
Pick m_i corresponding to first k elements
Chosen RIRs

Figure 5.2: Block diagram for retrieving the k most relevant RIRs to query q.

higher the value of the product. By using the vector identify for the inner product:

$$< \boldsymbol{a}, \boldsymbol{b} >= \|\boldsymbol{a}\| \, \|\boldsymbol{b}\| \cos\angle(\boldsymbol{a}, \boldsymbol{b}) \tag{5.7}$$

We can further write Eq. 5.6 as:

$$\begin{aligned} F(m, q) &= \quad < \boldsymbol{w}_m^T, \boldsymbol{q} > && (5.8) \\ &= \quad \left\|\boldsymbol{w}_m^T\right\| \|\boldsymbol{q}\| \cos\angle(\boldsymbol{w}_m^T, \boldsymbol{q}) && (5.9) \\ &= \quad \cos\angle(\boldsymbol{w}_m^T, \boldsymbol{q}) && (5.10) \end{aligned}$$

Where $\|\cdot\|$ is the euclidean vector norm, and the quantity $\cos\angle(\boldsymbol{w}^T, \boldsymbol{t})$ is the *cosine similarity* between $\boldsymbol{w}^T$ and $\boldsymbol{t}$. We can constrain $\|\boldsymbol{q}\| = 1$ (e.g. by dividing it by its length) and we can have also constrained vector $\boldsymbol{w}^T$ to have length 1.

In order to find the k most relevant RIRs m to query q, we (1) calculate the cosine similarities between $\boldsymbol{q}$ and all $\boldsymbol{W}$, (2) sort them in descending order, and (3) we select the first k. A block diagram of the process can be seen in Fig. 5.2.

To find the k most relevant tags for a specific RIR m we work in a similar fashion. However instead of cosine similarity on the occurrence matrix $\boldsymbol{W}$, we (1) check euclidean distances between m, characterised by a feature vector d_m that characterises the RIR,

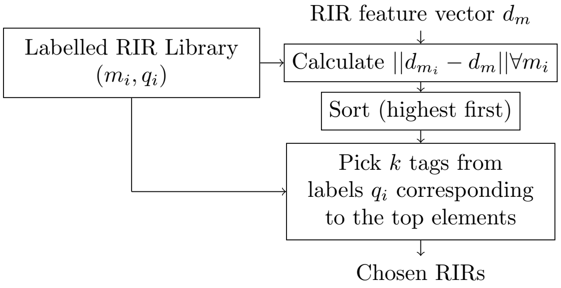

RIR feature vector d_m
Labelled RIR Library
(m_i, q_i)
Calculate $||d_{m_i} - d_m|| \forall m_i$
Sort (highest first)
Pick k tags from labels q_i corresponding to the top elements
Chosen RIRs

Figure 5.3: Block diagram for retrieving the k most relevant tags to RIR m.

and all of the RIRs λ in our dictionary, characterised by feature vectors d_λ, (2) sort them in ascending order, and (3) pick the first k tags that correspond to the top labels in that step. A block diagram of the process can be seen in Fig. 5.3. This depends on the assumption that similar RIRs are going to be labelled similarly, which we have found works in practice. For tag and query representation we use the following:

$$\boldsymbol{t}_n = v_{word} \qquad \forall n \in 1..|D| \tag{5.11}$$

$$\boldsymbol{q} = \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{t}_i \tag{5.12}$$

where t_n is the representation vector for the $n-$th word, v_{word} is the ConceptNet Numberbatch embedding vector for *word*, $|D|$ is the number of words in our dictionary, and q is the vector representation of the search query (which is a collection of tags). The factor $\frac{1}{M}$ normalises it so that $\|\boldsymbol{q}\| = 1$. Choosing the ConceptNet Numberbatch embeddings allow us to encode some commonsense similarity information in our queries. For example, when searching for a *small-room* our query will have a vector representation that is more similar to the representation of *chapel* than for *cathedral.*

5.3.2 Acoustic Features for RIR retrieval

In Section 5.3.1, we mentioned that a RIR m is characterised by a feature vector $\boldsymbol{d}_m$ and that we use similarity between a RIR m and each RIR λ in our library to retrieve the most relevant tags based on the labels of the RIRs that are most similar to m. In this section, we explain what the features in $\boldsymbol{d}_m$ are and how they are derived.

Since our RIRs are essentially audio recordings we could use methods for content-based sound retrieval similar to the ones presented in [141]. For example, by extracting the frames of the audio signal, fitting a Gaussian Mixture Model for each tag, and using the average of the log-likelihood of each model and each frame as our scoring function F to rank each RIR. This would require extracting and fitting a model for at least hundreds of frames for each impulse response.

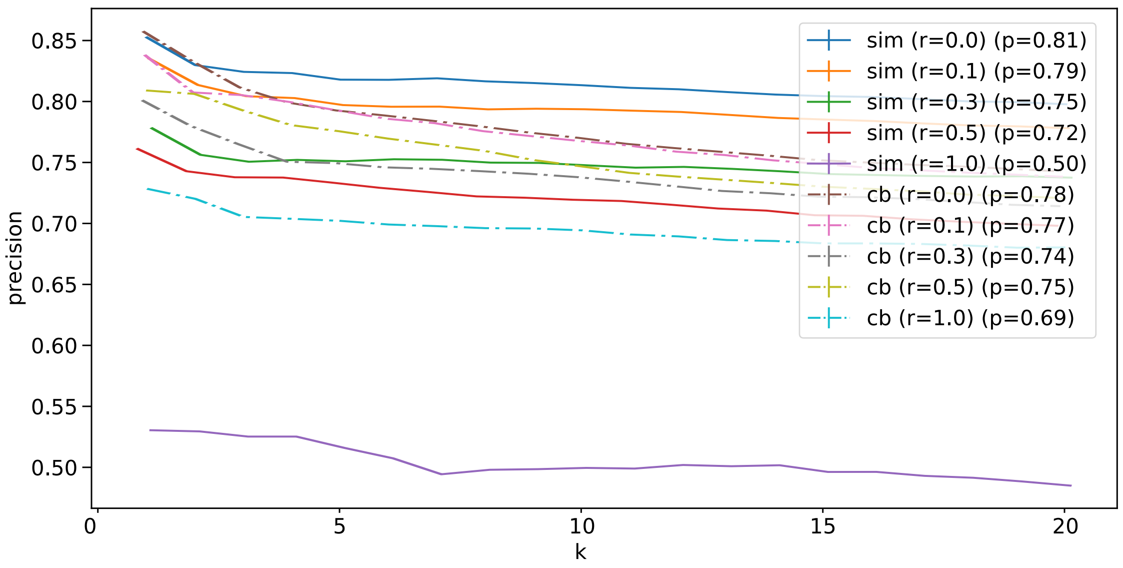
Compared to arbitrary audio files, however, RIRs have been studied extensively and a number of perceptual characteristics can be extracted that can sufficiently describe them. Instead of extracting hundreds of frames for each recording, we can therefore just extract a handful of those characteristics. There are a lot of those features to choose from. For this work we chose the perceptual characteristics mentioned in Section 4.6. To construct the vector $\boldsymbol{d}$ which characterises each RIR:

$$\boldsymbol{d} = [T_{60} \; E_d \; D_{rr} \; T_c \; S_c]^T \tag{5.13}$$

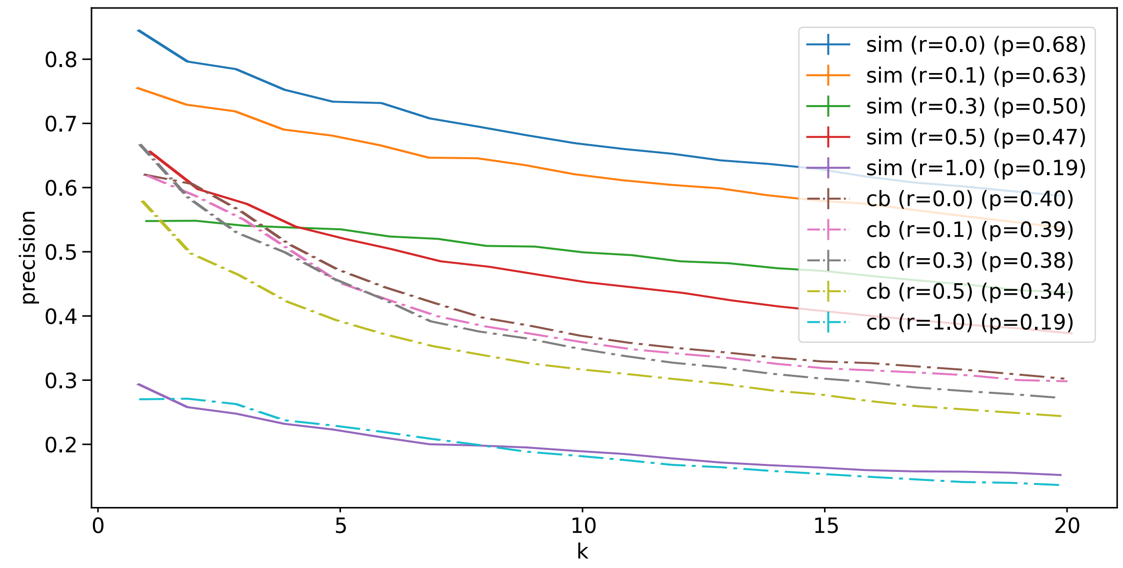
We chose those measurements over others because they can be directly mapped to the features used in [128] and can be computed equivalently for both algorithmically synthesised and recorded RIRs. They, therefore, allow us to use the dataset provided in [128, 129] with minimal effort. Definitions for each characteristic and the process through which they are derived can be found in detail in Section 4.6.

5.3.3 Experimental Results

For evaluating our tagging and retrieval method we used the dataset described in [128]. The dataset consists of the reverberation parameters and the impulse response characteristics of the effect in Section 4.5 mapped to sets of tags describing those impulse responses. The authors of [128] gathered the data online by asking users to listen and describe, using simple words, the effect of various algorithmic reverberation settings on three excerpts of piano, guitar, and drums. The reverberation architecture used was the



(a) Precision curves for retrieved RIRs



(b) Precision curves for retrieved tags

Figure 5.4: Precision curves for retrieval and tagging. Precision curves labelled as *sim* use the orthonormal basis for tag representation given in Eq. 3.3 and *cb* use the representation based on the ConceptNet Numberbatch embeddings in Eq. 5.11. Ratio r is the percentage of training data that have been replaced with synonyms. Average precision p reported is the percentage of relevant (a) RIRs or (b) tags retrieved respectively at the top k places.

Description / Filename	**Space Category**	**Suggested Tags**
. . . recording in an underground car park.	Chamber, Hall	hall, big, deep, spacey, – metallic
Has a slightly nice resonance in it from the metal pipes on the ceiling	Open Air	church, heavy, organ, slow
The Spokane Woman's Club hall is a a highly reflective space with bare walls,a hardwood floor and a curved ceiling. . .	Auditorium, Ballroom, Hall	sharp, spacious, distant, warm, echo, strong bright, electric, vibrant cool
Outback Climbing Centre	Recreation	spacey, big, room, muffled echo, deep, hollow, distant, rolling, soft
Steinman Hall	Venues	nice, heavy, clear, deep, romantic, sad, bass, warm, melancholy,love

Table 5-C: Labelling of recorded RIR. The first two rows show RIRs from OpenAIR [5] and the last two show RIRs from the EchoThief [11] library.

same to the one we discuss in Section 4.5 and was implemented in the browser using the Web Audio API. Since the dataset has been updated several times since its creation, we chose the version used in [129] which contained 6791 labellings of 256 different reverberation settings. While impulse response measurements were provided with the dataset, we resynthesised the impulse responses in order to extract the measurements described in Section 4.6 which can be used both for realistic as well as synthesised impulse responses. In order to do that, we re-implemented the reverberation effect from Section 4.5 while taking into account modifications from [128] to compensate for limitations of the Web Audio API. Those modifications consisted of adding a delay of $0.1ms$ to the dry signal and the all-pass filter and using a bi-quad filter instead of the first-order lowpass filter given in [124]. Using this implementation, we generated a room impulse response audio file for every set of parameters in the dataset in [128] and assigned those impulse response audio files to the sets of tags corresponding to those parameters. For this work, we report two separate results sets; precision of tagging and retrieval on a withheld part of the original dataset, and automatic tagging on 4 realistic impulse response recordings from two freely available RIR libraries [5, 11].

For the first case, we pseudo-randomly (using a pre-defined random seed) split our data into three equal-sized segments. We use the first two as training and development and keep the last one for testing. We report results of the methods presented in Section 5.3.1 built on both the training and development set and tested on the testing set. The first two parts were used in the process of developing our models, and the last was kept completely separate in order to assure that our reported results were not biased by our development process. Similar to [141], we report average per-query (P_q) and per-IR (P_m) precision defined as the ratio of relevant documents retrieved at the top k positions:

$$P_\nu = \frac{|relevant_\nu \cap retrieved_\nu|}{|retrieved_\nu|}, \nu \in \{q, m\} \tag{5.14}$$

where $|\cdot|$ denotes the cardinality of a set. Per-query and per-IR precision curves for $k = 1 \dots 20$ are shown in Figs. 5.4(a), and 5.4(b) respectively. Average precision p over the curve is reported for each curve. Curve labelled as *sim* denotes precision of the similarity-based method, and *cb* the method based on the Numberbatch embeddings. Ratio r is the percentage of our training labels that have been replaced with synonyms. The reason for this replacement is that there are cases in the original dataset where impulse responses, that should be labelled the same, were labelled using synonyms (e.g. *big-hall* and *large-hall*) or with highly correlated words (e.g. *church*, and *cathedral*). Ignoring these correlations leads to similar impulse responses being scored (and therefore ranked) independently. This would cause a search for a RIR with a specific tag to fail, even if the RIR can be found if searched with a similar, but not the same tag. We would like therefore to be able to retrieve that RIR even when we do not use the exact tag, but one with similar meaning. To test how our method achieves that we replace part of our training set with synonyms derived by WordNet [62] and report their precision curves on Figs. 5.4(a) and 5.4(b). We observe that though the similarity-based method performs slightly better than the method based on the Numberbatch embeddings (higher average precision p when no replacement takes place), the latter loses much less in precision when provided with synonym data (the difference between the maximum and minimum average precision p is better in the case of the NUMBERBATCH embeddings). This is due to the fact that the similarity-based method assumes that every possible label comes from a fixed, previously known dictionary and therefore cannot deal with out-of-dictionary terms.

In table 5-C we see how a system built on our method labelled four real RIRs found from OpenAIR [5] and EchoThief [11]. The first is a library of freely available impulse responses accompanied by metadata about the space and method they were recorded with, and the second is an online library of RIRs extracted from noisy environments (such as playgrounds). In the first RIR, which is from an underground car park, there is an "interesting resonance" because of some metal pipes. The system based on our method, instead of tagging with just *big* and *spacey* that relates to the car park's size, managed to tag it as *metallic*. More results on recorded RIRs on OpenAIR and EchoThief are available as supplementary material[4].

5.3.4 Conclusion

Though we showed examples of how the system performs on real recorded data in table 5-C, it would be desirable to conduct listening tests using listeners accustomed to the effect of the reverberation and preferably with experience in mixing. Such an experiment can be a MUSHRA [143] style listening test using the Web Audio Evaluation Tool [144] with careful choice of criteria for both listening subjects, as well as appropriate listening stimuli. Alternatively, Napping® [145] experiments can be conducted for the construction of a sound wheel [146]. Designing and running such experiments however is not trivial and is out of the scope of this paper. We have also constrained the number of used perceptually relevant characteristics to the ones used in the dataset from [129] and also characteristics relevant to the reverberation IR itself and not to the sound it is convolved with or to the effect it will have in the final mix. If such information were known, it could be "plugged in" as prior and posterior probabilities for tagging and retrieval. For example, [147] show that there is a strong relation between musical tempo and choice

[4] `https://code.soundsoftware.ac.uk/projects/chourdakisreiss2019aes`

of echo delay times in artificial reverberation, something which we could exploit in order to weigh tags according to the source input. On the other hand, [148] show that reverb loudness and early decay time have a significant impact on the perception of a mix. Tags, therefore, could be ranked according to the perceptual effect they have on the mix itself and not just to a single audio source. Such an extension, however, would require a dataset of reverberation parameters collected on multitrack mixes.

The methods we presented here could probably be applied to other effects as well. For example, [129, 149] show that the descriptor map used in [128] can be used for equalisation and compression as well, and [150] show that there are statistical correlations among the various tags when used in different audio effects. In the context of radio drama this could allow us to better construct retrieve EQ parameters in the same way we can retrieve reverberation IRs.

5.4 Tag-based retrieval of Music

In Chapter 2 we discussed the role of Music in radio drama production. We categorised the roles of music into *Linking*, *Mood* and *Indexal*. The last one is used similarly to a sound effect for which we discussed how to retrieve in Section 5.2. Linking, and Mood music relate to the underlying theme of the text. While we did not deal with music retrieval in this thesis, we discuss previous works that generate music from text.

Generating music from text has been studied before in the context of music composition. In [151] the authors associate relations between musical elements, such as pitch or tempo to emotions elicited from story text. They use those associations and simple rules to compose piano music. The author in [152] extends their work to 12 synthesised instruments and also makes it adhere to narrative plot structure (Section 2.2.1). While those works discuss composing, and not retrieving music, their use of semantic analysis in narrative text is closely related to the work in this thesis.

A similar study can be found in [153] where the author uses a four-stage approach that extracts semantic information of text, uses sound libraries to select sounds based on the extracted information and create compositions based on those selected sounds. It then evaluates those compositions based on the system's 'understanding' of ambient music generation. Their method suggests that it can be used for composing ambient music in the cases where thematic information (e.g. emotions) is provided explicitly in the text. While the work has been presented as a study in computational creativity, their approach to extracting emotional themes is relevant to our work. Also contrary to [151, 152], they use this information to construct queries for searching a sound library.

Taking into consideration the works mentioned above we discuss further in Section 8.3.2 how a future approach in music retrieval for radiodrama composition can be achieved.

We consider coreference resolution and text simplification as factors with two values for each factor: *yes* when it is present and *no* when absent. We tested the effect of these factors in successful to total queries made ratio, the ratio of retrieved files to total queries made, and the maximum span of successful queries. Boxplots for each task can be seen in Figure 5.1. Means and standard deviations for each factor-value pair can be seen in Table 5-A. We also performed 2-way ANOVA to check for significant differences when checking for the aforementioned ratios and span. We found that when text simplification is present, there is a significant increase in the ratios as well as a small, but statistically significant decrease in the maximum span of successful queries. On the other hand, coreference resolution does not affect significantly the results. The differences as well as p values for the ANOVA can be seen in Table 5-B.

We conclude the section therefore by observing that a text simplification pre-processing step leads to an increase in the successful queries or at least a decrease to the total number of queries made and therefore to a more efficient use of online libraries of sound effects. This is not unexpected, previous work has shown that text simplification can improve information retrieval from text [154]. Our approach in evaluation however, while indicative, is not complete due to the metrics used which do not take in consideration how useful the retrieved sound effects are in a real production scenario. For a real production scenario, a dataset constructed with the involvement of radio drama sound managers, or at least listeners of radio drama would be more appropriate. Such a dataset however does not exist at this point. We briefly discuss possible paths of its construction in Section 8.3.5.

5.5 Summary

In this chapter, we presented computational methods for retrieval of assets used in the production of radio drama. More specifically we presented methods for using the analysis done in Chapter 3 to retrieve sound effects from story sentences and reverberation impulse responses that correspond to descriptions of locations found in the text. We also briefly discussed how the aforementioned methods that utilise information extracted about emotion in the text can be extended to retrieve music.

Chapter 6

A Machine Learning Approach to Application of Intelligent Artificial Reverberation

6.1 Introduction

In Section 5.3 we discussed a method for retrieving room impulse responses based on extracted tags from the text. Such a method is adequate when information about the environment is available from the text. There are cases however that is not the case. Consider that we are writing a part where someone plays a small piece of piano (Indexal Music, see Section 2.5). The sound designer can introduce reverberation manually by controlling the parameters discussed in Chapter 4 in order to convey the perception of 'space'. Later, whether in the same drama or in a future one, when a character plays the same piano, it is desirable to automatically suggest applying the same reverberation since the piano will most likely be placed in a similar location in the future. In addition, understanding that there might be slight differences in the location each time we need the reverberation to be fine-tuned by the user. This method does not work only for musical instruments, but also-non musical speech. Our work, presented in [155, 156] and discussed below, presents and evaluates a method that achieves this functionality.

We discussed reverberation and its usage in radio drama in Chapter 4. It is important to recall that users of a reverberation effect control its parameters and tend to change these over time based on how the audio sounds. They assign specific audio features (or their changes) to specific parameters (or changes). Our goal is to simulate this process automatically using a supervised learning approach to train classifiers so that they automatically assign effect parameter sets to audio features. This way, we can train our reverberation effect to decide how to choose its parameters based just on the observed audio (e.g. choose the same reverberation settings for the same piano piece).

In order to create a reverberation effect that applies reverb automatically, we need to train it. Training can be done a-priori by e.g. an expert user of the reverberation effect, or on-line by the user of such an effect. Training is a process that involves user-interaction with the effect and so the parameters to be trained must make sense to the user. In Section 4.5 we discussed a reverberation effect architecture that uses such mapping.

Audio sources can be characterised by a multitude of features. Musical instrument tracks, for example, can be characterised by timbre, tempo, etc. An automatic reverberator trained on a set of audio is expected to be able to apply reverberation correctly on similar audio. For this reason, in order to create a reverberation effect that is as general as possible, we need to train it to a large and diverse set of audio data.

In order to train our system, we perform feature selection to select the best features from a 31-dimensional feature space from 8 features found in the literature. Smoothing is then applied to the resulting features. We then compare 4 different classifiers on the classification task where our samples are vectors of audio features and classes are the parameter-set clusters. The training data consists of the control parameters provided by the user with a simple interface that allows them to control a simple reverberation effect. Testing is performed using cross-validation and multi-stimulus MUSHRA-style [143] tests.

6.2 Previous Work

There has been a lot of research in Adaptive Digital Audio Effects for automatic multi-track mixing but in almost all cases they focus on achieving a pre-specified goal. Parameter automation and intelligent control have been applied to many of the most popular audio effects (e.g. gain and faders [157], equalisation [158], panning [159] and dynamic range compression [160]), but to the best of the authors' knowledge it has not been attempted on artificial reverberation. Furthermore, all of the aforementioned approaches except [157] which uses Linear Dynamical Systems to estimate mixing weight coefficients, use fixed rules, rather than rules that learned from training data. On the other hand, to the knowledge of the authors, there are no published works on Automatic Application of Reverberation.

Key work for the current paper can be found in [124, 125] where they present the mapping from the reverberation parameters to measurements of the reverberation. In that paper, the authors do not go as far as to provide a mapping from the measurements to the parameters, but they allow the control of the reverberation effect using high-level descriptive terms. Similar work can also be found in [161] where the authors present a real-time feedback delay network (FDN) reverberator that allows control of perceptually relevant descriptors.

Work using semantic descriptors can be found in [142] where they use a Reverberation Effect among others for their Semantic Audio Feature Extraction (SAFE) project, which allows users to assign high-level descriptive terms to low-level audio feature changes that are caused by effect parameter changes. In a similar fashion, [128] created a map of high-level descriptive terms that correspond to low-level reverberation effect parameters. Relevant work in [162] performs classification for drum sounds in order to control effect parameters, but still relies on fixed rules.

6.3 Effect Architecture

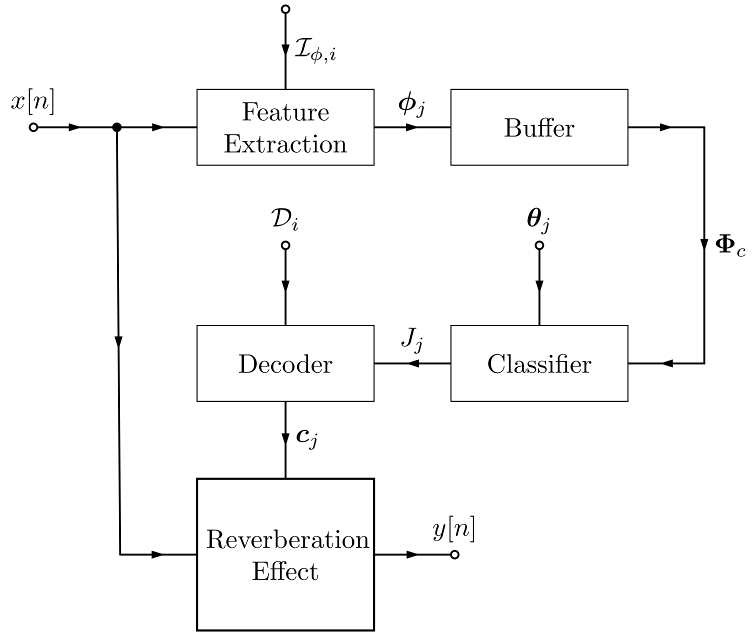


Figure 6.1: Reverb application

Our proposed design uses the traditional adaptive DAFX design [163] limited to one track and can be seen in detail in Figure 6.1. It consists of an algorithmic reverberation effect where the values of the parameters are decided by a classifier model. The classifier model can be trained on-line or off-line. The architecture of the model training process can be seen in Figure 6.2 where $\boldsymbol{\phi}_i$ is the feature vector of the i-th frame, $\boldsymbol{\Phi}_i$ is a matrix of features which consists of the vertical concatenation of the feature vectors (as row vectors), from frame 1 to frame i. In a similar fashion, $\boldsymbol{p}_i$ is the vector of desired characteristics of the impulse response provided by the user, $\boldsymbol{c}_i$ is the low-level filters parameter vector to which $\boldsymbol{p}_i$ are mapped. $\boldsymbol{\theta}_i$ is the classifier parameter vector returned as a result of the training after the i-th frame and $\boldsymbol{\mathcal{D}}_i$ a dictionary that maps class labels to reverberator parameter sets.

Note that, several features require the accumulation of a number of samples in a

Parameter (Unit)	**Controls**	**Min**	**Max**
T_{60} (s)	$60dB$-Reverberation Time	0.02	4
D (echoes/s)	Echo Density	1000	10000
D_{rr} (dB)	Clarity	-20	10
T_c (s)	Central Time	0.01	2
S_c (Hz)	Spectral Centroid	200	$f_s/4$

Table 6-A: Perceptual Characteristics of the Impulse Response. In this chapter, we assume that these characteristics are set by the user using UI elements and therefore each of the parameters has the minimum and maximum allowed value shown above. f_s is the sampling rate.

buffer (i.e. spectral features) to be computed. In such cases, latency equal to the size of the buffer × the size of the frame is introduced. Similarly, some classifier models require the accumulation of several values before being able to make a decision and therefore introduce latency equal to number of previous values × buffer size × size of each frame. Consequently, our architecture, although implementable in real-time, can introduce latency that depends on the features chosen, as well as the models used. Therefore one should be careful in their choice of features and classifier model.

In this chapter, we use the reverberation effect we discussed in Section 4.5 due to the simplicity of its architecture and the fact that it can be trained directly from measurements of the reverberation. While this design has stereo input and output, we use it with monophonic signals split to stereo. We retain the stereo output in order to have a more natural-sounding reverberation effect. The parameters of the reverberation effect (the gain and delay coefficients of each filter in the architecture) and their limits can be seen in Table 4-A. Those parameters are directly mapped to characteristics of the reverberation impulse response (Table 6-A). We recall from Section 4.7 that a mapping between measures of reverberation and DSP filter coefficients can be found by solving the numerical problem in Appendix A.2.

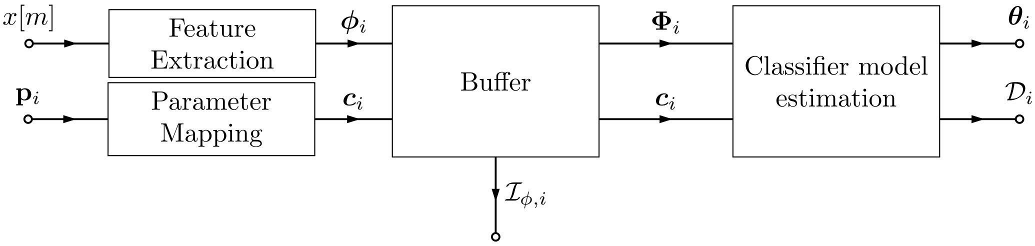


Figure 6.2: Training of classifier models

Feature	**Used in**
ZeroCrossingRate	Instrument Identification Source Identification
13 *MFCCs*	Instrument Identification Genre Classification
12 *Spectral Contrast Coefficients*	Instrument Identification Genre Classification
Root Mean Square	Instrument Identification Voice/Music Discrimination Audio Activity Detection
Crest Factor	Instrument Identification
Spectral Centroid	Instrument Identification Genre Classification
Spectral Roll-off	Instrument Identification Genre Classification
Spectral Flux	Instrument Identification

Table 6-B: Used features and their usage in the literature.

6.4 Feature Extraction

Application of reverberation to a track can depend on the instrumentation, the type of music and the percussive-ness of the track, among others. For our task, we use 8 different features. Their names and their role can be seen in Table 6-B [164–166]. The reason for choosing these features is because they have been used extensively in the literature for classification of instruments based on the above characteristics.

Before extracting the features from our audio, we first split the audio into 23ms frames (1024 samples at 44.1Hz) using the onset-based audio segmentation method described in [167] which is based on the *Spectral Contrast* feature. The reason for choosing this kind of segmentation, as shown in the original paper, is that it appears to give higher classification accuracies for at least the music-genre classification task. We then concatenate our features into a 31 dimensional vector[1] for each frame. Next, we use Principal Component Analysis to filter out non-separable or noisy features and reduce our feature vectors' dimensionality [168].

6.5 Classification and Training

We use classification on the audio features in order to control the values of the reverberation effect parameters. Given short excerpts of audio tracks together with the desired reverb characteristics, for training (Figure 6.2):

1. Convert the given reverberation characteristics $\boldsymbol{p}_i$ of values $[T_{60,i}, D_i, D_{rr,i}, T_{c,i}, S_{C,i}]^T$ to a set of filter parameters $\boldsymbol{c}_i = [d_{1,i}\ d_{a,i}\ g_{1,i}\ g_i\ G_i]^T$ and add $\boldsymbol{c}_i$ to a set $\mathcal{C}$. $\mathcal{C}$ is the set of *parameter classes* and its cardinality $|\mathcal{C}|$ is the number of parameter classes. $\mathcal{C}_m$ denotes the m-th element of $\mathcal{C}$.

2. Assign a label J_i to the i-th frame if the chosen parameters for that frame belong to a class in $\mathcal{C}$:

$$J_i = \sum_{k=1}^{|\mathcal{C}|} \left(k \cdot \delta\left[||\mathcal{C}_k - \mathbf{c}_i||\right]\right) \tag{6.1}$$

   $||\cdot||$ denotes a vector norm and $\delta[\cdot]$ is Kronecker's delta. Each number J_i is the class label for the i-th frame. Keep a dictionary structure $\mathcal{D}_i$ for the classes introduced up to that point, comprised of the parameter sets and the labels to which they correspond:

$$\mathcal{D}_i = \{(J_k, \mathcal{C}_k) : k = 1, \ldots, |\mathcal{C}|\} \tag{6.2}$$

[1] *MFCCs* and *Spectral Contrast* features have 13 and 12 dimensions respectively.

3. Segment the audio excerpts into frames and calculate a 31-dimensional feature vector $\boldsymbol{\phi}_i$ for each frame:

$$\boldsymbol{\phi}_i \quad = [\phi_{1,i} \ \phi_{2,i} \ \ldots \ \phi_{31,i}]^T \tag{6.3}$$

4. The vectors $\boldsymbol{\phi}_i^T$ are vertically concatenated to form a matrix $\tilde{\Phi}_i$ which is smoothed across columns with a Gaussian window (as a column vector of 41 elements). We then perform principal feature analysis [168] on the resulting matrix to derive $\boldsymbol{\Phi}_i$. We save the selected column numbers as the set $\mathcal{I}_{\phi,i}$.

5. Use matrix $\boldsymbol{\Phi}_i$ together with the labels J_i to estimate the parameters $\boldsymbol{\theta}_i$ of the chosen classifier.

From the training stage above we store the dictionary $\mathcal{D}_i$ and the classifier parameters $\boldsymbol{\theta}_i$. For the application of reverberation:

1. Segment the audio track, to which we want to apply reverb, into frames and calculate a feature vector $\boldsymbol{\phi}_j$ for each frame. For each $\boldsymbol{\phi}_j$, we keep the rows the numbers of which are in $\mathcal{I}_{\phi,i}$.

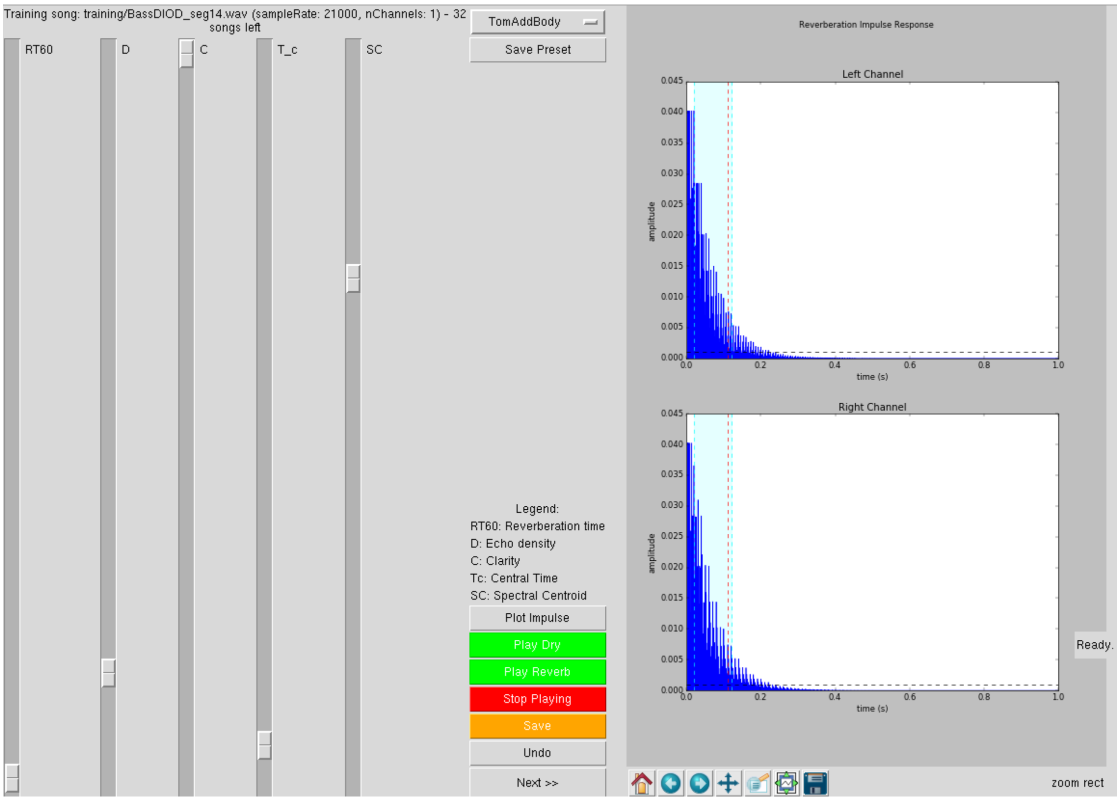
2. Use the classifier to select a label J_j for the j-th frame.

3. Use the dictionary structure $\mathcal{D}_i$ derived in the training phase as a function in order to convert from class labels J_j to reverberation effect parameters $\boldsymbol{c}_j$ for each frame:

$$\boldsymbol{c}_j = \mathcal{D}_i[j] \tag{6.4}$$

where $\mathcal{D}_i[j] = \mathcal{C}_{J_j}$.

We compare 4 different classifiers: Gaussian Naive Bayes Classifier[169, p. 217], One-vs-All Linear Support Vector Machine (SVM) Classifier [170], Hidden Markov Model Maximum A-posteriori Classifier [169, p. 610], and a hybrid HMM classifier with obser-



Training song: training/BassDIOD_seg14.wav (sampleRate: 21000, nChannels: 1) - 32 songs left

Legend:
RT60: Reverberation time
D: Echo density
C: Clarity
Tc: Central Time
SC: Spectral Centroid

Figure 6.3: A screenshot of the user interface used for training. The user is presented with a list of audio segments and is asked to add reverb according to their preference. These preferences are saved and used as training data.

vations taken from a set of SVMs [171]. Each classifier can be completely described by a vector of parameters $\boldsymbol{\theta}$. Given a set of training data, each of these classifiers is trained in a different way (usually using a variation of the Expectation-Maximisation algorithm) to estimate their parameters.

In order to train our classifier models, we use excerpts from 254 audio files taken from the Open Multitrack Testbed [172]. First, the audio data is segmented into meaningful parts (i.e. song phrases, guitar solo parts, etc.) using a similarity matrix and a novelty function as found in [166].

A user is presented with a simple GUI (Figure 6.3) where they can listen and apply reverb to the extracted parts by choosing the characteristics of the reverberation seen in Table 6-A. For the purpose of conducting listening tests, we asked 3 people familiar with the effect of reverberation to train our models. The segmented parts are split into frames using the method described in [167] and a tuple of features and parameters are extracted for each frame as described in Section 6.4. Features are filtered with a low-pass filter. The resulting data set is used to train the models described in Section 6.5. All our models were implemented in the Python programming language using the SciPy [173] library for Machine Learning and the Essentia library [174] for onset segmentation, feature extraction and storage[2].

6.6 Results

We tested our models both by measuring classification performance, as well as conducting a multi-stimulus MUSHRA-style [143] listening test.

6.6.1 Classification Performance

In order to validate our models, we split our data into 6 sets in order to reduce training times. Every set included 45 audio files except the last which included 29. Every file was a part of a Bass, Keyboards, Vocals, Percussion or Saxophone track. The files were randomly split into sets. In order to validate our classification scheme, we define the weighted macro f_1-score for K classes:

$$f_1 = \sum_{k=1}^{K} \frac{n_k}{n} \cdot \frac{2tp_k}{tp_k + fp_k + fp_k} \tag{6.5}$$

In the definition above: n_k is the number of samples belonging to class k, n the total number of samples, tp the number of samples classified correctly as belonging to class k, fp the number of samples classified incorrectly as belonging to class k, and fn the number of samples that belong to class k but classified incorrectly to some other class. We validate our models as such:

1. Split every set into 10 parts.

[2] Supplementary material for this research can be found at `https://code.soundsoftware.ac.uk/projects/chourdakisreiss2016`

2. Use 9 parts for training and 1 for testing. Do this for every combination of 10 parts. Store the predicted labels as well as the metrics tp, fp, and fn for every run.

3. Measure the weighted macro f_1-scores for the predicted values.

We also use cross-validation to estimate the most suitable Markov chain number for our sequential models, as well as the number of Gaussian components for the case of the HMM with Gaussian emission distribution. Using the full training set, we can see the overall *weighed f_1-scores* in Table 6-C, and the average *Mean Squared Errors* in Table 6-D.

Train. Set	$\lvert\mathcal{C}\rvert$	GNB	SVM	HMM	HMM$^{\text{SVM}}$
1	7	0.79	**0.82**	0.70	0.70
2	9	0.80	**0.81**	0.69	0.49
3	6	**0.81**	0.79	0.75	0.73
4	8	**0.78**	0.77	0.65	0.59
5	7	**0.82**	**0.82**	0.73	0.60
6	7	**0.87**	**0.87**	0.73	0.52

Table 6-C: Average weighted f_1-scores. Highest scores for each set are in bold. $|\mathcal{C}|$ is the number of classes calculated for each set.

Tr. Set	$\lvert\mathcal{C}\rvert$	GNB	SVM	HMM	HMM$^{\text{SVM}}$
1	7	0.0067	**0.0065**	0.0114	0.0117
2	9	0.0015	**0.0010**	0.0025	0.0045
3	6	**0.0091**	0.0097	0.0106	0.0096
4	8	**0.0014**	**0.0014**	0.0035	0.0062
5	7	0.0082	**0.0047**	0.0069	0.0135
6	7	0.0044	**0.0041**	0.0066	0.0204

Table 6-D: Mean Squared Errors for the normalised parameters. Lowest MSEs for each sets are in bold. $|\mathcal{C}|$ is the number of classes calculated for each set.

The high f_1-scores are important because they represent the rate of agreement, between the automatic reverberation effect and the user that trained it, on the parameters of the reverberation. Mean squared error effectively measures how far the estimated parameters are from the parameters chosen by the user. This means that while the clas-

User	$\|\mathcal{C}\|$	GNB	SVM	HMM	HMM$^{\mathrm{SVM}}$
A	30	**0.79**	0.73	0.06	0.11
B	22	**0.74**	0.66	0.17	0.16
C	32	0.81	**0.84**	0.12	0.18

Table 6-E: Weighted f_1-scores for the user-trained models. Highest scores for each user are in bold. $|\mathcal{C}|$ is the number of classes calculated for each user.

User	$\|\mathcal{C}\|$	GNB	SVM	HMM	HMM$^{\mathrm{SVM}}$
A	30	**0.0104**	0.0138	0.0510	0.0568
B	22	**0.0141**	0.0226	0.0538	0.0386
C	32	**0.0087**	0.0091	0.0444	0.0480

Table 6-F: MSEs for the user-trained models. Lowest MSEs for each user are in bold. $|C|$ is the number of classes calculated for each user.

sification accuracy may be high, so the effect and the users agree most of the time, the differences on the parts they do not agree may be too high for the model to be useful. Therefore, the most useful model is the model with the least mean squared error. In our case, the multi-class SVM approach performs best regarding MSE in all but one cases, while it performs similar to the GNB in regards to f_1-scores.

6.6.2 Perceptual Evaluation

Perceptual evaluation of the data was performed using multi-stimulus MUSHRA-style listening tests in the Web Audio Evaluation Tool(WAET)[144]. This was in order to check how our models performed when trained by different users of the reverberation effect.

For this test, we used 33 audio files from our dataset. We normalised them in regards to mean loudness and converted them to mono. We used 3 expert users of the reverberation effect from the Centre of Digital Music, to train our system by applying suitable reverberation to each of them. For each of the "trainers", we kept the parameters they used for 27 of those files and trained our models as described in Section 6.5 (Classification performance for each of those models can be seen in Tables 6-E and 6-F). Using the GNB and SVM models for each trainer, we then applied automatic reverberation to the 6 remaining files. These files consisted of excerpts from two singing tracks, a bass guitar, a saxophone, a drum, and a piano track.

For each file, we created a multi-stimulus trial. Each of the trials included a visible outer reference (the original file with reverberation applied manually by one of the three "trainers"), the same reference hidden in the stimuli, and an anchor (the original file with no reverberation applied to it). It also included 6 files with automatically applied reverberation (one from a GNB model, and one from a SVM model, for each of the three "trainers" that trained those models). Subjects were asked to rate each of the stimuli in regards to how close it sounds to the reference.

Sixteen test subjects participated in the listening test. Those did not include the "trainers". They were mostly PhD students and Post-doctoral associates from the Centre for Digital Music at the Queen Mary University of London, with the exception of one student not from the Centre, and a freelance employee. Three listeners were active users of the reverberation effect (two of them professionally), while the rest just knew what the effect sounded like. The average time of the test taken was 30 minutes and the test was considered difficult by most participants. The tests were all done using WAET in local mode on the desktop computer of the Media and Arts Technology studio control room at the same university.

Figure 6.4 shows the mean rating, averaged over all participants, and the 95% intervals, for each stimulus in each trial. If our reverberator was successful, we would expect each model to be rated close to its respective reference, e.g. `A-gnbc` or `A-svmc` should be close to the reference for `A-sax`. For `C-drums`, `A-sax`, and `A-voice-1` we can see that `C-smvc`, `A-gnbc`, and `A-svmc` score higher than the rest. For the case of `B-bass` we see that while the `B-` models were not rated closer than the rest, `B-svmc` is still very close to the reference. For `B-piano`, the models seem to perform poorly, while for `A-voice-2`, the `A-` models seem to have failed. In general for this small listening test, the tracks based on `-svmc` models appear more similar to the respective tracks with reverberation applied by the trainers. The listening test however fails to give very clear results. We suspect

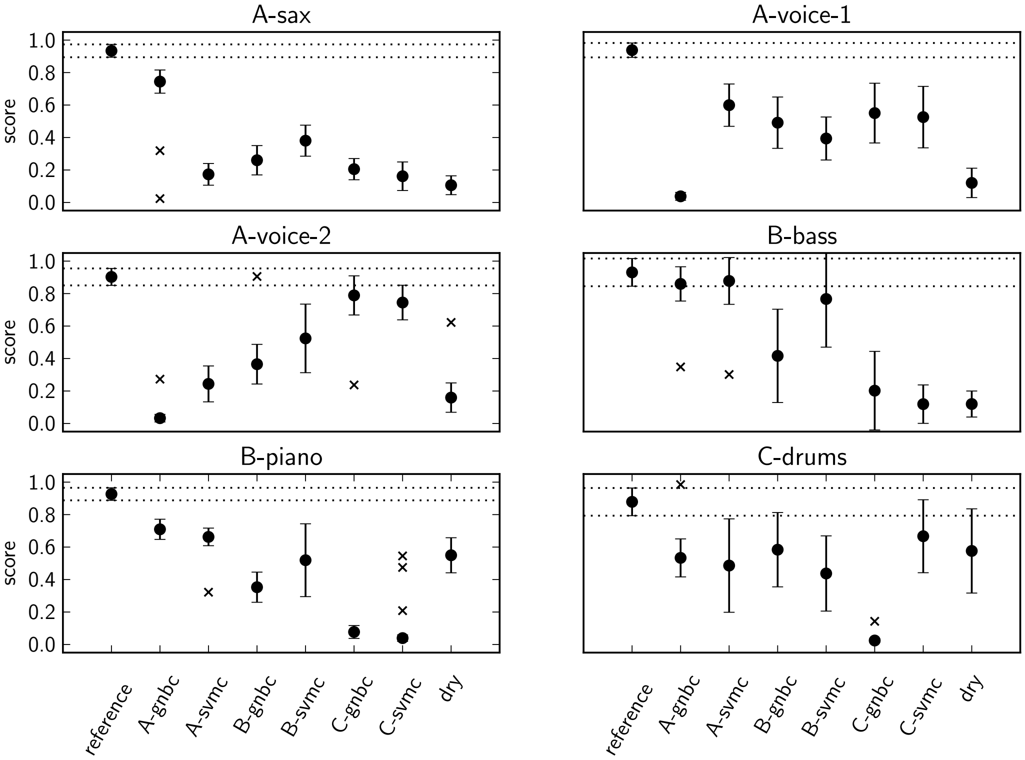


Figure 6.4: Results of the MUSHRA-style tests. The bars represent the upper and lower limits for the 95% confidence intervals. Full circles are mean values, `x` symbol points are outliers, and dotted lines represent the upper and lower standard error borders of the reference. On the x-axis are the labels of the stimuli. Each of the letters `A`, `B`, or `C` represents a reverberated track generated from a model trained by the corresponding expert. Suffixes `-svmc` and `-gnbc` represents whether it was based on a Support Vector Machine or a Gaussian Naive Bayes classifier.

this was due to the difficulty of the question and the different concept of similarity for each subject.

6.7 Conclusion

From tables 6-C to 6-F we can see that for our datasets, the non-sequential models performed better than the sequential counterparts, which performed comparably or even worse than the Naive Bayes classifier. This suggests that the Hidden Markov Models failed to capture correctly the temporal progression of our data. One of the reasons for this could be the onset segmentation method we use prior to feature extraction, which leads to uncorrelated feature vectors, as opposed to classical frame segmentation and thus damaging the Markov assumption. The disparity between the sequential and non-sequential classification results in Section 6.6.2 can also be attributed to the large number of classes that were produced as a result of the training by the users (and as a result, the smaller number of training samples for each class). The above suggest further exploration with different models and configurations. The best model so far seemed to be the One-vs-All Support Vector Machine classifier which performed best regarding weighted f_1-score and Mean Squared Error. Our choice of models becomes clearer when we take into account that our simple non-sequential models do not require past samples in order to make a decision, so we can use our models in real-time with a minimum latency of 23 ms (a simple frame). This paper, in general, described an approach on a reverberation effect that could control a reverberator given desired characteristics of the impulse response, and also remember those characteristics in the future. An implementation for the method described is an audio effect that allows the user to select desired reverberation characteristics to be applied to specific tracks, and have the system suggest similar reverberation for newly introduced, but similar tracks, scenarios that adhere to the goals set in Section 6.1.

6.8 Limitations

While this initial approach appears promising, there are things to be desired regarding individual steps. Mathematically deriving characteristics of the reverberation does not necessarily lead to perceptually correct parameters. For example, impulses that are very closely placed together may not be perceived as distinct echoes, but Eq. 4.17 will count them as such. Figuring out more perceptually robust reverberation features will greatly improve this work. Another issue is that we did not take into account features relating to stereo signals such as Interaural Cross-Correlation, Lateral Energy Fraction, Apparent Source Width, etc. Future research could take the direction of providing mappings between such features and low-level parameters of a stereo reverberator. The architecture of the reverberator itself can be of concern. The Moorer reverberator, while serving as a good basis for our work given its simple design, is limited (for example, it does not allow for independent control of early and late reverberation). One could try to exchange the current architecture with a more recent reverberator design [123] or even try to implement a model agnostic architecture so that it could be used with commercially available reverberation effects. Adept [175] provides a framework that could aid in the design of such a system. Regarding perceptual evaluation, there is work to be done on how to efficiently evaluate such systems. Our question on how "similar" the tracks with automatically applied reverb sounded to the reference, was deemed very difficult to answer by our test subjects which made drawing conclusions difficult. One should use a more clearly defined objective for testing (e.g. reducing masking in a multi-track context).

The original Adaptive Digital Audio Effect architecture [163] supports multitrack DAFX, while our method has only been tested for effects applied on a single track. A logical next step would then be to extend our architecture to multitrack audio content. Finally, the ability of our effect to be trained directly from measurements of reverberation could allow it to be trained directly from impulse responses or even reverberant sound samples. There are numerous works in the literature that would allow us to estimate Reverberation Time [176–179], Echo Density [127], Clarity/Definition [178], Central Time and Spectral Centroid. [180] also gives an easily measurable set of features which correlate to subjective reverberation and which could be included with small alterations to our model.

Chapter 7

Rendering, Mixing, and Mastering of Radio Drama

7.1 Introduction

In this chapter, we complete the portfolio of proposed assistive methods with methods for rendering a radio drama, given a script generated in Chapter C and the asset retrieval methods discussed in Chapter 5.1. Initially we discuss how such a script produces tracks for a DAW timeline to be further processed by a mixing engineer. Based on this approach, we present perceptual evaluation results of simple story renders and we examine the importance of each element in a mix. We then discuss how those tracks can be automatically grouped according to their narrative importance to allow the listener to control the mix at their device.

7.2 Using templates of synthesised speech

This section describes a method to allow a radio drama production team to easily oversee speech 'takes'. Such a system is helpful when the entirety of the team is available in the day of the recording. There are cases however that this requirement is not satisfied.

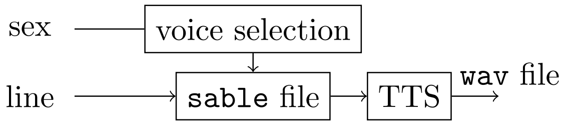


Figure 7.1: Sable construction and voice generation. The parameters *age* and *line* are extracted by the methods discussed in Chapter 3

The production team might be just a single person, for example, in the case we are talking about students of radio drama, or a team collaborating through the internet. In such cases the actors might not be immediately available but we would still like for the producer to be able to get a rough idea for how the drama would sound. In such cases, we use a speech synthesis engine that allows for control of timing and minimal expressivity controls. While such engines are still nowhere near human control of voice expression, this is not necessarily limiting since [18, p. 242] advises against a director offering the actors their representation of performance. Deep learning has provided recent methods for synthesising naturally sounding speech such as WaveNet [181], TacoTron [182], WaveGlow [183], and Deep Voice [184, 185]. All the aforementioned methods allow to 'replicate' existing voices with natural sounding speech synthesis. Those works however, lack control for voice characteristics such as volume, pitch, or accent (Section 2.4), although efforts for prosodic control exist [186]. Furthermore, the aforementioned approaches require a large quantity of 'source' voices and most are used with two or three different voices. For these reasons we use the 'classical' text-to-speech framework Festival [187] with the voices available from the Carnegie Mellon University's Festvox [188] project. The version used is 2.4, which includes TTS voices of 15 US English speakers. Furthermore, Festival allows some control over prosody using the Sable [189] XML-based markup language. Sable allows for speaker directives to be introduced to text and thus control elements of the speaker such as rate of speech, pauses, pitch, loudness. It can also correctly transform to speech elements such as dates and numerical elements, and assign the correct prosodic contours to the spoken sentences as well. The process is as follows:

1. Identify the character's *age* as well as their dialogue line using the method described

in Section 3.3.6.

2. Choose the voice from a list of known male and female Festival voices.

3. Construct a `.sable` file with a single tag:

   ```
   <SPEAKER name="<VOICE>">
   <LINE>
   </SPEAKER>
   ```

   where `<VOICE>` is the voice selected in step 2, and `<LINE>` the dialogue line.

4. Feed the `.sable` file to the Festival TTS system to generate a single PCM `.wav` file containing the spoken line.

5. Align them in a draft mix of the radio drama at the time when the character's line should be heard.

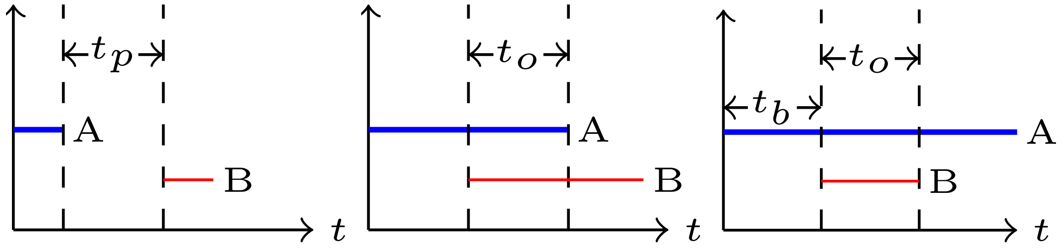
The method described above will let an aspiring producer to 'listen' to their radio drama even without the existence of lines spoken by real actors (e.g. because they experiment with other aspects of radio drama). A system using a method such as the one offered in Section 3.3.6 will then allow them to include actor voices at a future stage.

7.3 Rendering & Mixing

This section describes how a script created, using the methods shown in Chapter C, can be rendered into individual tracks of a DAW timeline. We begin with the concrete elements of radio drama: speech, music and sound effects. Then we discuss transformations on these elements such as transitions and use of reverberation and EQ.

7.3.1 Speech

The characters take 'turns' speaking and listening and each 'turn' is preceded by a conversational pause. Furthermore, to convey the feeling of conversation, stereo panning



(a) no interrupt (b) second speaker interrupts (c) second speaker fails to interrupt

Figure 7.2: A DAW timeline corresponding to three turn-taking situations in dialogue. A and B correspond to two different characters. In (a), t_p is the duration of the pause between A stops talking and B starts talking. In (b), t_o is the duration of overlap between A and B. (c) is characterised by a time t_b which is the offset at which B tries to, but fails to interrupt A.

is employed with the following cases:

- The narrator and the main character are always positioned at the centre of the stereo field.
- The other characters are positioned left and right based on their order of appearance in the script.

Those decisions are taken from the instructions on how to create perspective in [3, p. 137]. An example resulting DAW timeline can be seen in Figure 7.2(a). A more tricky decision is choosing the duration of pauses between individual lines, in cases where the lines have not been recorded as a dialogue e.g. when synthesised (Section 7.2). Duration for the turn-taking pauses discussed depend on the content of the dialogue. For example, [190] show that there is a difference between spontaneous face-to-face and telephone conversations. In this iteration of rendering methods, we do not make the distinction between the two. Furthermore, even in the same setting dialogue turn-taking can differ significantly. [12] have distinguished ten different cases in which turn-taking can take place. For example, a dialogue might proceed like in Figure 7.2(a) but also the second participant might try to interrupt the speaker and succeed or fail to do so, etc. We consider the following cases:

1. Successful turn. There is a pause P and the two lines do not overlap [12, Case 1] (Figure 7.2(a)).
2. Successful turn with a short overlap [12, Case 2](Figure 7.2(b)).
3. The second participant tries to interrupt but is unsuccessful [12, Case 5a] (Figure 7.2(c).

The first case is the default state of dialogue. A pause of mean duration $t_p = 380ms$ is inserted between speech segments A and B [12, Fig. 2]. The second case is denoted in the radio drama script by a dialogue line that ends with ellipses `(...)` or dashes `(--)` and another dialogue line that begins right after:

```
CHARACTER #1
The character says something that is interrupted--


CHARACTER #2
The second character begins at the point where the
previous character stopped talking.
```

The mean duration of the overlap for English is found to be $t_o = 257ms$ [12, Fig. 2]. We give an example for the last case:

```
CHARACTER #1
The character speaks for a long time


CHARACTER #2
(overlaps)
The character tries to interrupt CHARACTER #1 but is unsuccessful.
```

Where the `overlaps` parenthetical should be programmed as a directive. In this iteration of our method, this case is not implemented since more research needs to be done on deciding the exact position the overlap starts. The logarithm of the pause and over-

Variable	μ	σ
$\log_{10} t_p$	2.58	0.49
$\log_{10} t_o$	2.41	0.49

Table 7-A: Gaussian distribution parameters for the pause and overlap durations for dialogues in the English language, as found in [12].

Character type	**Peak level** (dB)	**Avg level RMS** (dB)
Narrator	−16.70	−34.16
Character (Nathan)	−21.60	−58.23
Character (Amelia)	−5.90	−41.40

Table 7-B: Loudness levels of Narrator and Characters in *The Turning Forest* [13]. We use those as a starting point for *Narrator*, *Main Character*, and *Secondary Character*.

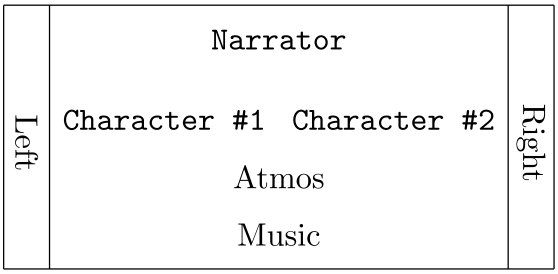


Figure 7.3: Example perspective hierarchy card of characters that can aid the actors when recording, or a mixing engineer when setting the levels and stereo panning for the character voice tracks.

lap durations have been found to follow a Gaussian distribution [12]. We repeat the distribution parameters in Table 7-C for convenience.

The volumes of the narrator and the rest of the actors are normalised to have a peak level in dB according to Table 7-B. This is an arbitrary decision but provides a 'good enough' starting point. Additionally, we construct a 'perspective hierarchy card' (Section 2.6.2) that gives an immediate impression of the acoustic hierarchy of characters. An example of such a card can be seen in Figure 7.3 and can aid both the actors when recording (e.g. to help decide their positions relative to the microphone) or a mixing engineer when setting stereo panning and loudness for the character voice tracks.

Type	Representation	Duration
Short pause	`(short pause)`	$1s$
Normal pause	`(pause)`	$2s$
Long pause	`(long pause)`	$5s$

Table 7-C: Types of pauses, their representation in the script and the duration they correspond to in seconds

7.3.2 Pauses

We briefly discussed the role of pauses in Chapter 2. Pauses play a multitude of functions, we discussed their role in 'turn-taking' in radio drama dialogue. Another important role is in transitions between scenes that take place in different locations or time, which we discuss in Section 7.3.5. Here we only discuss pauses that are inserted explicitly, for example for dramatic effect. Those are presented in the script in parentheticals and are distinguished in *short*, `long` pauses, and uncharacterised (normal) pauses:

```
CHARACTER #1
The character, e.g. a doctor is about to say something important,
e.g. whether the patient will live or die.


(short pause)


CHARACTER #1
The character finishes their speech.
```

The durations for each are chosen arbitrarily and are shown in Table 7-C.

7.3.3 Music

Rendering of music is done straightforwardly. As discussed in Section C.1.5 music can be given with the `MUSIC:` prefix, followed by a query given to the music recommender system, and an optional duration. In the case where no duration is specified, the music plays until the end of the scene or the end of the music piece, whichever comes first.

In the case a duration has been specified, the music starts fading when it has reached 3/4ths of that duration. Those numbers have been arbitrarily chosen but future work should be systematically derive them from current practices.

7.3.4 Sound Effects

Sound effects fall into two categories: backdrop sound effects, or ATMOS, and event-related sound effects. Those are given in the script using the `FX:` prefix, optionally followed by `SOUNDSCAPE:`. If the line starts with `FX: SOUNDSCAPE:` then a description of a soundscape is fed to the soundscape creation system discussed in Section 5.2 and the resulting sound effects are looped in the background, otherwise it just plays once. Sound effects are also actions in parentheticals that appear outside dialogues and are not pauses or comments, e.g:

```
CHARACTER #1
Spoken line

(a hammer hits)

CHARACTER #2
Spoken line
```

Where the description of the sound effect is passed as a query to the method in Section 5.2. Sound effects that are not soundscapes are affected by reverberation.

7.3.5 Transitions

Type	Script text	Duration
A cut	`(HARD) CUT TO:`	$4s$
Crossfade	`CROSS FADE TO:`	$5s$
Fade out/in	`FADE TO:`	$12s$

Table 7-D: Duration of transitions in seconds. The durations for each type of transition are derived from [14] and [3, p. 159].

Transitions between scenes should take from 4 to 15 seconds [14]. We discussed fading times in Section 2.7.1. We choose 5 seconds when cross-fading (`CROSSFADE TO:`) and 12 seconds when a fade-out/in occurs (`FADE TO:`) [3, p. 159]. For a cut, we chose the minimum transition duration of 4 seconds [14]. Those numbers are seen in Table 7-D. Fadeout affects music and ATMOS but does not affect non-ATMOS sound effects or speech. As an example, if the following part in the script:

```
INT. SCENE 1 - AN OFFICE ROOM
FX: SOUNDSCAPE: AN OFFICE ROOM

[...]

MUSIC: LINKING MUSIC
CROSSFADE TO:
EXT. SCENE 2 - A PUBLIC ROAD
```

The first scene has an office room ATMOS (keyboards clicking, telephones ringing) playing throughout the scene. Linking music starts playing just before the crossfade begins to occur. The music, as well as the office room ATMOS, start fading out and the ATMOS of the second scene starts increasing in volume. The whole transition takes five seconds.

7.3.6 Reverberation

In case of internal (INT.) scenes, reverberation is applied to the sound effects and speeches in the scene. Scene description is used as a query for a system based on the method in Section 5.3 to retrieve a relevant reverberation impulse response which is convolved with sound effects that are not ATMOS, and character speeches (excluding the narrator). Alternatively, the method in Chapter 6 can be used to apply reverberation based on contents and not the description of the scene.

7.3.7 Equalisation

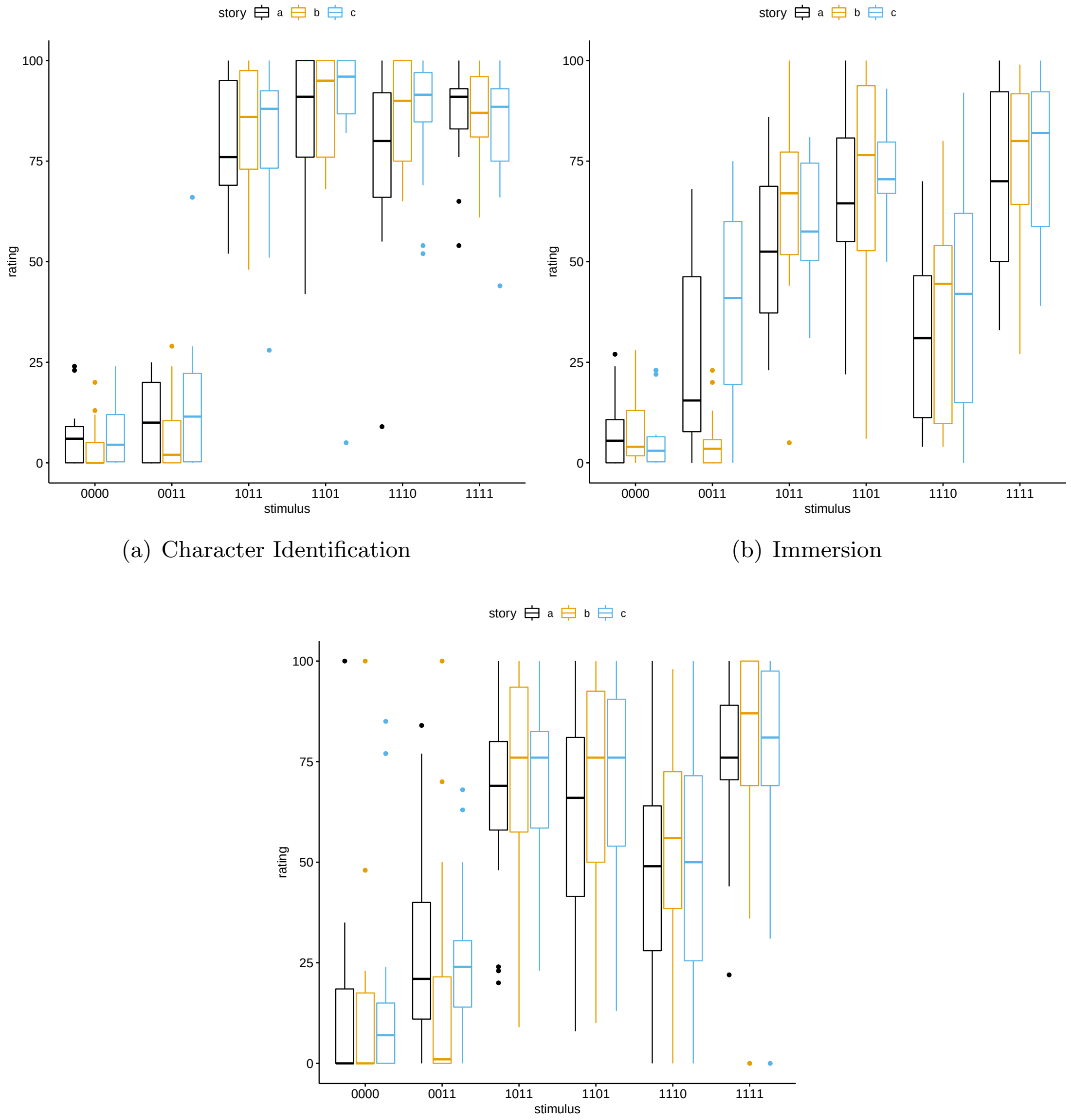
In Section C.1.2 Scenes are introduced with a INT. or EXT. which stand for *internal* or *external*, based on the type of locations the scenes take place. Sounds in external locations lose some of the lower frequency content and thus sound 'thin'. It is recommended that sounds (non-ATMOS sound effects and non-narrator speech) should have their 100Hz – 1KHz [3, p. 156] frequency attenuated. This can be achieved at placing a notch filter (Section 2.7.2) in that frequency range (e.g. at $10^{2.5} = 316Hz$). We do not discuss the exact parameters of an EQ to achieve that in this thesis. Instead, we add a note using a script comment:

```
(NB: ATTENUATE NON-NARRATOR SPEECH AND NON-SOUNDSCAPE SFX
BY PLACING A NOTCH BETWEEN 100HZ-1KHZ TO GIVE THE IMPRESSION
THAT THOSE EVENTS TAKE PLACE 'OUTSIDE')
```

Alternatively, [149] propose an EQ controlled using a library of semantic descriptors which could be adapted for the case of radio drama. However, more research on the subject is needed.

7.4 Evaluation of Script Rendering

Evaluation of the produced radio play renderings takes the form of a listening test. This subjective evaluation has the goal of identifying the extent to which the various parts



(a) Character Identification

(b) Immersion

(c) Preference

Figure 7.4: Box plots from the listening tests

of the sound production system contribute to story character recognition (task 1) and listener immersion (task 2), and how well they rank on the listeners' preference (task 3).

Each test was presented on 9 pages (3 stories for each of 3 tasks) implementing the MUSHRA[143] listening test environment using the Web Audio Evaluation Tool (WAET)[144]. This environment consists of samples that can be played in the browser,

ID	CHARA	PAN	REVB	SFX
0000				
0011				✓
1111	✓	✓	✓	✓
1011	✓		✓	✓
1101	✓	✓		✓
1110	✓	✓	✓	

Table 7-E: Evaluation segments and audio story elements they represent in Fig. 7.4. CHARA pertains to whether the story has different character voices or not, PAN whether it contains spatial panning, REVB whether it contains reverberation and SFX whether it contains spatial sound effects.

and each sample can be given a rating from 0 to 100, by using a vertical scroll bar. Scales at 0, 25, 50, 75, and 100 of the scroll bar are annotated according to each test. The samples were randomised in each page but the pages retained the same order across all participants. MUSHRA tests generally have a hidden anchor and reference as stimuli. Our tests have a hidden anchor but not a reference since it is non-trivial to generate an objective reference for our tasks.

We gathered 21 subjects, mostly non-native English speakers. Subject age was between 23-31. The subjects were also asked whether they had experience in radio/TV production, with theatre and whether they were regular consumers of radio-plays/audio-books. From the subjects, we omitted one person who reported not understanding the test.

7.4.1 Listening Segments

Three story segments were selected (Section B.2), with each having a narrator and two additional story characters. The first segment had all of the characters in the story as *male*, the second as *female*, and the third had a male narrator, a male story character, and a female story character. The choice of genders was made in order to consider character recognition based on the difference in genders between different characters since we expect both gender and voices to be important factors in recognising different characters [191]. In addition to the character voices, two environmental background sound effects were used (*a meeting room*, and *a forest*), the reverberation descriptors *clearer* and *dry* from the Reverbalize social reverberation map [149], and stereo panning. While Reverberation and Spatialisation are not very high in importance while identifying spaces [191], with elements such as actions and context being higher, our segments lack action sounds and are too short to provide a context. For the rest of the evaluation section, we will refer to Character voices, Sound Effects, Reverberation, and Panning as *audio story elements*. The segments were created by combining those elements but leaving one out each time. This approach allows us to avoid introducing a "the more, the better" bias to our listeners, something that could happen if we built each segment from the previous one with one more additional element. In addition, a hidden anchor with all elements disabled and an extra segment with all the elements enabled was used. The table of listening segments and their elements can be seen in Table 7-E.

7.4.2 Character Recognition

This task pertains to how each audio story element contributes to the improvement of the listener's ability to distinguish between 3 different characters. We expect however that the listener has already some cues from the story text. The question asked on the related pages was:

> "How easy does each segment make it to distinguish between the 3 characters (based on both sound and text)?"

The answers were on a continuous scale from 0 (*very hard*) to 100 (*very easy*). Full use of the scale was not required. If our system performs well, we expect the ratings for segments including character voices, to be rated much higher than the segments with just the narrator reading the text, and panning and reverberation to contribute to the ability of our system to convey character differences.

Results are shown in figures Fig 7.4(a). Though the bars mostly overlap, stimuli with different character/gender voices are rated much higher than the ones without. This observation is verified by a Kruskal-Wallis rank-sum test for each of the stories (Table

2-A), together with pairwise paired Wilcox tests (Table 2-B).

This appears to agree with the observation in [191]. From between the elements with sound, there is a hint of preference towards the ones with spatial panning (1111, 1101, and 1110) compared to the one without (1011). However, the pairwise Wilcox tests in Table 2-B did not show a significant difference.

7.4.3 Listener Immersion

This task pertains to how well our system can immerse the listeners in the story environment by usage of panning, reverberation, and environmental sound effects. The same segments as in Section 7.4.2 were used. The question asked was:

> "How easy does each segment make it to imagine yourself in the environment of the story?"

The question seeks to identify what elements act as cues to communicate the environment of the story to the listener. We expect environmental sound effects to contribute to listener immersion, and reverberation and panning to add to that contribution. The answers were again on a continuous scale from 0 (*very hard*) to 100 (*very easy*). The resulting boxplots can be seen in Fig. 7.4(b).

Like in the case of Character Recognition, different character voices seem to be the biggest factor facilitating immersion. Also, as in [191], sound effects seem to be an important factor since there is a large difference between stimuli with (1011, 1101, 1111) and without (1110) sound effects in the case different character voices are also there.

7.4.4 Listener Preference

The last test was a generic listener-preference test. The same segments used in the two previous subsections were used, but this time they were ranked based on how well each user preferred each one. The goal of this test was to check how much the listener liked each element. The question asked was:

> "Please listen to each segment again, how would you rank them in regards of preference?"

The answer was on a continuous scale from 0 (*very low preference*) to 100 (*very high preference*). This was the only part of the test which encouraged full use of the scale since it was checking for relative preference and not absolute user liking. The results can be seen in Fig.7.4(c). In this case, boxplots overlap too much to make clear observations. From Table 2-B however, we note that in general listeners prefer the segments with character voices (1011, 1101, 1111, 1110), although in the cases when the segment already contains reverb and sound effects the difference is not significant.

7.5 Summary

In this chapter, we provided a mechanism for producing a DAW timeline given a script of radio drama generated using the methods discussed in Chapter C. Using this mechanism, we rendered several radio drama excerpts which we evaluated based on the listener's ability to identify the different characters, the ability of the drama to 'immerse' the listener to their world, as well as their overall preference. We found that panning and reverberation indeed help to convey useful information but we did not survey how much of each effect is needed in each case. We also left out of the discussion questions relating to mixing and mastering for radio. Radio drama, as a form of art broadcast by radio, is subject to regulations for a constant loudness [192]. Methods such as for automatic dynamic range compression [160] could be adapted for radio drama. However, this is not currently examined. Automatically applying audio effects in multi-track mixes has also been tried for Panning [159], Equalisation[158], Reverberation[156], and while the context of such works are usually multitrack music mixes, we would expect future work adapting them to radio drama as well.

Chapter 8

Conclusion

8.1 Answers to the Research Questions

We begin wrapping up this thesis by answering the over-arching question asked in Section 1.6:

> **In what ways can advances in artificial intelligence and machine learning assist a creator when producing radio drama?**

Leveraging machine learning-based techniques for NLP with classical rule-based approaches for information extraction as well as external knowledge sources allows for retrieval of sound effects, music, and audio effect parameters for radio drama production directly from a source story when adapting it to radio drama. Draft mix takes can then be provided to the user using the aforementioned elements as well as templates based on historical practices. Furthermore, when mixing sound for radio drama, reverberation effect parameters can be chosen intuitively using desired spatial characteristics, the story source text itself, or even the content of the sound file to be added to the mix.

The above statement is expanded by answering the individual questions asked:

1. **In what elements can we deconstruct radio drama in order to make**

computational analysis possible?

The answer to this question comes in the form of a taxonomy of narratological elements for radio drama (Figure 2.14) as well as their function in providing the user with the auditory experience (Table 2-A). This taxonomy stemmed from previous works on radio drama as well as computational narratology.

2. **To what extent do recent advances in automatic story generation and natural language processing allow us to extract meaningful information from a story expressed in raw text? Can extracting such information be seen as a set of NLP tasks common in the literature? Can we use or devise algorithms to extract information that is either explicit or implicit? In what ways can external knowledge, i.e. knowledge elicited from online ontologies, help?**

   This question is answered in Chapter 3. Commonly studied NLP tasks can aid us in understanding a story text with the goal of producing a radio drama. Coreference resolution, Word Sense disambiguation coupled with external ontologies and a simple linear algorithm can give us information about characters, their properties, and the lines their dialogue lines and therefore provide necessary information to direct actors (Section 3.3.1). The task of Spatial Role labelling provides information about locations in the drama and thus gives information that can be used for building radio drama backdrop through the use of sound effects, as well EQ and reverberation (Section 3.4). Finally, a database of word-emotion associations can be used by a simple formula to assign emotional tags to sentences in the story and thus be able to recommend appropriate music for the drama as well as provide directions for emotional acting (Section 3.5).

3. **How can we adapt a reverberation effect to be able to apply reverberation based on desired reverb characteristics?**

   In Chapter 4, the mapping between level filter parameters (gains and delays) to perceptually relevant characteristics of reverberation was examined. Through algebraic transformations as well as supplementary numerical minimisation, a mapping from those characteristics to the low-level filter parameters was provided.

4. **Given a radio drama script that includes the elements of Research Question 1, how can assets for production be retrieved in an automatic way? Can audio effects retrieved, e.g. reverberation, be retrieved in a similar fashion?**

   Asset (e.g. sound effects) retrieval systems usually rely on querying a database using queries comprised of individual words as tags. In Chapter 5 we show that query text does not need to consist of individual precise tags, but retrieval using queries constructed directly from story text is also possible by pre-processing the sentences with a text simplification algorithm (Section 5.2). Furthermore, methods for retrieving audio files based on text queries can be used to retrieve audio effects as well which we showcase for the case of reverberation (Section 5.3). Furthermore, queries need not be limited to tags extracted from the text of story sentences. In Section 5.4 we discussed how implied emotional tags can be used to retrieve music. In Chapter 6 we also show that effect parameters can be retrieved by querying using the content of the sound that the effect needs to be applied to.

5. **Assume that RQ1-RQ4 can be answered positively and elements from a story can be effectively translated to speech, sound, music, and audio effects. How can those elements be combined to produce a final radio drama render? How can speech be timed and panned automatically to convey the effect of 'dialogue' between characters, as well as narrator speech? How should sound effects and music be introduced into the mix? Finally, how do different elements of production contribute to the listener's experience?**

   This question is answered in Chapter 7. Time arrangement of speech can be done using templates which are based on previous dialogue studies (Section 7.2). Panning can be done using simple rules that take into consideration the role of characters in the drama. Sound effects and music are mixed initially using a simple lookup table for desired loudness levels. Reverberation and EQ are applied based on a simple list of rules for applying those effects in the context of radio drama (Section 7.3). From the elements of production we examined, we found that assigning different voices to characters is the single significant factor for distinguishing characters in the mix. Different voices for characters was also a significant factor for listener immersion, together with the introduction of sound effects. Character voices and sound effects were also rated highly in the test for character preference as well (Section 7.4).

8.2 Summary

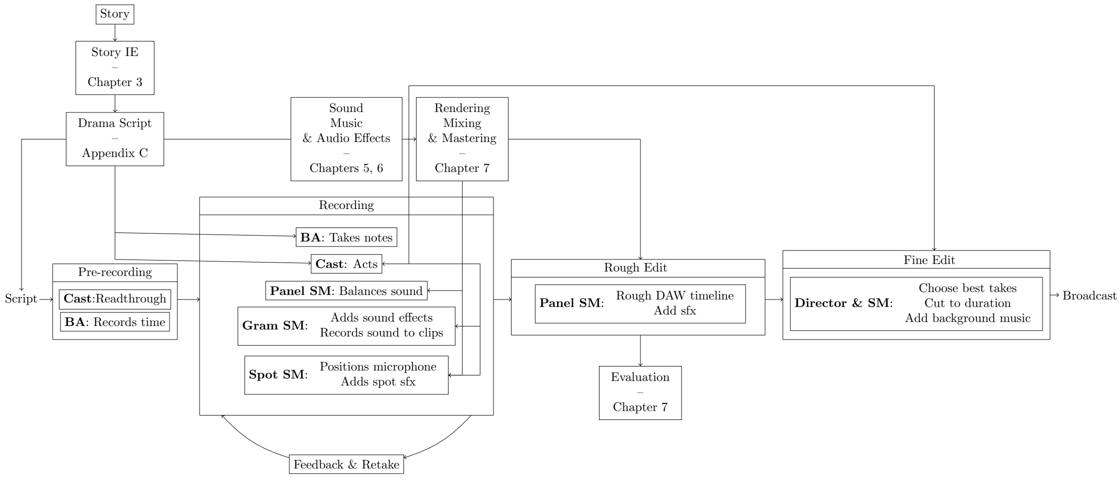

Story
Story IE – Chapter 3
Drama Script – Appendix C
Sound Music & Audio Effects – Chapters 5, 6
Rendering Mixing & Mastering – Chapter 7
Recording
BA: Takes notes
Cast: Acts
Panel SM: Balances sound
Gram SM: Adds sound effects Records sound to clips
Spot SM: Positions microphone Adds spot sfx
Pre-recording
Cast:Readthrough
BA: Records time
Script
Rough Edit
Panel SM: Rough DAW timeline Add sfx
Fine Edit
Director & SM: Choose best takes Cut to duration Add background music
Broadcast
Evaluation – Chapter 7
Feedback & Retake

Figure 8.1: An abstraction of the methods discussed in this thesis and the parts in the production process they affect.

This thesis introduced a toolchain of computational methods that can aid radio drama students, small producers, or even bigger teams to move through the process of creating a radio drama faster. Given a literary story for adaptation, it aids the user in the process of adapting to a radio drama script by identifying the important elements in a story, organising assets such as music and sound effects, speeding up and organising voice recordings and finally mixing and mastering the drama. A graph, summarising the discussed method and their roles in the production process, can be seen in Figure 8.1. In our knowledge, it is the first effort to combine advances both in language processing as well as in the audio processing domain that pertains to the complete production process of a popular art form.

At first glance, it might seem that the entirety of this thesis goes against the warnings of Tim Crook, who posited [18, p. 160]:

> "Detailed deconstruction and analysis of the human alchemy of creativity contains the risk of establishing cultural conventions which become oppressive and mutually exclusive to the members of the production hierarchy. I do not believe you can discover and produce great radio drama through formulas."

We respect the above concerns and we do not claim to establish a dictionary of techniques to mimic human creativity in radio drama. Instead, at every step, we try to make sure the agency of the human creator is respected; each of the methods we discussed can be ignored for the sake of artistic freedom, without drastically affecting the methods that depend on them. Radio drama, as a form of art, has produced a large variety of techniques throughout its history. We do not aspire to replicate them all but we expect that the methods introduced in this thesis provide a basis upon which to build, for researchers that aspire to do so.

8.3 Limitations/Future Work

In this section, we discuss aspects which we mentioned throughout the thesis but did not get into sufficient details. We hope that both this work provides a reasonable starting point for future exploration on computational methods for radio drama.

Tags	**Associated with**
`happy`, `uplifting`	`joy`
`dramatic`, `melancholic`	`sad`
`nature`	Locations related to nature
`dramatic`	Suspenseful events

Table 8-A: Associations between the tags given in [15] and tags or other information derived from text.

8.3.1 Object based mixing for hard of hearing listeners

This thesis focused on assisting an individual or a team throughout the production aspect of radio drama. The last link however in the radio drama pipeline is the listener themselves. As we discussed in Chapter 2, radio drama competes with other activities for the attention of the listener and this competition is unforgiving, once the attention is lost it is very difficult to be regained. An efficient way to establish the listener's attention is to keep spoken text simple, which can be done with Text Simplification (Section 3.7). Despite efforts to keep text simple, however, the attention is still lost if other non-important elements mask over speech insofar to make it hard for the listener to understand. This notion is also true for non-spoken elements such as important events that are communicated non-verbally. The effect becomes more intense in the case of the listener being affected by partial hearing loss, which is the focus of [193–195]. In those works, object-based mixing approaches are employed by a sound producer to allow the listener to control a radio or TV mix in order to allow them to perceive important audio elements more easily. In [196] we extended such works in order to perform such mixes automatically something that resulted in increased speech intelligibility for the automatically mixed drama examined in that paper. Future work could expand automatic mixing for speech intelligibility to other metrics that more directly affect the attention of the listener and thus become part of the radio drama production toolchain in this thesis.

8.3.2 Retrieving Music by extracting emotional theme from text

Our approach to recommending music is based on searching in a sound library annotated with emotional tags. In Section 3.5 we presented how those emotions can be extracted from narrative text. The library we use is described in [15] and is a library of freely distributed songs annotated with 59 tags for mood/theme. While those do not correspond directly to the emotional tags from [100] which we use, we restrict searching only for the tags that are directly related to the emotional tags found in that database. As an example we associated the tags 'joy' in the dataset in [100] to 'happy', and 'uplifting' in [15] as well as the tags 'sad' in [100] to 'dramatic' and 'melancholic' in [15]. The tag 'dramatic' is also associated with suspenseful events. Additionally, we associate the tag 'nature' whether a location in the text is related to nature (e.g. a meadow). The latter associations are extracted using ConceptNet (Section 3.2) using the query:

```
    http://api.conceptnet.io/query?rel=/r/RelatedTo&start=/c/en/nature&
end=<location>
```

where `<location>` is the extracted location. The associations are given in Table 8-A. Having the library given in [15] while keeping only the tags of the left column of Table 8-A allows us to retrieve music using the query-based approach we used in Section 5.3. The associations of Table 8-A are, however, superficial and evaluation requires either listening tests which have not been performed at the point of writing or user query data for quantitative evaluation such as the one performed in Section 5.3.

8.3.3 Text Simplification for radio drama scripts

Text simplification, as presented in Section 3.7, is extractive and rather crude: it is based on creating shorter sentences by recombining parts of a source sentence without any modification for readability or any concern whether the resulting sentence makes sense. This type of summarisation is called *extractive summarisation* and reuses part of the original sentence. A different approach would be using *abstractive summarisation* techniques which generate new sentences which in turn can be constrained further.

Exploring such techniques could lead to text simplification for more natural scripts.

8.3.4 Altering speech with the Personage NLG system

In a generated script (Chapter C) we convey emotions in the dialogue lines, by simply adding the appropriate emotion in a parenthetical. There are cases, however, where it is more natural as well as appropriate to give some cues for the emotion in the dialogue line itself. For example, shyness or hesitation can be conveyed by repeating part of the word in the text [197, Fig. 4]:

```
THE MOUSE
(hesitating)
I don't kn- know!
```

Transforming text according to transformation parameters such as *shy* or *angry* is the subject of [197]. They use the story representation format in [40, 198], formal rules and the PERSONAGE NLG system [199] to apply personality traits to excerpts of Aesop fables. Future work could explore how can such a system be adapted to the methods described in this thesis in order to generate speech text that adheres more closely to the emotions extracted from the text.

8.3.5 A dataset for story text-based sound effect retrieval

In Section 5.2, we did preliminary work on how text simplification and coreference resolution can affect sound effect retrieval when queried using story sentences. We concluded the section by briefly mentioning the need for a dataset appropriate for evaluating such retrieval effort. Such a dataset might be gathered by experts, i.e. radio drama sound managers, or listeners of the radio drama. Items in the dataset could be pairs of sentence story text associated with the appropriate sound effects. Additionally, Audio Event Detection techniques could be used to identify sound events that correspond to the input query, in the final radio drama scene itself.

8.3.6 Aiding with movement, 3D sound

In Section C.1.4 we discussed how the narrator and characters are positioned in a stereo recording of a stereo field. This discussion was limited to characters that are static in relation to one another, they do not move. We also avoided talking about the positions of sound effects in stereo sound. Data and knowledge elicited from the S3A[1] project [13] can be combined with methods from 3D synthesis with natural language, e.g. in [200] to construct 3D auditory scenes based on the content of the story.

8.3.7 Combining reverberation approaches

We presented two separate methods for adding reverberation based on content. In Section 5.3 reverberation is added based on story locations while in Chapter 6 reverberation is added based on the audio content of the asset itself. However, the methods are separate and it is not entirely straightforward how those can be combined. A simple approach would be to use the methods shown in [128] or Section 5.3 to decide initial parameters from the story itself, for content used in the drama. Then those parameters and the audio in the drama can be used to train the reverberation effect in Chapter 6 to apply reverberation in other cases where no such information is available (e.g. in a new drama). However, more research on how such methods can be combined is needed.

8.3.8 Controlling dramatisation

In Section 2.1 we referred to the difference between the various forms of audio drama: *sonic art*, *radio drama*, and *audiobook*. We also discussed its relation to the three main elements of radio: *music*, *sound effect*, and *speech*. It would be interesting to be able to allow our methods to control how much of each element we want to use when producing an audio drama. This would effectively mean that elements that are usually communicated with sounds (such as events) in a radio drama, in an audiobook they would be narrated. Another interesting idea would be to allow to switch between different types of narrator (e.g. between intra- and extra-diegetic), definitions that we briefly mentioned but did

[1] `http://www.s3a-spatialaudio.org/about-s3a`

not expand in Section 2.4.2.

8.4 Closing remarks

In the final section of this document, we want to address the question: “Why choose to examine each element in the creation of radio drama separately? We have seen recent approaches generating fictional articles [201] or poetry [202] by just feeding data to a cleverly-designed neural model, why not *learn* radio drama making from data?”. There is indeed increasing optimism in the artificial intelligence community stemming from the impressive achievements of deep learning methods which seem to improve by simply providing more data: a notable example being newer language modelling approaches [203]. The achievements of those approaches are not followed, however, by any exceptional examples in computer-generated art. Even in the cases where end-to-end approaches are used, careful data-preparation and curation of the final output are warranted by a human author. We believe this goes much further than not having enough data to generalise yet. After all, art is a social endeavour and any creation of such a system should allow the human creator to retain agency. This is difficult with the black-box data-intensive approaches currently leading the state of the art and is a problem independent from the level of generality achieved. Narrative for example, one of the elements of radio drama, demands a multi-aspectual approach at analysis [204]. While there are methods to regain some control (e.g. how Generative Adversarial Networks allow control of aspects such as perceived emotion in images [205]), those have yet to be showcased from data-driven methods in the domains this thesis examines. Even ignoring those issues, such approaches don’t currently perform sufficiently for elements crucial to radio drama, such as modelling suspense where, at least to our knowledge, no method tries to model it from surface text, in an end-to-end fashion (Section 3.8). Instead of delegating the creation of radio drama to such successful AI techniques in an end-to-end fashion, we believe that the focus should be into examining their functions and limitations on the separate steps of the creative process. Once those are sufficiently mastered, we believe it will raise our chances of being able to create radio dramas and similar art forms in a semi-automatic, way.

Finally, more focus should be paid into studying computer-assisted art in cases where different forms of art are employed together. Take music and audio effects in the case of radio drama as an example which are to elevate the listener's experience as we showed in Chapter 1. Research in automatic or computer-assisted music composition for or adaptive audio mixing specifically for radio drama is currently lacking, as we observed in that chapter. On their own, automatic composition has been studied on its own extensively since at least the 50's [206] and adaptive audio mixing since the 70's [29]. When studied through the prism of creating a radio drama, however, the intricate relations between the story of the drama, music, and audio mixing, add a new challenging dimension to those domains. Exploring those elements in the context of radio drama might lead to new interest directions of research that might improve how we treat music or audio effects, using artificial intelligence.

Appendix A

Reverberation

A.1 Calculating T_{60} from the Energy of Impulse Response Decay Curves

In Section 4.6 we mentioned measuring T_{60} from the energy of the decay curves of the impulse response. Here we provide more details on this method. The method consists of the following steps:

1. Take 8 frequency bands of the impulse response centred around $C_f = \{63, 125, 250, 500, 1000, 2000, 4000, 8000\}Hz$.

2. Calculate the energy of the decay curve of each band using backward integration of the normalised squared impulse response:

$$E_f(t) = \frac{1}{\|y_f\|_\infty} \int_\infty^t y_f^2(\tau) d\tau \tag{A.1}$$

   where y_f is the room impulse impulse response band-filtered around $f \in C_f$. The energy reduction in dB is given by:

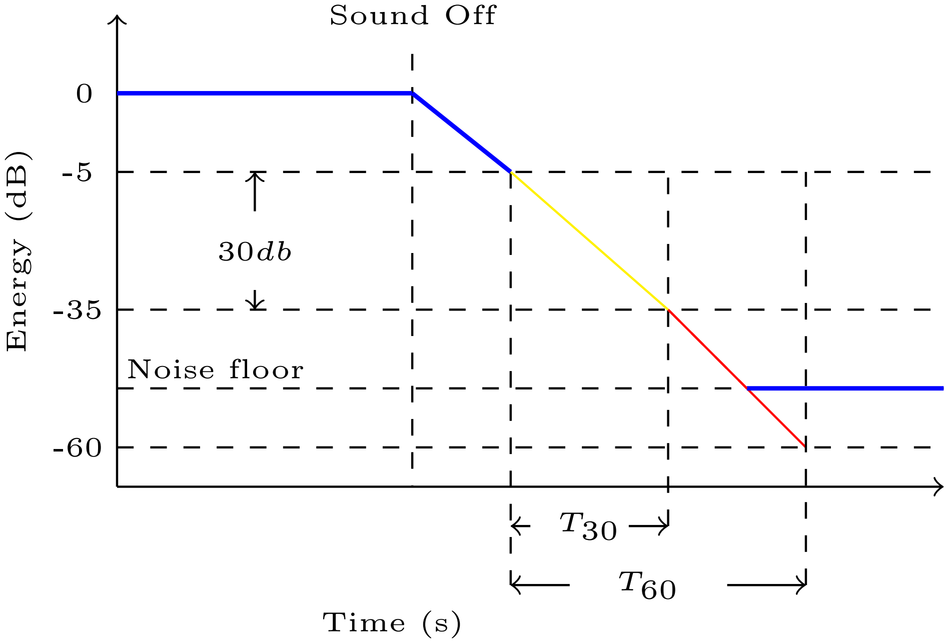


Figure 1.1: Estimating T_{60} by measuring T_{30}. Initially, the energy of the decay curve (blue line) is calculated. The points of the curve between -5 and $-35dB$ are used to fit a line whose slope d gives the energy decay rate in dB/s (yellow curve segment). The time taken for the yellow segment to drop by $30dB$ is T_{30} and T_{60} is double that time.

$$E_{f,dB}(t) = 10 \log_{10} \frac{E_f(t)}{\|E_f\|_\infty} \tag{A.2}$$

3. Using linear regression, calculate a line segment that matches the decay curve more closely in the range $[-5, -35]dB$. The fitted line has the equation:

$$E_{l,dB} = dt + b \tag{A.3}$$

where the slope d gives the energy reduction in decibels as a function of time, and b is the intercept of the calculated line. We can compute the times the energy to reach $-5dB$ and $-35dB$:

$$t_5 = \frac{-5-b}{d} \tag{A.4}$$

$$t_{35} = \frac{-35-b}{d} \tag{A.5}$$

and therefore the time it takes for the energy to drop $30dB$ is:

$$T_{30} = t_{-35} - t_{-5} = \frac{-30}{d} \tag{A.6}$$

The time it takes for the energy of the decay curve to drop by $-60dB$ is that time doubled.

$$T_{60} = 2T_{30} = \frac{-30}{d} \tag{A.7}$$

A.2 Mapping from Filter Parameters to Impulse Response Characteristics

In Section 4.7 we approximate the solution to a 4 × 4 system that maps characteristics for the impulse response, such as reverberation time and echo density, to filter gains and delays of the Moorer reverberator. Here we show how we derive the approximated system shown in Eq. 4.27 from the original 4 × 4 system shown in Eq. 4.26. Trying to solve the system of equations analytically, we found that[1]:

1. From D'_{rr} and g_1 we can derive G:

$$G = f_1(D'_{rr}, g_1) = Ae^{-\frac{D'_{rr}}{20}\log(10)} \tag{A.8}$$

2. From D'_{rr}, T'_{60} and g_1 we can derive d_1:

$$d_1 = f_2(D'_{rr}, T'_{60}, g_1) = \frac{T'_{60}\log(g_1)}{B} \tag{A.9}$$

[1] Note that g_c and g_a are treated as constants.

3. From E'_d, D'_{rr}, T'_{60}, and g_1 we can derive d_a:

$$d_a = f_3(D'_{rr}, T'_{60}, E'_d, g_1) = T'_c - \frac{\sum_{k=1}^{6} \frac{2.078125 \cdot 1.5^{1-k} \Gamma^{2 \cdot 1.5^{1-k}}}{D' d_a \left(1 - \Gamma^{2 \cdot 1.5^{1-k}}\right)^2}}{\sum_{k=1}^{6} \frac{\Gamma^{2 \cdot 1.5^{1-k}}}{1 - \Gamma^{2 \cdot 1.5^{-k+1}}}} \tag{A.10}$$

where:

$$\begin{aligned} A &= g_a \sqrt{(1+g_c)(1-g_c)} \sqrt{\frac{1}{\sum_{k=1}^{6} \frac{g_1^{2 \cdot 1.5^{-k+1}}}{1 - g_1^{2 \cdot 1.5^{-k+1}}}}} \\ B &= \log\left(\frac{0.001 e^{\frac{D'_{rr}}{20} \log(10)}}{A}\right) \\ \Gamma &= e^{\frac{2.078125}{D' T'_{60} d_a} B} \end{aligned} \tag{A.11}$$

So if we could pick the correct value of g_1 and we have the target values T'_{60}, D'_{rr}, E'_d, and T'_c we can derive the other 3 parameters. Unfortunately, it is non-trivial to find a closed-form solution but we find a value for g_1 numerically given our constraints. We can rewrite the optimisation problem above as:

$$\begin{aligned} & \underset{g_1}{\text{minimise:}} \; f_0(g_1) = \sqrt{\boldsymbol{e}^T \boldsymbol{e}} + \text{Var}[\boldsymbol{e}]^2 \\ & \text{subject to:} \\ & \qquad 0 < g_1 < 1, \\ & \quad \text{where:} \\ & \qquad G = f_1(C', g_1) \\ & \qquad d_1 = f_2(C', T'_{60}, g_1) \\ & \qquad d_a = f_3(C', T'_{60}, D', g_1) \end{aligned} \tag{A.12}$$

We can numerically compute g_c (e.g. using Newton's method). The rest give us a (non-convex) 4 × 4 system. Given a set of target IR characteristics $(T^+_{60}, D^+, C^+, T^+_c, S^+_c)$ we approximate a set $(d'_1, d'_a, g'_1, g'_c, G')$ that brings the actual characteristics close to these values (see Section 4.7).

Appendix B

Evaluation of Script Rendering

B.1 Statistical Analysis of the Results

In Section 7.4 we report on the results of the listening tests for three tasks: Character Recognition, Listener Immersion, and Listener Preference. Here we report inferential statistics that verify those observations. For every task and story, we test for the effect of the different stimuli on the user rating. In all tasks and stories the data acquired violate both normality (Shapiro-Wilk test $p > 0.05$) and equal variance (Hyun & Feldt $\tilde{\epsilon} < 0.85$) and therefore it is not suitable for parametric tests (e.g. parametric ANOVA). Instead, we perform a Kruskal-Wallis non-parametric omnibus test followed by pairwise Wilcox post-hoc tests for the effect of individual stimuli on user rating. The statistics for the omnibus test can be seen in Table 2-A and for the pairwise tests in Table 2-B.

Task	**Story**	χ^2	*df*	p
1	1	67.624	5	3.196×10^{-13}
	2	60.729	5	8.592×10^{-12}
	3	54.039	5	2.057×10^{-10}
2	1	45.997	5	9.097×10^{-9}
	2	55.395	5	1.083×10^{-10}
	3	47.288	5	4.963×10^{-9}
3	1	47.388	5	4.736×10^{-9}
	2	53.994	5	2.102×10^{-10}
	3	54.117	5	1.983×10^{-9}

Table 2-A: Kruskal-Wallis [16, p. 204–215] rank-sum test for the three tasks. χ^2 is the chi-squared statistic, *df* the degrees of freedom and p the p-value of the test. A $p < 0.05$ corresponds to a significant difference between at least two stimuli. There is a significant difference between the stimuli for all stories for all three tasks.

Task	Story	1					2					3				
	Stimulus	0000	0011	1011	1101	1110	0000	0011	1011	1101	1110	0000	0011	1011	1101	1110
1	0011	0.1707					0.7220					0.4487				
	1011	0.0032	0.0032				0.0058	0.0058				0.0122	0.0135			
	1101	0.0032	0.0032	0.9546			0.0058	0.0058	0.722			0.0122	0.0122	0.4487		
	1110	0.0038	0.0034	0.9546	0.9546		0.0058	0.0058	0.722	0.722		0.0122	0.0122	0.4487	0.4487	
	1111	0.0032	0.0032	0.9546	0.9546	0.7938	0.0058	0.0058	0.722	0.722	0.722	0.0122	0.0122	0.4487	0.4487	0.4487
2	0011	0.1882					0.5729					0.0377				
	1011	0.0150	0.0783				0.0136	0.0131				0.0017	0.0719			
	1101	0.0165	0.0018	0.3719			0.0136	0.0131	0.6377			0.0131	0.0251	0.0658		
	1110	0.0142	0.5508	0.0679	0.0232		0.0476	0.0167	0.0659	0.1013		0.0230	1.0000	0.0950	0.0315	
	1111	0.0142	0.0150	0.2015	0.5508	0.0333	0.0149	0.0131	0.2941	0.6377	0.0131	0.0017	0.0470	0.1664	1.0000	0.0032
3	0011	0.1962					0.1962					0.1450				
	1011	0.0150	0.0143				0.0150	0.0143				0.0047	0.0021			
	1101	0.0143	0.0494	0.4331			0.0143	0.0494	0.4331			0.0030	0.0030	0.7322		
	1110	0.0408	0.4331	0.0622	0.3823		0.0408	0.4331	0.0622	0.3823		0.0310	0.1450	0.0912	0.1412	
	1111	0.0143	0.0143	0.4331	0.4293	0.0184	0.0143	0.0143	0.4331	0.4293	0.0184	0.0095	0.0030	0.7322	0.7322	0.0192

Table 2-B: Pairwise paired Wilcox tests with Hochberg's correction for pairwise testing for all three tasks and stories [17]. A p value less than 0.05 corresponds to a significant difference between the two stimuli.

B.2 Story Segments of the Listening Tests

In Section 7.4 the tests conducted had three parts (tasks) and each task used three different story segments each consisting of a narrator and two different characters. The story segments are listed in script form in Figures 2.1, 2.2, and 2.3.

```
CAST LIST:
YOUNG MOUSE: male, young, animal.
OLD MOUSE: female, young, animal.

SCENE 1. INT. THE PROPOSAL -- A CONFERENCE ROOM
FX: SOUNDSCAPE: A CONFERENCE ROOM

(long pause)

YOUNG MOUSE
By this means we should always know when she was
about and could easily retire while she was in the
neighbourhood.

NARRATOR
This proposal met with general applause until an old
mouse got up and said:

OLD MOUSE
That is all very well but who is to bell the cat.
```

Figure 2.1: Excerpt from “Belling the Cat” rendered for the listening tests in Section 7.4. Script metadata is omitted. When rendering for the listening tests, reverb settings, panning settings, and soundscape settings were manually chosen instead of automatically fetched.

```
CAST LIST:
JUPITER: male, god.
VENUS: female, god.

SCENE 1. EXT. ARGUMENT BETWEEN VENUS AND JUPITER -- NEXT TO A RIVER
FX: SOUNDSCAPE: A RIVER FLOWING

JUPITER
See.

NARRATOR
Said Jupiter to Venus

JUPITER
How becomingly she behaves. Who could tell that yesterday she was but a Cat? Surely her nature is change

VENUS
Wait a minute...
```

Figure 2.2: Excerpt from “The Cat Maiden” rendered for the listening tests in Section 7.4. Script metadata is omitted. When rendering for the listening tests, reverb settings, panning settings, and soundscape settings were manually chosen instead of automatically fetched.

```
CAST LIST:
THE FOX: female, animal.
THE CAT: female, animal.

SCENE 1. EXT. THE FOX BRAGGING -- IN A FOREST
FX: SOUNDSCAPE: A FOREST

NARRATOR
A Fox was boasting to a Cat of its clever devices for escaping its enemies.

THE FOX
I have a whole bag of tricks.

THE FOX
Which contains a hundred way of
escaping my enemies.

THE CAT
I have only one...

NARRATOR
Said the Cat.

THE CAT
But I can generally manage with that.
```

Figure 2.3: Excerpt from “The Fox and the Cat” rendered for the listening tests in Section 7.4. Script metadata is omitted. When rendering for the listening tests, reverb settings, panning settings, and soundscape settings were manually chosen instead of automatically fetched.

Appendix C

A Regular Grammar for Radio Drama Scripts

In this chapter, we introduce a regular grammar for radio drama scripts based on the Fountain[1] format for Film and TV scripts and also provide a method for adapting a source story to this format. Introducing such a grammar serves the following:

1. It allows to easily present the stories in this thesis in a format that can be read by a radio drama producer.

2. It can be easily parsed by a Context-Free Grammar parser and automatically produce rough mixes of a radio drama for the producer.

We begin by introducing the Fountain format which we base the grammar on. We continue by presenting the elements of the radio drama that we need to take in consideration and how those are added in the original format. Finally, we present the derived format as rules for an Extended Context-Free Grammar and we give an example of a radio drama script written in this format.

[1] `http://www.fountain.io`

Terminal	**Regular Expression**	**Description**
int	`[1-9][0-9]*`	An integer number
sp	`:sp:`	A single space character
nl	`:nl:`	A character that changes to a new line
nb	`[A-Za-z0-9'@:sp:;*!:,._-?"]`	All characters, numbers, punctuation and space except `:nl:`
name	`[A-Z0-9:sp:-"_@]+`	Capitalised names
tag	`[a-z0-9-]+`	Tags (e.g. emotional tags)
ac	`[A-Z0-9'"*!m,:sp:_0;:]`	Same as *nb* but only allows capital letters
date	multiple	A regular expression for dates such as 11/12/2019 or December 11, 2019

Table 3-A: Common regular expressions used in this chapter

C.1 Adapting the Fountain Format for Radio Drama

In its original form, the Fountain script format allows parsing and editing for the following elements:

- Metadata – Title, Author, Information about the adaptation, Draft number, etc.
- Scenes – Placement (internal, external), Description, etc.
- Characters – Speech, Description
- Dual Dialogue – When two characters speak at the same time.
- Actions – Character actions, events, etc.
- Transitions – Scene transitions, such as 'hard cut to...'
- Scenes and Acts – Organising the script into Acts of Scenes.
- Parentheticals – Short modifiers of actions, character speech given in parentheses.
- Notes – Extra 'comment' information. Useful for example for reviewing script drafts.

Observe that there are a few elements from Chapter 2 that relate directly to radio drama and are missing from the list (e.g. Music and Sound Effects):

- Music – Intro, Outro, Linking, or as Sound Effect.
- SFX – Event SFX, ATMOS, etc.

Below we present how the Fountain script format allows us to both generate scripts according to the elements extracted in Chapter 2 as well how the same script format is used to guide production, for each individual element. The parser for the grammar was generated using LARK[2] which generates Earley parsers [207].

C.1.1 Metadata

While we did not explicitly discuss metadata, they play a very important role in production. Apart from giving information about the play and the author, draft and contact information are included in order to facilitate communication between the members of e.g. a small production team or between e.g. a teacher and the student of radio drama. Fountain allows us to convey the following metadata:

- **Title** – The title of the script.
- **Author** – Who wrote that.
- **Source** – Was it adapted from a literary story?
- **Draft Date** – Date of the draft.
- **Contact** – Contact details of the author.

The above can be written in the form of an extended context-free grammar:

```
Metadata → Title Author? Source? Date? Contact?
Title → ``TITLE:''i nl? (nb nl)+
Author → ``AUTHOR:''i nl? (nb nl)+
Source → ``SOURCE:''i nl? (nb nl)+
Date → ``DATE:''i nl? date nl
Contact → ``CONTACT:''i nl? (nb nl)+
```

[2] `https://github.com/lark-parser/lark`

where *date*, *nl*, and *nb* are given in Table 3-A and the literals in teletype (e.g. `title`) correspond to the string literals in double quotation marks ‘‘, ’’. As an example, the *Metadata* grammar rule above will capture the following script metadata:

```
Title:  The Crow and the Fox
Author: Emmanouil Theofanis Chourdakis
Source:
        Adapted from Aesop’s Fable:
        "The Crow and the Fox"
Date:   5/12/2019
Contact:
        Emmanouil Theofanis Chourdakis
        e.t.chourdakis@qmul.ac.uk
```

In general, in this work we do not deal with metadata. However, it is trivial to use the above rules generatively to e.g. automatically fill title, author, date and contact information:

```
Title:  <TITLE>
Author: <AUTHOR_NAME>
Source:
        Adapted from:
        <TITLE>
Date:   <DATE>
Contact:
        <AUTHOR_NAME>
        <AUTHOR_EMAIL>
```

Where slot `<TITLE>` gets substituted for the original fable’s title, `<AUTHOR_NAME>` is the name of the producer, `<DATE>` the current date and `<AUTHOR_EMAIL>` the e-mail of the producer.

Slot	**Description**
`<TITLE>`	The original title of the story (automatically entered)
`<AUTHOR_NAME>`	The author's name (manually entered)
`<AUTHOR_EMAIL>`	The author's email (manually entered)
`<DATE>`	The current (automatically entered)

Table 3-B: Slots and descriptions where they are extracted from.

C.1.2 Scenes

In FOUNTAIN, scenes are marked with an initial `INT.` or `EXT.` marking whether the scene is interior or exterior. Since there are no camera shots in radio drama however using those markers is rare (although not impossible to find). It is most often the case that a scene is denoted by a starting `SCENE` literal, followed by the number of the scene, and a description of the scene. In the form of extended context-free grammar rules, this becomes:

```
SceneTitle → SceneType sp* ``.''? sp* IoE sp* ac
SceneType → ``SCENE''i sp* int
SceneType → ``PROLOGUE''i
SceneType → ``INTRO''i ``DUCTION''i?
SceneType → ``EPILOGUE''i
SceneType → ``OUTRO''i
IoE → (``INT''i | ``EXT''i) sp* [.:]
Content → (Dialogue|Paren|Fx|Music|Transition) nl+
Scene → SceneTitle nl+ Content+
```

Note that the terminals are case-insensitive which we denote with the suffix 'i'. We will discuss about the non-terminals in *Content* in subsequent sections. *SceneTitle* can be either `PROLOGUE`, `INTRO(DUCTION)`, `EPILOGUE`, or `OUTRO` or the `SCENE` literal followed by an integer number denoting the number of scene. We provide an example of a scene title able to be parsed using the above rules:

```
SCENE 1. EXT. A FOX MEETS A CROW IN A FOREST.
 ...
```

When parsing, we can extract the location using the method described in Section 3.4,

Slot	**Description**
`<INT. or EXT.>`	Whether the scene is internal or external (manually entered)
`<DESCRIPTION>`	The description of the scene (manually entered)
`<LOCATION>`	The location of the scene (automatically entered)

Table 3-C: Slots and descriptions where they are extracted from.

which will allow us to retrieve appropriate sound effects in production. When the rules are used generatively, the name of the scene should optimally become the theme of the story conveyed as drama at that point. Since, however, we do not perform theme extraction from text, when generating we use the description `<DESCRIPTION>` as a scene title, followed by the location of our main character (in our case, the fox). Before adding the text of the location, we change the articles 'the' for 'a' or 'an' based on the first letter of that follows. The scene integer starts counting from 1 and increases with each new scene. The template for scene construction is:

```
SCENE 1. <INT. or EXT.> <DESCRIPTION> -- <LOCATION>.
...
```

The string of the location is used to drive retrieval of the assets discussed in Chapter 5.

C.1.3 Cast List

After title and crediting information, we include an optional cast list segment. In screenplay, or BBC radio drama scripts, characters are introduced using a short introductory sentence or directly from the dialogue lines. While we still allow this feature, we follow the advice of Keith Crawford[3] to include a Cast List to be able to observe the characters at a glance:

```
CAST LIST
The Crow: animal, old, female.
The Fox: animal, young, male.
...
```

[3] https://www.aboutwriting.org/how-to-format-a-radio-play-script-the-cheap-and-easy-way/

Slot	**Description**
`<CHARACTERN>`	A character name (automatically extracted)
`<CHARNTAGM>`	The `M`-th tag of the `N`-th character (automatically extracted)

Table 3-D: Slots and descriptions where they are extracted from.

A cast list is also observed in many of the radio drama scripts found in the Simply Scripts archive[4]. This format also allows for easy parsing by the computer. The grammar rules for the cast list segment is:

```
Castlist → (‘‘CAST LIST’’i|‘‘CHARACTER’’i ‘‘S’’?) nl CInfo
CInfo → (CName (sp∗ ‘‘:’’ sp∗ Tags)? ‘‘.’’ nl)+
CName → name
Tags → Tag sp∗ (‘‘,’’ Tag)∗
```

When created automatically, for each character c extracted from the story and its corresponding tags $\mathcal{T}_c$ add a new $CInfo$ line where $CName = c$ and each $t_c \in \mathcal{T}_c$ is added as a Tag according to the above rules. For example in the fable "The Crow and the Fox" we recognise one character (*The Fox*, recall that we consider only characters who speak a line) which has the tags *animal* and *male*, therefore the Cast List section becomes:

```
CAST LIST
THE FOX: animal, male.
```

The template for generating the cast list of a radio drama is:

```
CAST LIST
<CHARACTER1>: <CHAR1TAG1>, <CHAR1TAG2>.
<CHARACTER2>: <CHAR2TAG1>, <CHAR2TAG2>.
...
```

And the slots are substituted as in Table 3-C. It is up to the author of the radio drama, to add more characters using the same format.

[4] https://www.simplyscripts.com/radio_all.html

C.1.4 Dialogue

Every character line is formatted as the name of the character, followed by a series of *dialogue lines*. Each line might be a *parenthetical* or one or more text sentences. Parentheticals are words or phrases between ‘‘(’’ and ‘‘)’’ and are used to affect the speech of the character. Emotions extracted from the original story which are related to character speech (Section 3.5) are given in the radio drama script as parentheticals. Dialogue might also include *pauses* included in the source text denoted as ellipses (...) or long dashes (–). As an example, the story text:

“You – You disgust me!”, said Mary with contempt.

will be given in the radio drama script as:

```
MARY
(with contempt) You -- You disgust me!
```

Parentheticals can precede dialogue lines like above. Alternatively, they appear in a separate line:

```
MARY
(with contempt)
You disgust me!
```

Parenthetical do not just convey emotional affect, but can also be used to give directions to the actors:

```
MARY (OFF)
You disgust me!
```

Where the `OFF` instructs the actor to move away from the microphone. Further directives can be seen at the BBC guidelines for radio drama writing [208]. The CFG rules for parsing and subsequently processing script dialogue to guide production are:

```
Dialogue → CName nl (DLine nl)+ nl
```

Slot	**Description**
<CHARACTER>	A character name (automatically extracted)
<MODIFIER>	The emotional tag of the character line (automatically extracted)
<CHARACTER_LINE>	The character line (automatically extracted)

Table 3-E: Slots and descriptions where they are extracted from.

```
DLine → Parenthetical | (Parenthetical? nb)
Parenthetical → ‘‘(’’ nb ‘‘)’’
```

The template for dialogue is the following:

```
<CHARACTER>
(<MODIFIER>)
<CHARACTER_LINE>
```

And the slots can be substituted as in Table 3-E.

C.1.5 Actions, Sound Effects, and Music

Sound effects in the script start with the `FX:` literal followed by a sound effect description using all caps. The `MUSIC:` literal is the same but introduces music. The CFG rules for sound effects in the script are:

```
Parenthetical → ‘‘(’’ nb ‘‘)’’
Fx → ‘‘FX:’’i sp∗ (ac nl)+
Music → ‘‘MUSIC:’’i sp∗ (ac nl)+
```

When generating the script, the locations extracted from the source story are used as soundscapes. For example the first sentence of the “Wily Lion”:

> A fat bull was feeding in a meadow when a lion approached him.

Can be tagged as a soundscape as follows:

```
FX: SOUNDSCAPE: A MEADOW
```

The `‘‘MUSIC:’’` tag can be used with time modifiers:

```
MUSIC: FOLK MUSIC PLAYS FOR ABOUT 5 SECONDS
```

The grammar also supports actions in parentheses. There are cases when actions might be referred to in the script that are there to set the setting but are not communicated with sound. Those are distinguished from parentheticals in dialogue (Section C.1.4) in that they are separate from the dialogue segment (at least two new lines separate them from a dialogue segment). In some cases, however, they represent short sound effects or pauses, e.g.:

```
SCENE 1. EXT. THE LION
FX: SOUNDSCAPE: A MEADOW


(THE LION is approaching THE BULL)


THE LION
I cannot help saying how much I admire
your magnificent figure!


(short pause)


THE LION
What a fine head...
```

The names of characters in actions are capitalised. This is a common convention in writing scripts and helps increase the readability of the script. The template is the action enclosed in parentheses and a *nl* character above and below:

```
(<action>)
```

C.1.6 Suspense

In Section 2.2.1 we referred to *suspense* as the emotional strain caused to the listener by impactful events to the life or well-being of characters. We discussed how to extract such events in Section 3.8 and here we discuss how we communicate such events in the radio drama script. As we already discussed in Section 2.2.1, suspense is communicated to the listener through the use of timing and music. We introduce the latter in the radio drama script as suspenseful music to the scene leading to the events that elicit suspense.

```
SCENE X. EXT. ...
MUSIC: SUSPENSEFUL
```

In addition, we introduce a long pause before the suspenseful event.

```
(long pause)
(a character dies)
```

C.1.7 Comments

Comments are an important part of writing radio drama script. They play a pivotal role in communication between the various factors of production [1]. In our case, they are denoted with parentheticals, in the same way as actions that begin with the prefix `NB:` (Nota Bene) optionally annotated with a take number. An example resembling the ‘coloured pens’ approach in [1, Fig. 4] is given below:

```
(NB: Take 1: I cannot... how: This line will be ignored by the parser.)
```

Since everything after `NB:` is ignored, it is up to the production team to decide on their own comment formats.

C.1.8 Transitions

In Section 2.7, we discussed about how the fading effect is used to signify scene transitions. Our grammar supports scene transitions similar to Fountain by using lines that end in `TO:` e.g[5]:

```
NARRATOR
...The Emperor and all his Court came to see the spectacle,
and ANDROCLES was led out into the middle of the arena.


FADE TO:


SCENE 2. EXT. THE ARENA.
FX: SOUDSCAPE: ROMAN CIRCUS
```

Here, `FADE TO:` signifies a soft transition between the previous scene and the arena. Possible transitions are:

- `FADE TO:` – Soft transition between scenes, see above.
- `CROSSFADE TO:` – Crossfade between scenes.
- `SEGUE TO:` or `HARD CUT TO:` – A hard transition between scenes.

The grammar rules for a cut are:

```
Transition → Ttype sp+ ”TO:”i
Ttype → ‘‘FADE’’i ((sp∗|‘‘−’’) (‘‘IN’’|‘‘UP’’))?
Ttype → ‘‘CROSSFADE’’i
Ttype → ‘‘HARD’’i? ‘‘CUT’’i
Ttype → ‘‘SEGUE’’i
```

Since we do not have a method yet to detect events and scene changes accurately, we only use `‘‘FADE TO:’’` when fading from the introductory music to the first (and currently only) scene in the drama:

```
MUSIC: ...


NARRATOR
```

[5] Aesop's Fable: "Androcles and the Lion"

```
You are about to head Aesop's fable "The Wily Lion".
Adapted to radio by ...


FADE TO:


SCENE 1...
```

The template is simply:

```
<TYPE> TO:
```

surrounded by a required *nl* character before and after that line. The `<TYPE>` slot can be substituted with one of `HARD CUT`, `CUT`, `FADE`, `CROSSFADE`, or `SEGUE`.

C.1.9 Example: "The Crow and the Fox"

To showcase how information extracted in Chapter 3 with the CFG introduced above we will show below as an example how this method is applied to the following fable segment[6]:

> A crow was sitting on a branch of a tree with a piece of cheese in her beak when a Fox observed her and set his wits to work to discover some way of getting the cheese. The fox, coming and standing under the tree, looked up and said, "What a noble bird I see above me! Her beauty is without equal, the hue of her plumage exquisite. If only her voice is as sweet as her looks are fair, she must be–without doubt–the queen of the birds. Won't you sing a song for me, O Queen of the Birds?"

Applying the methods of Chapter 3 we extract the following information:

- **Title** – The Crow and the Fox
- **Location** – The tree

[6] Aesop's fable: *"The fox and the crow"*

- **Character** – The fox (animal, male)
- **Emotions (Overall)** – Joy, Trust
- **Emotions (Characters)** – none

The first step is to add the script metadata. Provided that the system already knows the author's name and email address, as well as the current date, the metadata becomes:

```
TITLE: The Crow and the Fox
AUTHOR: Emmanouil Theofanis Chourdakis
SOURCE: Adapted from "The Crow and the Fox"
DATE: 09/13/2019
CONTACT:
Emmanouil Theofanis Chourdakis
e.t.chourdakis@qmul.ac.uk
```

The next step is to introduce the characters. We have one character (the fox) which has the tags *animal* and *male*.

```
CHARACTERS
THE FOX: animal, male.
```

Next, we introduce the story to the listeners and play a short segment of intro music tagged using the overall extracted emotions of *joy* and *trust*. A short initial line by the Narrator is commonly used in radio drama to acclimate the listener to the setting [49, Ch. 6].

```
PROLOGUE INT: INTRODUCTION TO THE STORY


NARRATOR
The crow and the Fox, a story by Emmanouil Theofanis Chourdakis
```

```
MUSIC: PLAY MUSIC THAT ELICITS JOY AND TRUST FOR 5 SECONDS


FADE TO:
```

The parts that follow relate to the main story. Initially, event simplification can be performed using the algorithm in Section 3.7. We keep the parts that do not belong to any character as narrator speech:

```
SCENE 1. EXT. A TREE


NARRATOR
A crow was sitting on a branch of a tree with a
piece of cheese in her beak. A fox observed her.
A fox set his wits to discover some way of getting
the cheese. The fox coming under the tree. The fox,
standing under the tree. The fox looked up, and said:
```

We observe that the narrator speech in this case is inadequate for radio drama. The speech however can act as a 'template' for the author to correct. Additionally, the author can correct the scene headings and add some ATMOS effects:

```
SCENE 1. EXT. THE PIECE OF CHEESE


FX: SOUNDSCAPE: A QUIET FOREST


NARRATOR
A crow was sitting on a branch of a tree with a
piece of cheese in her beak. A fox observed her
and started thinking of a way to get this piece
of delicious cheese...
```

Extracted character speech is added verbatim:

```
THE FOX

What a noble bird I see...
```

Finally, an outro scene is added after a fade-out.

```
FADE TO:


OUTRO INT. CREDITS


NARRATOR

You were listening to The Crow and the Fox.

A story by Emmanouil Theofanis Chourdakis


MUSIC: PLAY MUSIC THAT ELICITS JOY AND TRUST FOR 5 SECONDS
```

This generated script can also be parsed back using the formal grammar. A segment of the parse tree can be seen in Figure 3.1.

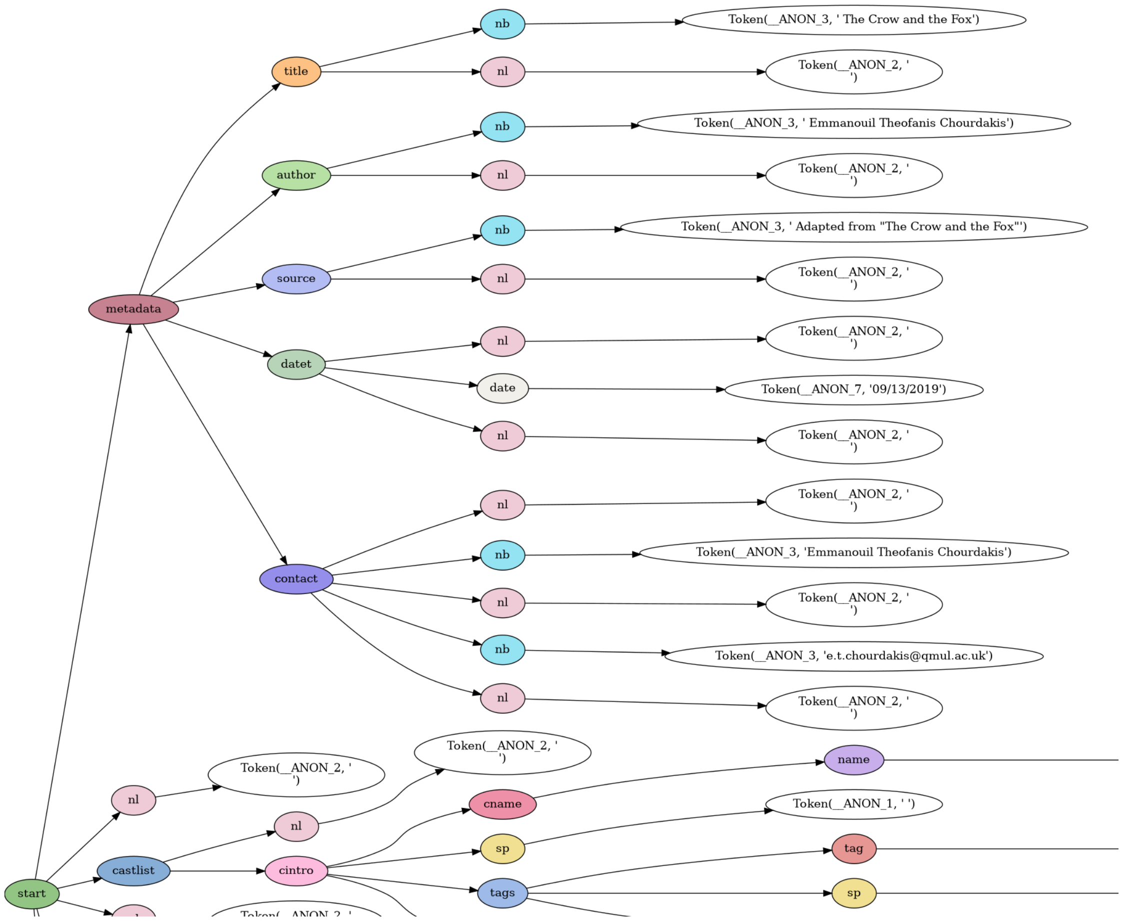


Figure 3.1: A segment of the parse tree.

Bibliography

[1] C. Baume, M. D. Plumbley, and J. Calic, "Use of audio editors in radio production," in *138th Audio Engineering Society Convention*, May 2015.

[2] T. Akimoto, "Computational modeling of narrative structure: A hierarchical graph model for multidimensional narrative structure," *International Journal of Computational Linguistics Research*, vol. 8, no. 3, p. 92–108, April 2017.

[3] R. J. Hand and M. Traynor, *The radio drama handbook: Audio drama in context and practice.* A&C Black, 2011.

[4] J. Nivre *et al.*, "Universal dependencies v1: A multilingual treebank collection," in *10th International Conference on Language Resources and Evaluation*, May 2016.

[5] D. T. Murphy and S. Shelley, "OpenAIR: An interactive auralization web resource and database," in *129th Audio Engineering Society Convention*, November 2010.

[6] E. Nichols and F. Botros, "SpRL-CWW: Spatial relation classification with independent multi-class models," in *9th International Workshop on Semantic Evaluation*, June 2015.

[7] O. Kolomiyets *et al.*, "Semeval-2013 task 3: Spatial role labeling," in *2nd Joint Conference on Lexical and Computational Semantics*, June 2013.

[8] E. Bastianelli *et al.*, "UNITOR-HMM-TK: Structured kernel-based learning for

spatial role labeling," in *2nd Joint Conference on Lexical and Computational Semantics*, June 2013.

[9] A. Mazalov, B. Martins, and D. Matos, "Spatial role labeling with convolutional neural networks," in *9th Workshop on Geographic Information Retrieval*, November 2015.

[10] N. Ramrakhiyani, G. Palshikar, and V. Varma, "A simple neural approach to spatial role labelling," in *41st European Conference on Information Retrieval*, April 2019.

[11] SuperHOAX, "EchoThief impulse response library," http://www.echothief.com/, 2013, accessed: 2018-10-30.

[12] K. Weilhammer and S. Rabold, "Durational aspects in turn taking," in *15th International Congress of Phonetic Sciences*, August 2003.

[13] J. Woodcock *et al.*, "Presenting the S3A object-based audio drama dataset," in *140th Audio Engineering Society Convention*, June 2016.

[14] E. Barnouw, "Handbook of radio writing," 1947.

[15] D. Bogdanov *et al.*, "The MTG-Jamendo dataset for automatic music tagging," in *Machine Learning for Music Discovery Workshop in the 36th International Conference on Machine Learning*, June 2019.

[16] M. Hollander, D. A. Wolfe, and E. Chicken, *Nonparametric statistical methods.* John Wiley & Sons, 2013, vol. 751.

[17] Y. Hochberg, "A sharper bonferroni procedure for multiple tests of significance," *Biometrika*, vol. 75, no. 4, p. 800–802, 1988.

[18] T. Crook, *Radio drama: theory and practice.* Psychology Press, 1999.

[19] H. Blue, *Words at war: World War II era radio drama and the postwar broadcasting industry blacklist.* Scarecrow Press, 2002.

[20] X. Wang and C.-C. J. Kuo, “An 800 bps VQ-based LPC voice coder,” *Journal of the Acoustical Society of America*, vol. 103, no. 5, p. 2778–2778, May 1998.

[21] H. Chignell, *Key Concepts in Radio Studies.* SAGE, 2009.

[22] A. Ingram, *Wireless wisdom: The planner’s guide to going beyond the obvious in radio.* Radio Advertising Bureau, 1994.

[23] E. Huwiler, “Radio drama adaptations: An approach towards an analytical methodology,” *Journal of Adaptation in Film & Performance*, vol. 3, no. 2, p. 129–140, September 2010.

[24] M. Hilmes and J. Loviglio, *Radio reader: Essays in the cultural history of radio.* Psychology Press, 2002.

[25] J. Loviglio and M. Hilmes, *Radio’s New Wave: Global Sound in the Digital Era.* Routledge, 2013.

[26] S. Klein *et al.*, “Automatic novel writing: A status report,” University of Wisconsin-Madison, Tech. Rep., July 1973.

[27] D. Goldberg *et al.*, “Using collaborative filtering to weave an information tapestry,” *Communications of the ACM*, vol. 35, no. 12, p. 61–71, 1992.

[28] M. Schedl *et al.*, “Music recommender systems,” in *Recommender systems handbook.* Springer, 2015, p. 453–492.

[29] D. Dugan, “Automatic microphone mixing,” *Journal of the Audio Engineering Society*, vol. 23, no. 6, p. 442–449, August 1975.

[30] B. De Man *et al.*, “Ten years of automatic mixing,” in *3rd Workshop on Intelligent Music Production*, September 2017.

[31] D. McWhinnie, *The art of radio.* Faber & Faber, 1959.

[32] M. Lopez and S. Pauletto, “The design of an audio film for the visually impaired,” in *15th International Conference on Auditory Display*, May 2009.

[33] ——, “The sound machine: a study in storytelling through sound design.” in *5th Audio Mostly Conference*, July 2010.

[34] S. B. Chatman, *Story and discourse: Narrative structure in fiction and film.* Cornell University Press, 1980.

[35] *scene, n.* Oxford University Press. [Online]. Available: https://www.oed.com

[36] G. Freytag, *Freytag’s Technique of the drama: An exposition of dramatic composition and art.* Elias J. Macwean, 1900.

[37] R. J. Gerrig and A. B. Bernardo, “Readers as problem-solvers in the experience of suspense,” *Poetics*, vol. 22, no. 6, p. 459–472, 1994.

[38] B. O’Neill, “A computational model of suspense for the augmentation of intelligent story generation,” Ph.D. dissertation, Georgia Institute of Technology, 2013.

[39] N. E. Fuchs and R. Schwitter, “Specifying logic programs in controlled natural language,” in *Workshop on Computational Logic for Natural Language Processing*, April 1995.

[40] D. K. Elson, “Detecting story analogies from annotations of time, action and agency,” in *3rd Workshop on Computational Models of Narrative*, May 2012.

[41] B. Li *et al.*, “Story generation with crowdsourced plot graphs,” in *27th AAAI Conference on Artificial Intelligence*, July 2013.

[42] H. Kamp, *A theory of truth and semantic representation.* Blackwell Publishers, 2002.

[43] L. Banarescu *et al.*, “Abstract meaning representation for sembanking,” in *7th Linguistic Annotation Workshop and Interoperability with Discourse*, August 2013.

[44] C. F. Baker *et al.*, “The berkeley framenet project,” in *17th international conference on Computational linguistics*, August 1998.

[45] K. K. Schuler, "Verbnet: A broad-coverage, comprehensive verb lexicon," 2005.

[46] P. Kingsbury and M. Palmer, "From TreeBank to PropBank," in *3rd International Conference on Language Resources and Evaluation*, May 2002.

[47] B. Martin, *Semiotics and storytelling.* Philomel Productions, 1997.

[48] V. Propp, *Morphology of the Folktale*, 2nd ed. University of Texas Press, 2010.

[49] D. F. Esta, "Writing and producing radio dramas: communication for behavior change," 2005.

[50] A. Crisell, *Understanding radio.* Routledge, 2006.

[51] A. Beck, *Radio Acting.* Methuen Publishing, 1997.

[52] M. Chion, *The voice in cinema.* Columbia University Press, 1999.

[53] J. D. Reiss and A. McPherson, *Audio effects: theory, implementation and application.* CRC Press, 2014.

[54] S. S. Reich, "Significance of pauses for speech perception," *Journal of Psycholinguistic Research*, vol. 9, no. 4, p. 379–389, 1980.

[55] S. Gustafson-Capkova and B. Megyesi, "A comparative study of pauses in dialogues and read speech," in *7th European Conference on Speech Communication and Technology*, 2001.

[56] F. Goldman-Eisler, "The distribution of pause durations in speech," *Language and Speech*, vol. 4, no. 4, p. 232–237, 1961.

[57] A. Clark *et al.*, *The handbook of computational linguistics and natural language processing.* John Wiley & Sons, 2013.

[58] L. Del Corro and R. Gemulla, "Clausie: clause-based open information extraction," in *22nd International Conference on World Wide Web*, May 2013.

[59] E. Chourdakis and J. Reiss, “From my pen to your ears: automatic production of radio plays from unstructured story text,” in *15th Sound and Music Computing Conference*, July 2018.

[60] H. Liu and P. Singh, “Conceptnet—a practical commonsense reasoning tool-kit,” *BT technology journal*, vol. 22, no. 4, p. 211–226, 2004.

[61] R. Speer, J. Chin, and C. Havasi, “Conceptnet 5.5: An open multilingual graph of general knowledge,” in *31st AAAI Conference on Artificial Intelligence*, February 2017.

[62] G. A. Miller, *WordNet: An electronic lexical database.* MIT press, 1998.

[63] J. Valls-Vargas, S. Ontanón, and J. Zhu, “Toward character role assignment for natural language stories,” in *9th Artificial Intelligence and Interactive Digital Entertainment Conference*, October 2013.

[64] D. Suciu and A. Groza, “Interleaving ontology-based reasoning and natural language processing for character identification in folktales,” in *10th International Conference on Intelligent Computer Communication and Processing*, September 2014.

[65] A. Groza and L. Corde, “Information retrieval in folktales using natural language processing,” in *11th International Conference on Intelligent Computer Communication and Processing*, September 2015.

[66] K. van Dalen-Oskam *et al.*, “Named entity recognition and resolution for literary studies,” *Computational Linguistics in the Netherlands Journal*, vol. 4, p. 121–136, 2014.

[67] M. Droog-Hayes, G. Wiggins, and M. Purver, “Automatic detection of narrative structure for high-level story representation,” in *The 5th AISB Computational Creativity Symposium*, April 2018.

[68] L. Jahan, G. Chauhan, and M. A. Finlayson, “A new approach to animacy detection,” in *27th International Conference on Computational Linguistics*, August 2018.

[69] L. Jahan and M. Finlayson, “Character identification refined: A proposal,” in *First Workshop on Narrative Understanding*, June 2019.

[70] M. Honnibal and I. Montani, “spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing,” 2017, to appear.

[71] M. Marcus, B. Santorini, and M. A. Marcinkiewicz, “Building a large annotated corpus of english: The Penn Treebank,” *Computational Linguistics*, vol. 19, no. 2, p. 313–330, 1993.

[72] M. A. Finlayson, “Propplearner: Deeply annotating a corpus of russian folktales to enable the machine learning of a russian formalist theory,” *Digital Scholarship in the Humanities*, vol. 32, no. 2, p. 284–300, 2015.

[73] M. Droog-Hayes, G. A. Wiggins, and M. Purver, “Detecting summary-worthy sentences: the effect of discourse features,” in *13th International Conference on Semantic Computing*, January 2019.

[74] K. Clark and C. D. Manning, “Deep reinforcement learning for mention-ranking coreference models,” in *International Conference on Empirical Methods in Natural Language Processing*, November 2016.

[75] L. Vial, B. Lecouteux, and D. Schwab, “Sense vocabulary compression through the semantic knowledge of wordnet for neural word sense disambiguation,” in *10th Global WordNet Conference*, July 2019.

[76] B. Pouliquen, R. Steinberger, and C. Best, “Automatic detection of quotations in multilingual news,” in *Recent Advances in Natural Language Processing*, September 2007.

[77] R. Krestel, S. Bergler, and R. Witte, "Minding the source: Automatic tagging of reported speech in newspaper articles," in *6th Language Resources and Evaluation Conference*, May 2008.

[78] K. Glass and S. Bangay, "A naive salience-based method for speaker identification in fiction books," in *18th Annual Symposium of the Pattern Recognition Association of South Africa*, November 2007.

[79] D. K. Elson and K. R. McKeown, "Automatic attribution of quoted speech in literary narrative," in *24th AAAI Conference on Artificial Intelligence*, July 2010.

[80] H. He, D. Barbosa, and G. Kondrak, "Identification of speakers in novels," in *51st Annual Meeting of the Association for Computational Linguistics*, August 2013.

[81] T. O'Keefe *et al.*, "A sequence labelling approach to quote attribution," in *2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, July 2012.

[82] S. Pareti *et al.*, "Automatically detecting and attributing indirect quotations," in *2013 Conference on Empirical Methods in Natural Language Processing*, October 2013.

[83] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the society for industrial and applied mathematics*, vol. 5, no. 1, p. 32–38, 1957.

[84] M. Droog-Hayes, "The effect of poor coreference resolution on document understanding," in *European Summer School in Logic, Language and Information*, July 2017.

[85] P. Kordjamshidi, M.-F. Moens, and M. van Otterlo, "Spatial role labeling: Task definition and annotation scheme," in *7th conference on International Language Resources and Evaluation*, May 2010.

[86] P. Kordjamshidi *et al.*, "Semeval-2012 task 3: Spatial role labeling," in *1st Joint Conference on Lexical and Computational Semantics*, Jun. 2012.

[87] K. Roberts and S. M. Harabagiu, "Utd-sprl: A joint approach to spatial role labeling," in *First Joint Conference on Lexical and Computational Semantics*, 2012.

[88] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in *13th Conference on Computational Natural Language Learning*, Apr. 2009.

[89] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[90] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual string embeddings for sequence labeling," in *27th International Conference on Computational Linguistics*, August 2018.

[91] A. Akbik *et al.*, "FLAIR: An easy-to-use framework for state-of-the-art NLP," in *2019 Conference of the North American Chapter of the Association for Computational Linguistics*, Jun. 2019.

[92] J. Pennington *et al.*, "Glove: Global vectors for word representation," in *2014 Conference on Empirical methods in Natural Language Processing*, October 2014.

[93] M. Grubinger *et al.*, "The IAPR TC-12 benchmark: A new evaluation resource for visual information systems," in *International workshop ontoImage*, February 2006.

[94] E. F. Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *7th Conference on Natural Language Learning at HLT-NAACL*, May 2003.

[95] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *30th International Conference on Machine Learning*. Jmlr, June 2013.

[96] W. G. Parrott, *Emotions in social psychology: Essential readings.* Psychology Press, 2001.

[97] A. Balahur, J. M. Hermida, and A. Montoyo, "Detecting implicit expressions of emotion in text: A comparative analysis," *Decision Support Systems*, vol. 53, no. 4, p. 742–753, 2012.

[98] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams engineering journal*, vol. 5, no. 4, p. 1093–1113, 2014.

[99] A. Balahur *et al.*, "Emotinet: A knowledge base for emotion detection in text built on the appraisal theories," in *16th International Conference on Application of Natural Language to Information Systems*, June 2011.

[100] S. M. Mohammad, "From once upon a time to happily ever after: Tracking emotions in mail and books," *Decision Support Systems*, vol. 53, no. 4, p. 730–741, 2012.

[101] A. Neviarouskaya, H. Prendinger, and M. Ishizuka, "Recognition of affect conveyed by text messaging in online communication," in *2nd International Conference on Online Communities and Social Computing*, July 2007.

[102] C.-Y. Lu *et al.*, "Automatic event-level textual emotion sensing using mutual action histogram between entities," *Expert systems with applications*, vol. 37, no. 2, p. 1643–1653, 2010.

[103] C. O. Alm, D. Roth, and R. Sproat, "Emotions from text: machine learning for text-based emotion prediction," in *Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, October 2005.

[104] F. Sugimoto and M. Yoneyama, "A method for classifying emotion of text based on emotional dictionaries for emotional reading." in *Artificial Intelligence and Applications*, February 2006.

[105] V. Francisco and R. Hervás, "Emotag: Automated mark up of affective information in texts," in *Doctoral Consortium at the 8th EUROLAN summer school*, Jul. 2007.

[106] A. Neviarouskaya, H. Prendinger, and M. Ishizuka, “Recognition of affect, judgment, and appreciation in text,” in *23rd international conference on computational linguistics*, August 2010.

[107] V. Francisco *et al.*, “Emotales: creating a corpus of folk tales with emotional annotations,” *Language Resources and Evaluation*, vol. 46, no. 3, p. 341–381, 2012.

[108] M. Ptaszynski *et al.*, “Affect analysis in context of characters in narratives,” *Expert Systems with Applications*, vol. 40, no. 1, p. 168–176, 2013.

[109] M. Shardlow, “A survey of automated text simplification,” *International Journal of Advanced Computer Science and Applications*, vol. 4, no. 1, p. 58–70, 2014.

[110] M. Verhagen *et al.*, “Semeval-2007 task 15: Tempeval temporal relation identification,” in *4th international workshop on semantic evaluations*, June 2007.

[111] ——, “Semeval-2010 task 13: Tempeval-2,” in *5th international workshop on semantic evaluation*, July 2010.

[112] P. Comisky and J. Bryant, “Factors involved in generating suspense,” *Human Communication Research*, vol. 9, no. 1, p. 49–58, 1982.

[113] R. Perez y Perez, “Mexica: A computer model of creativity in writing,” Ph.D. dissertation, 1999.

[114] Y.-G. Cheong and R. M. Young, “A computational model of narrative generation for suspense.” in *21st National Conference on Artificial Intelligence*, July 2006.

[115] B. O’Neill and M. Riedl, “Dramatis: A computational model of suspense,” in *28th AAAI Conference on Artificial Intelligence*, July 2014.

[116] R. Doust and P. Piwek, “A model of suspense for narrative generation,” in *10th International Conference on Natural Language Generation*, September 2017.

[117] R. Doust, “A fundamental element for narrative parsing,” in *6th International Workshop on Computational Creativity, Concept Invention, and General Intelligence.*, December 2017.

[118] J. A. Moorer, “About this reverberation business,” *Computer Music Journal*, vol. 3, no. 2, p. 13–28, 1979.

[119] V. Valimaki *et al.*, “Fifty years of artificial reverberation,” *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, no. 5, p. 1421–1448, 2012.

[120] M. R. Schroeder, “Integrated-impulse method measuring sound decay without using impulses,” *Journal of the Acoustical Soc. of America*, vol. 66, no. 2, p. 497–500, 1979.

[121] S. Müller and P. Massarani, “Transfer-function measurement with sweeps,” *J. Audio Eng. Soc*, vol. 49, no. 6, p. 443–471, 2001.

[122] M. R. Schroeder and B. F. Logan, “‘colorless’ artificial reverberation,” *Journal of the Audio Engineering Society*, vol. 9, no. 3, p. 192–197, 1961.

[123] V. Välimäki *et al.*, “More than 50 years of artificial reverberation,” in *60th International Conference on Dereverberation and Reverberation of Audio, Music, and Speech*, January 2016.

[124] Z. Rafii and B. Pardo, “Learning to control a reverberator using subjective perceptual descriptors.” in *10th International Society for Music Information Retrieval Conference*, October 2009.

[125] ——, “A digital reverberator controlled through measures of the reverberation,” Tech. Rep., October 2009.

[126] E. ISO, “3382-1, 2009,“acoustics—measurement of room acoustic parameters—part 1: Performance spaces,”,” *International Organization for Standardization, Brussels, Belgium*, 2009.

[127] J. S. Abel and P. Huang, "A simple, robust measure of reverberation echo density," in *121st Audio Engineering Society Convention*, October 2006.

[128] P. Seetharaman and B. Pardo, "Crowdsourcing a reverberation descriptor map," in *22nd ACM international conference on Multimedia*, October 2014.

[129] T. Zheng, P. Seetharaman, and B. Pardo, "Socialfx: Studying a crowdsourced folksonomy of audio effects terms," in *24th ACM international conference on Multimedia*, October 2016.

[130] P. Zahorik, "Direct-to-reverberant energy ratio sensitivity," *Journal of the Acoustical Society of America*, vol. 112, no. 5, p. 2110–2117, 2002.

[131] F. Adriaensen, "Acoustical impulse response measurement with aliki," in *4th International Linux Audio Conference*, April 2006.

[132] W. A. Sethares, R. D. Morris, and J. C. Sethares, "Beat tracking of musical performances using low-level audio features," *IEEE Transactions on speech and audio processing*, vol. 13, no. 2, p. 275–285, 2005.

[133] B. McFee *et al.*, "Librosa: Audio and music signal analysis in python," in *14th Python in Science Conference*, July 2015.

[134] M. Thorogood and P. Pasquier, "Computationally created soundscapes with audio metaphor." in *4th International Conference on Computational Creativity*, June 2013.

[135] M. Thorogood, P. Pasquier, and A. Eigenfeldt, "Audio metaphor: Audio information retrieval for soundscape composition," in *9th Sound and Music Computing Conference*, July 2012.

[136] P. Cano *et al.*, "Semi-automatic ambiance generation," October 2004.

[137] M. Casu, M. Koutsomichalis, and A. Valle, "Imaginary soundscapes: the soda project," in *9th Audio Mostly: A Conference on Interaction With Sound*, October 2014.

[138] F. Font, G. Roma, and X. Serra, "Freesound technical demo," in *21st ACM international conference on Multimedia*, October 2013.

[139] J. A. Carroll *et al.*, "Simplifying text for language-impaired readers," in *9th Conference of the European Chapter of the Association for Computational Linguistics*, June 1999.

[140] E. K. Canfield-Dafilou, E. Callery, and C. Jette, "A portable impulse response measurement system," in *15th Sound and Music Computing Conference*, July 2018.

[141] G. Chechik *et al.*, "Large-scale content-based audio retrieval from text queries," in *1st ACM International Conference on Multimedia Information Retrieval*, April 2008.

[142] R. Stables *et al.*, "SAFE: A system for the extraction and retrieval of semantic audio descriptors," in *Extended Abstracts for the late-breaking demo session of the 15th International Society for Music Information Retrieval Conference*, October 2014.

[143] R. B. ITU-R, "1534-1,"method for the subjective assessment of intermediate quality levels of coding systems (mushra)"," *International Telecommunication Union*, 2003.

[144] N. Jillings *et al.*, "Web Audio Evaluation Tool: A browser-based listening test environment," in *12th Sound and Music Computing Conference*, July 2015.

[145] F. Husson *et al.*, "Multivariate exploratory data analysis and data mining," *R package*, 2016.

[146] T. H. Pedersen and N. Zacharov, "The development of a sound wheel for reproduced sound," in *138th Audio Engineering Society Convention*, May 2015.

[147] P. D. Pestana, J. D. Reiss, and A. Barbosa, "User preference on artificial reverberation and delay time parameters," *Journal of the Audio Engineering Society*, vol. 65, no. 1/2, p. 100–107, 2017.

[148] B. De Man *et al.*, "Perceptual evaluation and analysis of reverberation in multitrack music production," *Journal of the Audio Engineering Society*, vol. 65, no. 1/2, p. 108–116, 2017.

[149] P. Seetharaman and B. Pardo, "Audealize: Crowdsourced audio production tools," *Journal of the Audio Engineering Society*, vol. 64, p. 683–695, 2016.

[150] R. Stables *et al.*, "Semantic description of timbral transformations in music production," in *22nd ACM international conference on multimedia*, October 2016.

[151] H. Davis and S. M. Mohammad, "Generating music from literature," *arXiv preprint arXiv:1403.2124*, 2014.

[152] J. Salas, "Generating music from literature using topic extraction and sentiment analysis," *IEEE Potentials*, vol. 37, no. 1, p. 15–18, 2018.

[153] S. Harmon, "Narrative-inspired generation of ambient music." in *8th International Conference on Computational Creativity*, June 2017.

[154] B. B. Klebanov, K. Knight, and D. Marcu, "Text simplification for information-seeking applications," in *2004 International Conference On the Move to Meaningful Internet Systems*, October 2004.

[155] E. T. Chourdakis and J. D. Reiss, "Automatic control of a digital reverberation effect using hybrid models," in *60th Audio Engineering Society Conference on Dereverberation and Reverberation of Audio, Music, and Speech*, January 2016.

[156] ——, "A machine-learning approach to application of intelligent artificial reverberation," *Journal of the Audio Engineering Society*, vol. 1/2, p. 56–65, February 2017.

[157] J. Scott *et al.*, "Automatic multi-track mixing using linear dynamical systems," in *8th Sound Music Computing Conference*, July 2011.

[158] S. Hafezi and J. D. Reiss, "Autonomous multitrack equalization based on masking reduction," *Journal of the Audio Engineering Society*, vol. 63, no. 5, p. 312–323, 2015.

[159] E. Perez Gonzalez and J. D. Reiss, "A real-time semiautonomous audio panning system for music mixing," *EURASIP Journal on Advances in Signal Processing*, 2010.

[160] Z. Ma *et al.*, "Intelligent multitrack dynamic range compression," *Journal of the Audio Engineering Society*, vol. 63, no. 6, 2015.

[161] T. Carpentier, M. Noisternig, and O. Warusfel, "Hybrid reverberation processor with perceptual control," in *17th International Conference on Digital Audio Effects*, September 2014.

[162] J. J. Scott and Y. E. Kim, "Instrument identification informed multi-track mixing." in *14th International Society for Music Information Retrieval Conference*, November 2013.

[163] V. Verfaille, U. Zolzer, and D. Arfib, "Adaptive digital audio effects (A-DAFx): A new class of sound transformations," *IEEE Transactions on audio, speech, and language processing*, vol. 14, no. 5, p. 1817–1831, 2006.

[164] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the cuidado project," *CUIDADO IST Project Report*, vol. 54, no. 0, p. 1–25, 2004.

[165] T. Ganchev, N. Fakotakis, and G. Kokkinakis, "Comparative evaluation of various MFCC implementations on the speaker verification task," in *10th International Society for Music Information Retrieval Conference*, September 2005.

[166] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *2000 IEEE International Conference on Multimedia and Expo.*, vol. 1. IEEE, 2000, p. 452–455.

[167] K. West and S. Cox, "Finding an optimal segmentation for audio genre classification." in *Int. Soc. Mus. Inf. Retr. (ISMIR)*, 2005, p. 680–685.

[168] Y. Lu *et al.*, "Feature selection using principal feature analysis," in *Proceedings of the 15th ACM international conference on Multimedia*, 2007, p. 301–304.

[169] C. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2007.

[170] M. Aly, "Survey on multiclass classification methods," *Neural Networks*, vol. 19, p. 1–9, 2005.

[171] J. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in large margin classifiers*, vol. 10, no. 3, p. 61–74, 1999.

[172] B. De Man *et al.*, "The Open Multitrack Testbed," in *137th Audio Engineering Society Convention*, October 2014.

[173] E. Jones *et al.*, "Scipy: Open source scientific tools for python," 2001.

[174] D. Bogdanov *et al.*, "ESSENTIA: an audio analysis library for music information retrieval," in *14th International Society for Music Information Retrieval Conference*, November 2013.

[175] O. Campbell *et al.*, "ADEPT: A framework for adaptive digital audio effects," in *2nd AES Workshop on Intelligent Music Production*, September 2016.

[176] T. Cox *et al.*, "Extracting room reverberation time from speech using artificial neural networks," *Journal of the Audio Engineering Society*, vol. 49, no. 4, p. 219–230, 2001.

[177] J. F. Santos and T. H. Falk, "Blind room acoustics characterization using recurrent neural networks and modulation spectrum dynamics," in *60th Audio Engineering Society Conference on Dereverberation and Reverberation of Audio, Music, and Speech*, February 2016.

[178] A. Belhomme *et al.*, "Blind estimation of room acoustic parameters using kernel regression," in *60th Audio Engineering Society Conference on Dereverberation and Reverberation of Audio, Music, and Speech*, February 2016.

[179] I. J. Kelly *et al.*, "Robust estimation of reverberation time using polynomial roots," in *60th Audio Engineering Society Conference on Dereverberation and Reverberation of Audio, Music, and Speech*, February 2016.

[180] E. Kahle and J.-P. Jullien, "Some new considerations on the subjective impression of reverberance and its correlation with objective criteria," in *Sabine Centennial Symposium*, June 1994.

[181] A. Tamamori *et al.*, "Speaker-dependent wavenet vocoder," in *Interspeech*, August 2017.

[182] Y. Wang *et al.*, "Tacotron: Towards end-to-end speech synthesis," in *Interspeech*, August 2017.

[183] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2019.

[184] S. O. Arik *et al.*, "Deep voice: Real-time neural text-to-speech," in *34th International Conference on Machine Learning*, August 2017.

[185] A. Gibiansky *et al.*, "Deep voice 2: Multi-speaker neural text-to-speech," in *Advances in Neural Information Processing Systems*, December 2017.

[186] R. Skerry-Ryan *et al.*, "Towards end-to-end prosody transfer for expressive speech synthesis with tacotron," in *37th International Conference on Machine Learning*, July 2018.

[187] R. Clark, K. Richmond, and S. King, "Festival 2 – build your own general purpose unit selection speech synthesiser," in *5th ICSA Workshop on Speech Synthesis*, June 2004.

[188] G. K. Anumanchipalli, K. Prahallad, and A. W. Black, "Festvox: Tools for creation and analyses of large speech corpora," in *Workshop on Very Large Scale Phonetics Research*, January 2011.

[189] R. Sproat *et al.*, "Sable: a standard for TTS markup," in *5th International Conference on Spoken Language Processing*, November 1998.

[190] L. Ten Bosch, N. Oostdijk, and J. P. De Ruiter, "Durational aspects of turn-taking in spontaneous face-to-face and telephone dialogues," in *7th International Conference on Text, Speech and Dialogue*, September 2004.

[191] M. Lopez, "Perceptual evaluation of an audio film for visually impaired audiences," in *138th Audio Engineering Society Convention*, May 2015.

[192] EBU, "Guidelines for production of programmes in accordance with EBU R 128," European Broadcasting Union, Geneva, Tech. Rep., January 2016.

[193] L. Ward, B. Shirley, and W. Davies, "Turning up the background noise; the effects of salient non-speech audio elements on dialogue intelligibility in complex acoustic scenes," in *Reproduced Sound Conference*, November 2016.

[194] L. Ward *et al.*, "The effect of situation-specific non-speech acoustic cues on the intelligibility of speech in noise," in *Interspeech*, Stockholm, Sweden, August 2017.

[195] L. A. Ward, "Accessible broadcast audio personalisation for hard of hearing listeners," in *Adjunct Publications of the 2017 ACM International Conference on Interactive Experiences for TV and Online Video*, June 2017.

[196] E. T. Chourdakis *et al.*, "Modelling experts' decisions on assigning narrative importances of objects in a radio drama mix," in *22nd International Conference on Digital Audio Effects*, September 2019.

[197] E. Rishes *et al.*, "Generating different story tellings from semantic representations of narrative," in *6th International Conference on Interactive Digital Storytelling*, November 2013.

[198] D. Elson, "Dramabank: Annotating agency in narrative discourse." in *8th International Conference on Language Resources and Evaluation*, May 2012.

[199] F. Mairesse and M. A. Walker, "Controlling user perceptions of linguistic style: Trainable generation of personality traits," *Computational Linguistics*, vol. 37, no. 3, p. 455–488, 2011.

[200] A. Chang *et al.*, "Text to 3d scene generation with rich lexical grounding," in *International Joint Conference on Natural Language Processing*, July 2015.

[201] A. Radford *et al.*, "Language models are unsupervised multitask learners," Tech. Rep.

[202] T. B. Brown *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, May 2020.

[203] A. Vaswani *et al.*, "Attention is all you need," in *Thirty-first Conference on Neural Information Processing Systems*, December 2017.

[204] P. Gervás and C. León, "The need for multi-aspectual representation of narratives in modelling their creative process," *5th Workshop on Computational Models of Narrative*, July 2014.

[205] X. Chen *et al.*, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *30th Conference on Neural Information Processing Systems*, December 2016.

[206] L. A. Hiller and L. M. Isaacson, *Composition with an electronic computer.* McGraw-Hill, 1959.

[207] J. Earley, "An efficient context-free parsing algorithm," *Communications of the ACM*, vol. 13, no. 2, p. 94–102, 1970.

[208] M. Carless, *BBC Radio Format: Scene Style*, https://downloads.bbc.co.uk/writersroom/scripts/bbcradioscene.pdf, Feb. 2004.