TU WIEN Informatics

An Evaluation of Emotional Speech Disentanglement Methods

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Software und Information Engineering

eingereicht von

Alexandru Nagy
Matrikelnummer 12123679

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Dipl.-Ing. Dr.rer.nat. Thomas Gärtner
Mitwirkung: Dipl.-Ing. David Penz

Wien, 13. Oktober 2025

Alexandru Nagy	Thomas Gärtner

TU WIEN Informatics

An Evaluation of Emotional Speech Disentanglement Methods

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Software and Information Engineering

by

Alexandru Nagy
Registration Number 12123679

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Dipl.-Ing. Dr.rer.nat. Thomas Gärtner
Assistance: Dipl.-Ing. David Penz

Vienna, October 13, 2025

Alexandru Nagy	Thomas Gärtner

Erklärung zur Verfassung der Arbeit

Alexandru Nagy

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 13. Oktober 2025

Alexandru Nagy

Danksagung

Ich möchte mich bei meinem Betreuer David Penz für seine fortwährende und beständige Unterstützung sowie für die zahlreichen Besprechungen während des vergangenen Jahres bedanken. Ebenso danke ich der Forschungbereich Machine Learning E194-06 für die Bereitstellung der notwendigen Ressourcen, um das vollständige Modellgrid zu trainieren und alle erforderlichen Experimente durchzuführen. Schließlich möchte ich Professor Thomas Gärtner danken, der mein Thema genehmigt und somit die Erstellung dieser Arbeit überhaupt erst ermöglicht hat.

Acknowledgements

I would like to thank my supervisor David Penz for continued and consistent support and frequent meetings over the course of a year. I would also like to thank the Research Unit of Machine Learning E194-06 for providing me the resources necessary to train the full model grid and more and run any necessary experiments. Lastly, I would like to thank Professor Thomas Gärtner for approving my theme and thus making the writing of this thesis possible.

Kurzfassung

Die Fähigkeit, Emotionsinformationen von anderen generativen Faktoren der Sprache zu trennen, ist eine bedeutende Aufgabe, die mehrere nachgelagerte Aufgaben im Zusammenhang mit Emotionsklassifikation oder -entfernung, Spracherzeugung und sprecherinvarianter Sprach-Analyse verbessern kann. Historisch gesehen war es schwierig, diese Trennung objektiv zu messen, die formal als Disentanglement bezeichnet wird. Gründe dafür sind die Abhängigkeit der generativen Faktoren voneinander oder das Fehlen von Informationen über reale Daten. Unsere Arbeit behandelt daher die effektive Messung von Disentanglement-Methoden mit sowohl objektiven als auch subjektiven Metriken für Sprachdaten. Unsere Beiträge umfassen die Definition von Disentanglement im Kontext unseres Themas, das Training eines vielfältigen Modellgrids und die Durchführung von Experimenten damit. Wir trainieren eine state-of-the-art Architektur für Sprach-Disentanglement mit drei verschiedenen Disentanglement-Methoden und zwei Datensätzen, dem Emotional Speech Dataset und dem Crowd-sourced Emotional Multimodal Actors Dataset. Darüber hinaus messen wir die Effektivität des Modells mit verschiedenen Metriken und ziehen Schlussfolgerungen aus den Ergebnissen dieser Experimente.

Abstract

Being able to separate emotion information from other generative factors of speech is a meaningful task that can help improve several downstream tasks related to emotion classification or removal, speech synthesis and speaker invariant speech analysis. Historically, it has been difficult to objectively measure this separation, formally noted as disentanglement. Reasons for this difficulty include the interdependence of generative factors or lack of information about real life data. As such, our work deals with effectively measuring disentanglement methods with both objective and subjective metrics for speech data. Our contributions include defining disentanglement in the context of our theme, training a diverse model grid and running experiments with it. We train a state-of-the-art speech disentanglement architecture on three different disentanglement methods and two datasets, Emotional Speech Dataset and Crowd-sourced Emotional Multimodal Actors Dataset. Furthermore, we measure the effectiveness of the model with different metrics and draw conclusions from the results of these experiments.

Contents

CHAPTER 1

Introduction

Emotion recognition in speech is an area of research with significant applications in several other fields, such as human-computer interaction [38], mental health monitoring [30][21], and social robotics [22][28]. Separating the emotional component from other properties of speech allows for improvement in developing more robust and interpretable speech processing systems. Our aim is to apply a pre-existing neural network architecture, so as to separate the emotional part of speech from other factors such as speaker identity, content, and pitch. This separation not only has the potential to enhance the robustness and accuracy of emotion recognition systems [47] but also opens up possibilities for downstream tasks such as style change [35] and emotion removal [15]. The ability to manipulate and control emotional content independently of delivery variations provides opportunities for developing adaptive and personalized technologies. For instance, style change allows for speech synthesis with controlled emotional expressions, which is significant in areas like virtual assistants [42][29]. Similarly, emotion removal can improve applications where neutrality in tone is important, such as speech separation [50].

One of the main issues in this field of research is the question of how to appropriately measure the effectiveness of this separation [13]. Since there is no formal definition or general process through which any kind of input data can be split into generative factors, it is also difficult to define precise measurements for this goal. Previous work [12] often use subjective means, such as plots, to determine if models are able to meaningfully separate different properties of their data. There have also been efforts made to define, review, categorize, and compile [13] [51] objective metrics that measure this separation into individual factors. Throughout our work, we define this separation (formally known as disentanglement) in the context of our topic, namely emotional speech, provide a theoretical foundation for easier understanding and employ both subjective and objective measurements to compare methods of disentanglement on a pre-established architecture.

Thus, in this work, we will explore methods to disentangle emotion and other generative factors of speech with machine learning and explore different ways of measuring the

effectiveness of disentanglement. In the first chapter 2, we will highlight other publications with adjacent themes, such as different architectures to learn disentangled representations or works that apply disentanglement methods in domains other than speech. In the second chapter 3, we will lay the theoretical foundation necessary to design meaningful experiments. In chapter 4, we will describe the methodology with which our models learn disentanglement in detail. In chapter 5 we will describe the environment in which we will run the experiments and detail which metrics we will employ and the reason why they are significant. We will then report on the results of the experiments in chapter 6. Lastly, we will summarize the findings of our review, discuss the limitations of our contributions and describe potential future work in our conclusion 7.

CHAPTER 2

Related Work

Bouchacourt et al. [3] provide an alternative way of learning disentangled representations using variational autoencoders (VAE). They restrict disentanglement to only two factors: style and content, and group input batches by the latter. This way, they create an architecture that works well with unseen combinations of style and content, as long as the content was learnt. The model is trained on MNIST and MS-Celeb-1M [16], and evaluated both qualitatively and quantitatively using classifier accuracy as the metric of choice. This recent paper [48] explores an alternative disentangled method for speech inputs. The authors use a sequential, factorized distillation approach to disentangle speaker identity, linguistic content and emotional representation into distinct subspaces. They use three objective metrics to measure each disentangled subspace: equal error rate for privacy evaluation, word error rate for linguistic content preservation, and unweighted average recall for emotional preservation.

The authors of this recent work [46] disentangle speech into separate content and emotion features. The model consists of two different encoders and takes raw audio data as input, as opposed to our choice of Mel-spectrograms. The disentangled features are then used to generate realistic 3D talking faces. The effectiveness of the disentanglement method is measured both quantitatively and qualitatively indirectly by measuring the performance of the image generation. This paper [27] proposes a method to convert the emotion information of speech by manipulating spectral and prosodic features rather than disentangling emotion into a separate embedding from the rest of the speech attributes. Spectral features are represented as Mel Cepstral Coefficients while prosodic features are represented by the fundamental frequency.

Similar to the concept of speech reconstruction, recent work [8] proposes a novel and scalable multi-domain approach to the task of image-to-image translation, through which only one generator network is necessary for more than two image domains. StarGAN leverages the power of classic generative adversarial networks (GAN) and uses two different labeled datasets in CelebA [26] and RaFD [24] at once during training. Both

qualitative evaluation, through voting, and quantitative evaluation, through classification error on synthesized images, is conducted. The authors show that their architecture outperforms the task competition in both aspects.

CHAPTER 3

Preliminaries

In order to design meaningful experiments for our research topic, it is important to provide a theoretical basis of the shape and characteristics of our data sets and of our main learning objective, namely disentanglement. As such, we will briefly explain the preprocessing steps necessary to convert audio signals into a neural network compatible format, present different definitions of disentanglement and explain what constitutes a disentangled representation in the context of our work.

3.1 Mel-Spectrograms

The task of translating speech into digital data has been extensively researched for a long period of time and has evolved standardized data formats, such as .mp3 [40] or Compact Disc [33]. One of the most popular solutions is the Waveform Audio File Format (WAVE, .wav) [10], introduced by Microsoft and IBM in 1991 for high-fidelity digital audio storage. For the context of this work, we will refer to the WAVE format as our standard data format. WAVE files provide key properties for the calculation of Mel-Spectrograms, which represent the standard input format for audio-related machine learning tasks [12][52][49][37].

Sample Rate: Number of samples per second in audio. In your case, all data will be sampled at 16 kHz.
Bit Depth: Number of bits used per sample, which affects the quality of the audio signal and therefore the spectrogram accuracy.
Number of Channels: Mono or stereo, with Mel-Spectrograms generally operating on single-channel data.
Duration, **Offset** and **Amplitude** at any point in time.

Figure 3.1 presents the full signal of an arbitrary .wav file in the "Full Audio Signal" plot with amplitude on the y-axis and time on the x-axis.

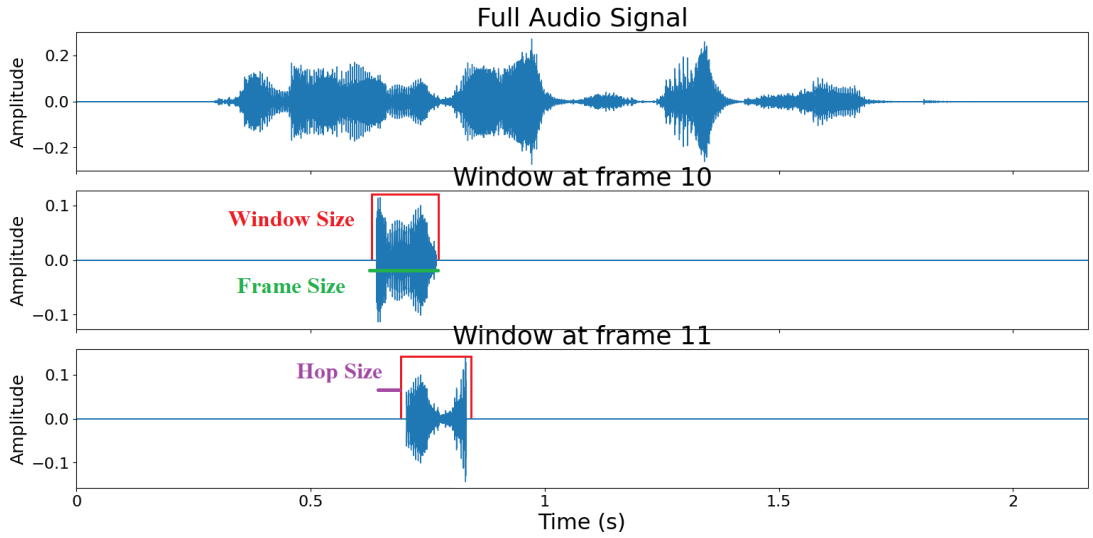

Figure 3.1: .wav audio signal and STFT parameters visualized

By applying the Short-Term Fourier Transformation [2][25], or in short STFT, we can convert a digital audio signal into a Spectrogram [45]:

$$S(m,k) = \sum_{n=0}^{N-1} x(n+mH) \cdot w(n) \cdot e^{-i2\pi n \frac{k}{N}} \tag{3.1}$$

where k is the frequency, m is the time (frame index), N is the number of samples within one time frame and $x(n+mH)$ represents the signal that's only present in the m'th frame with H being the hop size between frames. Therefore mH represents the first sample in the m'th frame. The window size represents the number of samples we apply windowing to, whereas frame size represents the number of samples in each chunk of the segmented signal that gets passed to the STFT. These two are usually identical, however, the frame size can be larger than the window size, which leads to 0-padding being passed to the STFT. The signal is multiplied by the windowing function $w(n)$. All of the parameters are visually explained in Figure 3.1, in the plots titled "Window at frame x", where a square window function is chosen for the display. By multiplying with the term $e^{-i2\pi n \frac{k}{N}}$, the signal is projected onto the $\frac{k}{N}$ frequency component. The final number of frequency bins can then be defined as $\frac{framesize}{2} + 1$ and the final number of frames can be defined as $\frac{samples - framesize}{hopsize} + 1$. The resulting spectrogram can be seen in Figure3.2:

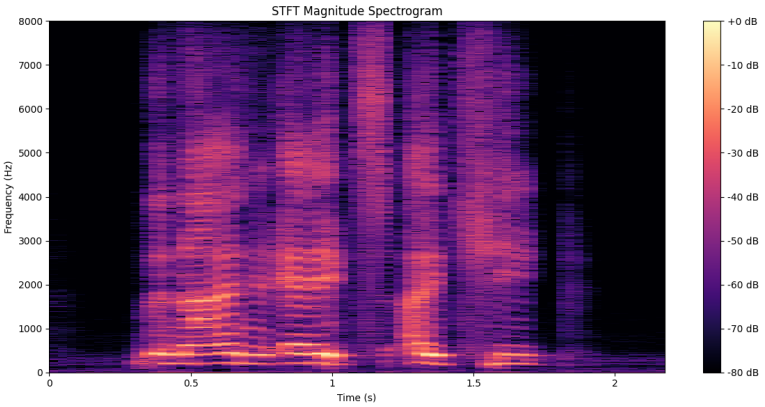

Figure 3.2: Spectrogram created by applying STFT to an arbitrary signal

The last step is related to our ability to perceive sound. Humans perceive sound logarithmically [36], while frequency is expressed linearly in the spectrograms created via the STFT process. Therefore, we want to achieve an audio feature that still keeps the time-frequency representation and the perceptually-relevant amplitude representation intact, while also encoding a perceptually-relevant frequency representation. This paper [41] defines the Mel-scale as a perceptually-relevant scale for pitch. The Mel-scale, visible in figure 3.3[1] is a logarithmic scale with the property that equal distances on the scale have the same perceptual distance, with a baseline of 1000 Hz = 1000 Mel.

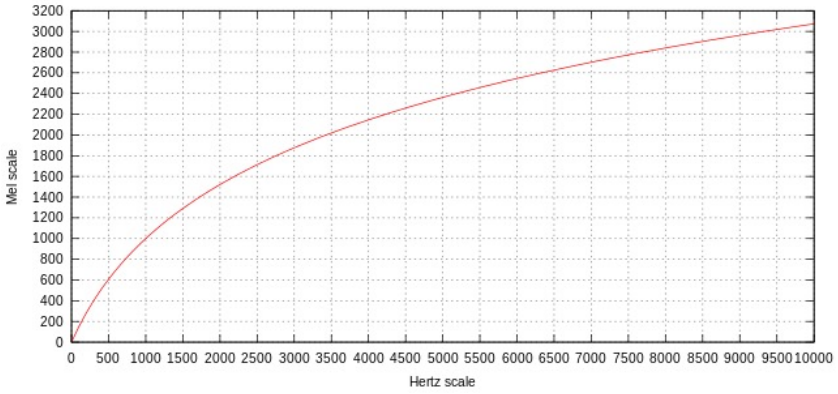

Figure 3.3: Mel-Scale

We can move from frequencies expressed in Hz to frequencies expressed in Mels using the

[1]Figure Source: https://wstyler.ucsd.edu/talks/l113_19_pitchloudnessloc_handout.html

following formula:

$$m = 2595 \times log(1 + \frac{f}{500}) \tag{3.2}$$

where f is the Hz-frequency and m is the Mel-frequency. An inversion is possible with the formula:

$$f = 700(10^{m/2595} - 1) \tag{3.3}$$

For more details about the conversion process, refer to the following papers [20][52]. The resulting Mel-Spectorgram can be seen in Figure 3.4. We can notice that the scale of the frequency axis changes, despite the minimum and maximum values staying the same. The final form of the output is a matrix of the dimensions $[\#Melbands, \#Frames]$. Mel bands are a hyperparameter needed for the conversion process, and we will set it to 80 for all experiments, inspired by StyleVC [12].

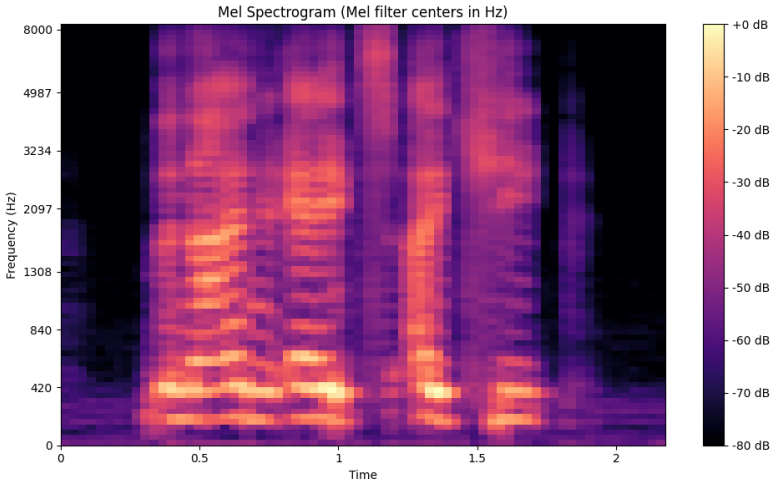

Figure 3.4: Mel-Spectrogram of an arbitrary audio signal

3.2 Disentanglement

Disentanglement and disentangled representations are concepts that have been explored and defined many times in recent years [17][13][51]. However, there is no one standard formal definition, largely due to the fact that real-life data is difficult to effectively split into formal, learnable dimensions. To objectively quantify this property, we will compare past definitions and adapt them for our audio-data based task.

Higgins et. al. [17] attempt to define disentanglement by focusing on *symmetries* [32]: by transforming an object, certain aspects of the object's state change while others remain unchanged. They further define the set of all transformations that would change an object's any arbitrary property or set thereof, but not its identity as a *symmetry group*,

while the consequences of these transformations are defined as *actions*. Take, for example, a data set with images of ducks. One possible transformation is isolating and changing the age of the duck. That way, certain aspects of its physical appearance change, but it's positioning, breed and background do not. The actions that only change one specific property are then defined as *disentangled group actions*. Symmetry groups can be further split into *subgroups*, each subgroup containing the transformations that would affect one property. When thinking of the same example as earlier, if we are able to isolate and manipulate only the size of the duck's beak, that would constitute a symmetry subgroup where the different disentangled group actions would be making it bigger or smaller. Suppose that a generative process exists that can produce *observations* based on a collection of properties (also defined as a *state*), then a dataset of such observations would serve as input for an inference model that could produce *vector representations*. An observation is an example / data point / snapshot drawn from a data set, containing information about one or more factors or properties. For example, an observation would be one specific image of a duck from our data set, and its state would be the collection of properties of this image, such as positioning, background, appearance, lighting, etc. As such, the authors finally reach the following definition:

> "A vector representation is called a disentangled representation with respect to a particular decomposition of a symmetry group into subgroups, if it decomposes into independent subspaces, where each subspace is affected by the action of a single subgroup, and the actions of all other subgroups leave the subspace unaffected."

In other words, a model that can produce a mapping between an abstract state space and real vector representations is desired. For more details on the formal definition of disentanglement, we refer to [17].

Other work [13] defines disentanglement as follows:

> "A disentangled representation is generally described as one which separates the factors of variation, explicitly representing the important attributes of the data".

In order to build an evaluation framework for disentangled representations, the authors describe three different criteria: disentanglement, completeness, and informativeness. These same criteria can also be found in the definition established by the previously discussed work, although disentanglement is called modularity and informativeness is called explicitness. Disentanglement refers to the property of a model where each learnt latent dimension of the representation contains information about a single factor of variation. Completeness measures how many latent dimensions are needed to contain a single factor of generation. For example, if we encode all information about a duck picture into a 10-dimensional latent vector, it can be considered disentangled if the

first dimension only contains information about its breed and nothing else (and the same stands for the other 9 dimensions) and it can be called complete if only the first dimension contains information about its breed (and the same stands for other factors of generation). Informativeness in a representation refers to its ability to completely describe the explanatory factors of interest, allowing for the retrieval of a complete factor realization from a specific point in the latent vector representation space. For a representation to possess perfect explicitness, it must have a generalizable relationship between the factors and their corresponding codes. The simplest and most desirable form of this relationship is linear, providing straightforward and intuitive connections between the factors and the latent codes. Ultimately, achieving explicitness ensures that a representation effectively captures the information content necessary for downstream tasks and enables a clear understanding of the underlying explanatory factors.

In our case, it is difficult to formally define our factors of variation. Throughout our research we will split speech into emotion, speaker identity, content and pitch, with pitch being the only mathematically definable dimension. As such, we will not concern ourselves with completeness, as the other 3 abstract factors of variation cannot be reasonably and effectively encoded into small latent spaces. While a formal representation is difficult, subjective recognition of speech, emotion, and voice is reasonably exploitable. By exploitable, we mean that we are able to work with data that has been accurately labeled by all 3 of these factors, with pitch being mathematically extractable. This will aid us in both the learning process 4 and the evaluation tasks 5.

CHAPTER 4

Method

In this chapter, we will introduce the main methodology used for our contribution to the research question. First, we will present the core architecture used across all experiments, in StyleVC [12]. Having one consistent architecture allows for a stable comparison of disentanglement methods. We will then describe all three disentanglement methods in detail. Finally, we will describe the neural network-based predictors we use to enhance the aforementioned architecture.

4.1 StyleVC

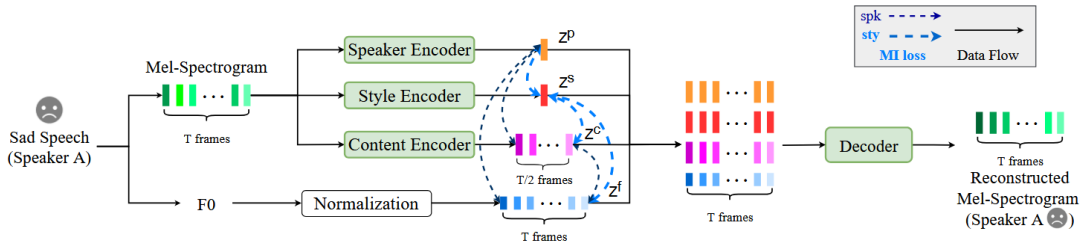

Figure 4.1: StyleVC architecture [12]

Figure 4.1 shows that the architecture consists of three separate encoders that all take audio signals converted to Mel-spectrograms as input and encode them into compressed embeddings. The three embeddings are then brought back to the original length in frames and, along with the normalized fundamental frequency of the signal, passed as input to the model decoder. The decoder will then learn to recreate the input in the original length. Due to the shape of each individual model, which we describe throughout this section, the architecture can accept inputs of any length above a threshold value of 128,

but during the training phase, the input length is standardized for consistency. As such, every input during the training phase will have the shape $[Batchsize(B), 80, T]$, where 80 is the number of Mel banks and $T = 128$ is the number of time frames.

The **fundamental frequency ($F0$)** represents the lowest frequency of a periodic waveform and is perceived as the pitch of the sound. The authors explain that $F0$ is taken as the 4th embedding when decoding, as it varies by speaker. It is extracted from each input speech waveform and log-normalized to zero mean and unit variance.

The **speaker encoder** is based on the AdaIN-VC [6] speaker encoder and is made up of a convolution bank with 8 layers, 6 blocks of 2 convolution layers each, an average adaptive pooling layer, 6 blocks of 2 dense layers each, and a linear layer. The convolution bank passes the input through eight 1D convolutions with kernel sizes of 1 to 8, leading to an output of shape $[B, (c_{bank} * \#kernels + c_{in}), T]$ where $c_{bank} = 128, \#kernels = 8, c_{in} = n\_mels = 80$. Before this output is passed on to the next block of convolutions, it first goes through a 1D convolution that reduces the number of channels to $c_h = 128$. Therefore, the convolution blocks receive input of the shape $[B, c_h, T]$. The convolution blocks have the following structure:

Algorithm 4.1: Convolution Blocks

Input : Input tensor *out*, subsample factors $subsample[1, \ldots, N]$
Output : Processed tensor *out*

for $l \leftarrow 1$ **to** N **do**
  $y \leftarrow$ FirstConvLayer[l](out);
  $y \leftarrow$ Activation(y);
  $y \leftarrow$ SecondConvLayer[l](y);
  $y \leftarrow$ Activation(y);
  if $subsample[l] > 1$ **then**
    $out \leftarrow$ AvgPool1D(out, kernel_size=subsample[l]);
  end
  $out \leftarrow y + out$;
end
return *out*

where $N = 6, subsample = [1, 2, 1, 2, 1, 2]$, the kernel size of each convolution layer is 5, the stride is equal to the subsampling factor of each block, and the activation function is the rectified linear unit (ReLU) [31] function.

$$\text{ReLU}(x) = \max(0, x) \tag{4.1}$$

Every second block halves the temporal resolution by applying a stride of 2 to the second convolution layer of the block and using average pooling with a kernel size and stride of 2 on the original input to match, leading to a final output of the shape $[B, c_h, T/8]$. The

convolutions are followed by a global adaptive average pooling layer, which reduces the temporal resolution to 1. The dense blocks have the following structure:

Algorithm 4.2: Dense Layers

Input : Input tensor *out*, number of dense blocks N
Output : Processed tensor *out*

for $l \leftarrow 1$ **to** N **do**
    $y \leftarrow \text{FirstDenseLayer}[l](out)$;
    $y \leftarrow \text{Activation}(y)$;
    $y \leftarrow \text{SecondDenseLayer}[l](y)$;
    $y \leftarrow \text{Activation}(y)$;
    $out \leftarrow y + out$;
end
return *out*

where $N = 6$ and the activation function is the rectified linear unit (ReLU) function. The output of the dense blocks retains the shape $[B, c_h]$. The final output layer is a fully connected linear layer that maps a tensor with hidden channel size c_h to the final speaker embedding z_{spk}. As such, the shape of the final result is $[B, z_{spk}]$ with $z_{spk} = 256$.

The **emotion/style encoder** is made up of 6 blocks of one 2D convolution layer and one batch normalization layer each, followed by the ReLU activation function in each of them. The output is reshaped and passed to a gated recurrent unit (GRU) [7][9] and then finally to two linear, fully connected layers. Each convolution layer has a kernel size of $[3, 3]$, a stride of $[2, 2]$ and padding of $[1, 1]$. Given an input of the shape $[B, 1, T, n\_mels]$, they change the shape as follows: $[B, 1, T, n\_mels] \rightarrow [B, 32, \frac{T}{2}, \frac{n\_mels}{2}] \rightarrow [B, 32, \frac{T}{4}, \frac{n\_mels}{4}] \rightarrow [B, 64, \frac{T}{8}, \frac{n\_mels}{8}] \rightarrow [B, 64, \frac{T}{16}, \frac{n\_mels}{16}] \rightarrow [B, 128, \frac{T}{32}, \frac{n\_mels}{32}] \rightarrow [B, 128, \frac{T}{64}, \frac{n\_mels}{64}]$. Given a number of blocks K, the final output of the convolution layers is $[B, 128, \frac{T}{2^K}, \frac{n\_mels}{2^K}]$. The tensor is reshaped to $[B, \frac{T}{2^K}, 128 * \frac{n\_mels}{2^K}]$ and passed to a GRU layer with input size $128 * \frac{n\_mels}{2^K}$ and hidden size $\frac{E}{2}$ with $E = 256$. The output of this layer has the shape $[1, B, \frac{E}{2}]$. The final linear layers transform this tensor from the shape $[B, 128]$ to $[B, 256]$ and then $[B, 256]$ again, before applying ReLU one last time. This represents the final emotion embedding z_{emo}.

The speaker and emotion encoders do not have dedicated loss function components. The encoded embeddings are instead used in all disentanglement methods we experiment with, which are explained in section 4.2.

The content encoder is based on the VectorQuantizedCPC[1] architecture. It consists of one convolutional layer, 5 linear layers, a vector quantization (VQ) layer, and a long short-term memory (LSTM) [18], recurrent layer. The first layer is a 1D convolution with kernel size 4, stride 2 and padding 1. The number of input channels is equal to n_mels and the number of output channel is equal to $c_h = 512$. This layer halves the time resolution. The block of linear layers functions as follows:

[1]Source code available at:`https://github.com/bshall/VectorQuantizedCPC/tree/master`

Algorithm 4.3: Linear Layers

Input : Input tensor *out*, number of repeated blocks N, channels dimension c_h, output dimension z_{dim}
Output : Encoded tensor *out*

for $i \leftarrow 1$ **to** N **do**
  $out \leftarrow \text{LayerNorm}(out, \text{dim} = c_h)$;
  $out \leftarrow \text{ReLU}(out)$;
  $out \leftarrow \text{Linear}(out, \text{in\_features} = c_h, \text{out\_features} = c_h)$;
end
$out \leftarrow \text{LayerNorm}(out, \text{dim} = c_h)$;
$out \leftarrow \text{ReLU}(out)$;
$out \leftarrow \text{Linear}(out, \text{in\_features} = c_h, \text{out\_features} = z_{dim})$;
return *out*

with $c_h = 512, z_{dim} = 64$ and $N = 4$. Notably, layer normalization is done across the channels. This creates a tensor of the shape $[B, 64, \frac{T}{2}]$. The VQ layer maps the continuous output of the previous layer to a codebook of 512 64-dimensional learnable vectors. This layer results in its own loss component for the speaker encoder: the commitment loss (L_{VQ}), with the commitment cost acting as a weight that encourages the encoder output to commit to embeddings. It measures how well the continuous encoder output matches the nearest embedding vector and is defined as

$$L_{VQ} = ||x - sg(q(x))||^2. \tag{4.2}$$

The formula represents the mean squared error (MSE) between the encoder output x and the quantized output $q(x)$, with $sg()$ meaning that the gradient is not being backpropagated through the quantized vectors. The codebook vectors are updated through a separate method during the forward pass, namely by exponential moving averages (EMA) [19].

$$EMA_t = \alpha \times x_t + (1 - \alpha) \times EMA_{t-1} \tag{4.3}$$

The formula is used to update two parameters:

$$ema\_count = decay \times ema\_count + (1 - decay) \times current\_assignments \tag{4.4}$$

$$ema\_weight = decay \times ema\_weight + (1 - decay) \times sum\_of\_assigned\_inputs \tag{4.5}$$

with $decay = 0.999$. *ema_count* keeps track of how many encoder outputs are assigned to each codebook embedding, while *ema_weight* represents the sum of all encoder outputs assigned to each embedding. *ema_count* is normalized to avoid division by zero:

$$n = \sum_{m=1}^{M} ema\_count[m]; ema\_count = \frac{ema\_count + \epsilon}{n + M \times \epsilon} \times n \tag{4.6}$$

where M is the total number of codebook embeddings, so in our case 512 and $\epsilon = 1e-5$. The embeddings are finally updated as the average of all currently assigned encoder outputs:

$$embedding_m = \frac{ema\_weight_m}{ema\_count_m} \tag{4.7}$$

The output of the VQ layer keeps the form $[B, 64, \frac{T}{2}]$. The LSTM layer outputs a sequence of hidden states of the size $c_{dim} = 256$, which is used to derive a second loss function term, namely the contrastive predictive coding loss term (L_{CPC}) [43]. It is calculated by having the model predict future frames z_{t+k} given past context vectors c_t. For each prediction step k, it compares the predicted frames to the real future frame, as well as several negative samples chosen from the same sequence at different time steps. The model is penalized for failure to correctly identify real future frames. As such, the final shape of the content embedding $z_{content}$ is $[64, \frac{T}{2}]$.

The **decoder** is based on Kaizhi Qian et al.'s decoder architecture [34] and consists of an LSTM layer, three 1D convolution layers, two more LSTM layers, a linear layer and a 5-layer 1D convolution stack for refinement (Postnet). The emotion embeddings and speaker embeddings are repeated until they reach the original sequence length T. The content embedding is upsampled via interpolation to achieve the same length. The $F0$ values are already of sequence length. All four embeddings are concatenated as input for the first LSTM layer, creating a $[B, T, 256+256+64+1]$ tensor. The LSTM layer outputs a tensor of shape $[B, T, 512]$. Every layer of the convolution block has a kernel size of 5, a stride of 1, a padding of 2, dilation of 1, and is followed by a 1D batch normalization layer and ReLU activation, leaving the tensor shape unchanged. The next LSTM layers double the feature channels, creating a tensor of shape $[B, T, 1020]$. The linear layer compresses the number of channels back to the shape of the original input spectrograms: $[B, T, 80]$. The postnet is composed of 5 1D convolutions, each with kernel size 5, stride 1, padding 2, dilation 1, and a layer of 1D batch normalization. The first convolution upscales the 80 input channel to 512, the next 3 leave the dimensions unchanged, and the last one compresses the size back to 80. The first 4 convolutions have the tanh activation function and the last one has linear activation, where:

$$\tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{4.8}$$

The decoder adds its own term to the final loss function, namely the reconstruction loss. It is calculated as the mean squared losses between the target spectrogram and the decoded spectrograms before and after the postnet block. Additionally, the mean absolute error losses (MAE/L1) between the target spectrogram and the decoded spectrograms before and after the postnet block are added to the term. The loss term can be formally

defined as:

$$L_{recon} = \frac{1}{N}\sum_{i=1}^{N}(\hat{M}_i - M_i)^2 + \frac{1}{N}\sum_{i=1}^{N}(\hat{M}_{post,i} - M_i)^2 + \frac{1}{N}\sum_{i=1}^{N}|\hat{M}_i - M_i| + \frac{1}{N}\sum_{i=1}^{N}|\hat{M}_{post,i} - M_i| \tag{4.9}$$

where N is the total number of dimensions within a Mel-spectrogram, M is the target mel, $\hat{M}$ is the predicted mel output before the postnet and $\hat{M}_{post}$ is the predicted me output after the postnet. This results in a combined loss of:

$$L_{VQ} + L_{CPC} + L_{recon} \tag{4.10}$$

The content encoder is self-contained and will not need further enhancements or disentanglement methods to learn content information in its own embeddings. From now on, we will focus on effectively disentangling speaker and emotion information in their respective embeddings.

4.2 Disentanglement-Specific Loss Functions

In this section we will describe every disentanglement method we will experiment with in detail by defining the resulting loss function component(s) and highlighting any additional steps necessary for the calculation of the losses.

4.2.1 Mutual Information Loss

The original StyleVC architecture uses MI-based learning to disentangle emotion and speaker information. To achieve this, the authors use MI-estimator networks based on the CLUB [5] architecture (mutual information contrastive learning upper bound). The estimators are used to approximate six different MI-CLUB values between the content, speaker, emotion and pitch embeddings, which then add up to the following two loss terms:

$$L_{spk-MI} = I(z_{spk}, z_{content}) + I(z_{spk}, z_{pitch}) + I(z_{content}, z_{pitch}) \text{ [12]} \tag{4.11}$$

$$L_{emo-MI} = I(z_{emo}, z_{spk}) + I(z_{emo}, z_{content}) + I(z_{emo}, z_{pitch}) \text{ [12]} \tag{4.12}$$

where L_{spk-MI} is the MI loss related to the speaker, L_{emo-MI} is the MI loss related to emotion, I represents the estimated MI-CLUB and z_{pitch} is the fundamental frequency embedding. These terms are added up to one general MI loss term:

$$L_{MI} = \lambda_{emo}L_{emo-MI} + \lambda_{spk}L_{spk-MI} \text{ [12]} \tag{4.13}$$

where $\lambda_{spk} = 2\lambda_{emo} = 0.02$.

4.2.2 Group Center Loss

Inspired by ML-VAE [3], we define a second disentanglement loss, namely the group center loss (GCL):

Algorithm 4.4: GroupCenterLoss Forward Pass

Data: embeddings $E \in \mathbb{R}^{B \times D}$, group labels $G \in \{0, \ldots, N-1\}^B$
Result: center_loss scalar
Function `GroupCenterLoss(`E, G`):`
  batch_centers $\leftarrow$ centers[G];
  distances $\leftarrow (E - \text{batch\_centers})^2$;
  sum_distances $\leftarrow$ sum(distances, dim $= 1$);
  center_loss $\leftarrow$ mean(sum_distances);
  return center_loss;

where E is z_{spk} and z_{emo} respectively, B is batch size, D is the dimension of each embedding, both being 256 in our case and N depends on the dataset used. The centers of each group are treated as model parameters: Initialized as random vectors and updated with each backwards pass by the optimizer. We get the formula:

$$L_{GC} = \frac{1}{B} \sum_{i=1}^{B} ||e_i - c_{g_i}||_2^2 \tag{4.14}$$

where B is the batch size, e_i is the embedding of the i-th observation, c_{g_i} is the center vector of group g_i and $||.||_2$ is the Euclidean norm.

4.2.3 Adversarial Loss

We train adversarial classifiers that penalize the model if it predicts speakers labels from emotion embeddings and vice versa. The architecture shares similarities with the one explained in 4.3, with the only difference being the inclusion of a gradient reversal layer based on Yaroslav Ganin and Victor Lempitsky's model [14] before the linear layer.

Algorithm 4.5: Gradient Reversal Layer

Input : input tensor x, gradient scaling factor λ
Output : output tensor y
begin
  $y \leftarrow x$; `// Forward pass: identity`
  `// Backward pass: gradients multiplied by` $-\lambda$
  `(implicitly handled)`
  return y;
end

The loss term is calculated with generalized cross entropy (GCE) between emotion embedding logits and speaker labels and vice versa.

$$L_{GCE} = \frac{1}{B}\sum_{i=1}^{B}\frac{1-p_{i,y_i}^{q}}{q} \tag{4.15}$$

where B is the batch size, p is the predicted probability of the true class y and q is a constant. When $q \to 0$, GCE becomes the standard CE-loss and when $q = 1$ the GCE becomes the MAE. p is calculated by applying the $softmax$ function to the input logits. Our GCE calculation is based on Alan Chou's[2] implementation of Zhilu Zhang and Mert R. Sabuncu's paper [53].

4.3 Speaker and Emotion Classifiers

In addition to the disentanglement specific loss functions, we investigate the impact of adding a classification loss term for the emotion and speaker encoders. The added classifiers are intended to provide even further guidance in the learning process in the form of additional supervision.

We use fully connected linear models with no hidden layers for classification. They each take the encoded embeddings as input and return logits in the shape of $[B, n\_emotions]$ and $[B, n\_speakers]$ respectively. The output is used to calculate the cross-entropy losses (L_{CE}) for each of the two classifiers, where:

$$L_{CE} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{C} y_{ij} log(p_{ij}) \tag{4.16}$$

where y is the one-hot encoded true label vector, p is the predicted probability vector, B is the batch size and C is the number of classes (emotions or speakers respectively). p is calculated by applying the $softmax$ function to the classifier output, where

$$\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{4.17}$$

The loss term penalizes confident but wrong predictions.

[2]Source code available at: `https://github.com/AlanChou/Truncated-Loss/blob/master/TruncatedLoss.py`

CHAPTER 5

Experiment Design

In this chapter, we will describe the technical setup of the experiment environment. We will present everything concerning the data we are working with (Section 5.1), the full grid of models we will train (Section 5.2) and the exact training goals we will measure (Section 5.3.1). We will also provide a detailed description of the metrics used to measure our goals, both subjective (Section 5.3.2) and objective (Section 5.3.3).

5.1 Data

In the data section, we will first present the characteristics and structure of two data sets we use to train our models. We then describe the preprocessing steps and Mel-spectrogram conversion parameters.

5.1.1 Datasets

The first data set we use is the Emotional Speech Dataset (ESD) [54]. The data set is made up of 350 utterances, spoken by 10 Mandarian speakers and 10 English speakers. Transcripts are provided for all utterances and every audio file is labeled by content, emotion and speaker ID. We subset only the English utterances, and as such the training data totals 17,500 utterances.

We train a subset of the model grid on a second dataset, namely Crowd-sourced Emotional Multimodal Actors Dataset (CREMA-D) [4]. This data set includes a higher diversity of speakers and emotion labels, with 91 actors of varied backgrouds and 6 different emotion labels as opposed to ESD's 5, further detailed by an emotion intensity label. Both the total number of utterances and the content variety are lower than those of ESD, with the values being 7442 and 12 respectively.

5.1.2 Preprocessing

The ESD data set follows a 70-20-10 split and the CREMA-D data set follows an 80-15-5 split (train-test-validation). The difference in splits is due to CREMA-D having $\tilde{5}0\%$ of the entries that ESD has. As such, we keep more CREMA-D entries for training so that reconstruction quality doesn't suffer. All utterances from both data sets are sampled at 16000 Hz, therefore no further down- or upsampling is needed. Both the frame size and the window size are set to 400 samples with a hop length of 160 samples. We use the Hann window function [39]:

$$w(n) = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right) \tag{5.1}$$

where $n = 0, 1, 2, ..., N-1$ and $N = window\_length, n = sample\_index$.

The frequency range is limited to 80-7600 Hz, which also serve as the minimum and maximum values of our spectrogram frequency axis. The conversion creates Mel spectrograms with 80 Mel banks and the time resolution is limited to 128 frames by sampling a random section of longer signals and repeating shorter signals to match. The fundamental frequency is extracted and aligned to match the length of the mel spectrogram, then converted to $log - F0$ where $F0 \neq 0$. All the Mel-spectrograms from the training set are gathered to compute global mean and standard deviation across frames and Mel banks for normalization.

5.2 Model Grid

This section contains a description of every model we will train to experiment with and the data set used respectively.

Table 5.1 shows the final training grid. All models were trained using the StyleVC architecture 4.1. Every disentanglement method refers to the losses outlined in section 4.2 and the data sets refer to those described in 5.1.1. The "Classifiers" column refers to the use of the classifiers described in 4.3 strictly in the training phase. The number of epochs is standardized per data set, with models trained on CREMA-D passing through the training data more times to compensate for the smaller data set. All models start with an initial learning rate of $1e-6$ which rises to $1e-3$ after 10 warmup epochs. The mutual information based and adversarial models use a batch size of 30, whereas the group center loss based models use a batch size of 300. All models handle every loss component with a single optimizer. For models based on group center loss, the number in their name refers to the weight of the GCL_loss components. The number in the names of the adversarial models refers to the scaling factor of the gradient reversal layer. Models that contain "lf" in their names separate the forward step into 2 different steps. The sparation is explained in the table 5.2.

All three of these models do the label forward step before the mutual information step. The reconstruction loss is a component for the backwards step in both forward functions.

Model Name	Disentanglement Method	Data Set	Epochs	Classifiers
mi	Mutual Information	ESD	500	neither
mi_emocls	Mutual Information	ESD	500	emotion only
mi_spkcls	Mutual Information	ESD	500	speaker only
mi_cls	Mutual Information	ESD	500	both
mi_lf1	Mutual Information	ESD	500	both
mi_lf2	Mutual Information	ESD	500	both
mi_lf3	Mutual Information	ESD	500	both
gcl1	Group Center Loss	ESD	500	neither
gcl05	Group Center Loss	ESD	500	neither
gcl1_cls	Group Center Loss	ESD	500	both
adv1	Adversarial	ESD	500	neither
adv100	Adversarial	ESD	500	neither
adv1_cls	Adversarial	ESD	500	both
adv5_cls	Adversarial	ESD	500	both
adv10_cls	Adversarial	ESD	500	both
mi_lf2_c	Mutual Information	CREMA-D	800	both
gcl1_c	Group Center Loss	CREMA-D	800	neither
gcl1_cls_c	Group Center Loss	CREMA-D	800	both
adv1_cls_c	Adversarial	CREMA-D	800	both

Table 5.1: Model Grid

Model	Description
lf1	No CE_loss in the mutual information forward step Keep CPC_loss and VQ_loss in both the label forward and mutual information forward steps
lf2	No CE_loss in the mutual information forward step Only keep CPC_loss and VQ_loss in both the label forward step
lf3	Only keep CPC_loss and VQ_loss in both the mutual information forward step

Table 5.2: Summary of lf1, lf2, lf3 configurations and behaviors

5.3 Evaluation Metrics

In this section we will first describe what properties of the learnt models we aim to evaluate. Then we will describe the metrics we employ to calculate these measurements. The metrics will be split into subjective and objective metrics.

5.3.1 Evaluation Goals

We define the goals for which we want to measure our grid. Firstly, we have the three criteria we mention in 3.2, namely disentanglement, completeness and explicitness. As mentioned, we do not measure the completeness for our models. We can further categorize disentanglement as intra-encoder or at an architecture level. The intra-encoder disentanglement refers to the disentanglement described in 3.2, where our objective is to measure how well each dimension of the final embeddings represents a single factor of variation. For example, a 256-dimensional emotion embedding would have good intra-encoder disentanglement if the first 50 dimensions of the embedding only contain information about happiness, the next 50 only about sadness and so forth. This means, however, that, for example, a model that correctly encodes all emotion information with its respective encoder into the correct embedding could report low intra-encoder disentanglement scores if certain parts of the embedding aren't exclusively occupied for specific emotion labels. Disentanglement at an architectural level refers to the ability of each encoder to encode only relevant information. For example, an arbitrary emotion embedding should only contain the emotion information of the original utterance, regardless of how that information is spread across the embedding. That means that an arbitrary predictor should not be able to predict speaker labels from emotion embeddings. Additionally, it is important to measure how well the architecture can reconstruct the original utterance, since the main scope of disentanglement is to allow for better downstream task performance. Measuring downstream task performance is not within the scope of our work, but is still an important consideration. For example, it is possible to measure reconstruction quality after we swap the emotion embedding before decoding back to a Mel-spectrogram, or to ensure the speaker identity and content remain unchanged under the same circumstances.

5.3.2 Subjective Metrics

The audio quality of the reconstructed speech is subjectively evaluated by the authors of this paper. We use both WaveGAN [11] and the inverse of the process described in 3.1 to convert decoded Mel-spectrograms back to .wav signals. No exhaustive study with other subjects is conducted to evaluate speech quality and emotional information. We use a second subjective measurement in t-distributed stochastic neighbor embedding (t-SNE) plots [44]. The goal is to reduce the dimensionality of the speaker, content and emotion embeddings into a 2-dimensional plane and subjectively measure disentanglement by the presence of same-label clusters or a lack thereof.

5.3.3 Objective Metrics

First, we use the accuracy of neural network classifiers similar to those in section 4.3 as a measure of architecture level disentanglement and explicitness. We use fully connected linear models with one hidden layer for classification, trained using cross entropy 4.3. We train 6 classifiers for each model in the grid, trying to predict speaker and emotion

labels from speaker, emotion and content embeddings. Explicitness is measured as the accuracy of the emotion embedding to emotion label and speaker embedding to speaker label classifiers, whereas disentanglement is measured by looking at the accuracy of all 3 classifiers together for predicting both labels. We say an encoder is disentangled at architecture level if the accuracy when trying to predict the label of the information it is meant to encode is high, while the other 2 accuracies are low. The metrics will be named by the following scheme: $cls_{embedding \to label}$. For example, $cls_{c \to e}$ is the accuracy when trying to predict emotion labels from content embeddings on the test split.

The DCI framework [13] provides separate predictor-based scores for disentanglement, completeness, and explicitness. This framework involves training regressors to predict underlying generative factors from encoded embeddings. Intra-encoder disentanglement and completeness are assessed by examining the regressor's internal parameters to infer predictive importance weights R_{ij} for each pair of generative factors and embedding dimensions. While describing this framework, we will denote the number of underlying generative factors in $\mathcal{V}$ with K and the number of dimensions that each point z in the embedding space Z has with D. **The intra-encoder disentanglement metric (D_{IENC})** measures the degree to which each latent variable in the learned code space Z captures at most one underlying generative factor in $\mathcal{V}$. The disentanglement score for a latent variable z_i is quantified by:

$$D_{IENC_i} = 1 - H_K(P_{i\cdot}), \tag{5.2}$$

where $H_K(P_{i\cdot})$ is the entropy, defined as:

$$H_K(P_{i\cdot}) = -\sum_{k=0}^{K-1} P_{ik} \log_K(P_{ik}), \tag{5.3}$$

with $P_{ij} = R_{ij} / \sum_{k=0}^{K-1} R_{ik}$. Here, R_{ij} represents the relative importance of z_i in predicting v_j. If z_i is primarily responsible for predicting a single generative factor, the score is 1; if it has equal importance for predicting all factors, the score is 0.

The completeness metric quantifies the degree to which each underlying factor is captured by a single latent variable in Z. The completeness score for a generative factor v_j is calculated as:

$$C_j = 1 - H_D(\tilde{P}_{\cdot j}), \tag{5.4}$$

where $H_D(\tilde{P}_{\cdot j})$ is the entropy of the distribution, defined as:

$$H_D(\tilde{P}_{\cdot j}) = -\sum_{d=0}^{D-1} \tilde{P}_{dj} \log_D(\tilde{P}_{dj}), \tag{5.5}$$

with $\tilde{P}_{dj} = R_{dj} / \sum_{d=0}^{D-1} R_{dj}$. If a single latent variable is predominantly responsible for predicting v_j, the completeness score is 1; if all latent variables contribute equally to predicting v_j, the score is 0. **The explicitness metric (E)** measures the degree to

which the learned representation in Z captures information about the underlying factors $\mathcal{V}$. This metric is defined by the prediction error when regressing from Z to $\mathcal{V}$. Let $f_j(z) \to \hat{v}_j$ be a regressor that predicts v_j based on the code space. The informativeness metric for a generative factor v_j is given by the average prediction error $E(v_j, \hat{v}_j)$, where E is an appropriate error function, such as Mean Squared Error.

We measure reconstructed speech quality using **Mel-Cepstral Distortion (MCD)** [23], which serves as a distance function between a reference and a synthesized audio signal. In order to calculate MCD, a discrete cosine transform (DCT) [1] is applied along the frequency axis for each time frame:

$$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)k\right], \quad k = 0, 1, \ldots, N-1 \tag{5.6}$$

where x_n are the input sequence values, X_k are the DCT coefficients, N is the total number of points and k is the frequency index. The first 16 coefficients from the DCT are kept as Mel-frequency Cepstral Coefficients (MFCC). The MCD is then defined as the average Euclidean distance between corresponding MFCC vectors of the two signals over all time frames:

$$MCD_{db} = \frac{10}{ln10}\sqrt{2} \times \frac{1}{N}\sum_{t=0}^{N-1}\sqrt{\sum_{k=1}^{P}(MC_{syn}(t,k) - MC_{ref}(t,k))^2} \tag{5.7}$$

where N is the number of frames, P is the number of MFCC coefficients per frame, and, $MC_{syn}(t,k)$ and $MC_{ref}(t,k)$ are the k-th MFCC values of the synthesized and reference signals at time frame t. $\frac{10}{ln10}\sqrt{2}$ represents the scaling factor that converts the Euclidean distance to the decibel scale.

CHAPTER 6

Experiments

In this chapter, we will report on the results of the experiments described in chapter 5. We will organize the results by disentanglement method and review the subjective evaluation first, followed by the objective evaluation.

We summarize every metric we will use to evaluate the model grid in table 6.1:

Metric	**Abbreviation**	**Description**
t-Distributed Stochastic Neighbor Embedding plots	t-SNE plots	Subjective metric for measuring architecture-level disentanglement by identifying same-label clusters.
Neural Network Classifiers	$cls_{embedding \rightarrow label}$	Predictor-based objective metric used to measure architecture-level disentanglement and explicitness; Embedding stands for content (c), speaker (s) or emotion (e) and label stands for speaker (s) or emotion (e). It is measured as accuracies.
Disentanglement (DCI)	D_{label}	Predictor-based objective metric used to measure intra-encoder disentanglement; Measured on a scale from 0 to 1, where 1 represents perfect disentanglement (every dimension of the embedding corresponds to a single generative factor).
Informativeness (DCI)	E_{label}	Average accuracy of the predictors trained to calculate disentanglement and completeness; Used to measure explicitness.
Mel-Cepstral Distortion	MCD	Objective spectral distance metric to measure decoder reconstruction quality; Measured in decibels.

Table 6.1: Metric Overview

6.1 Mutual Information Based Models

We first look at the t-SNE plots of the original StyleVC architecture (*mi*) in figure 6.1:

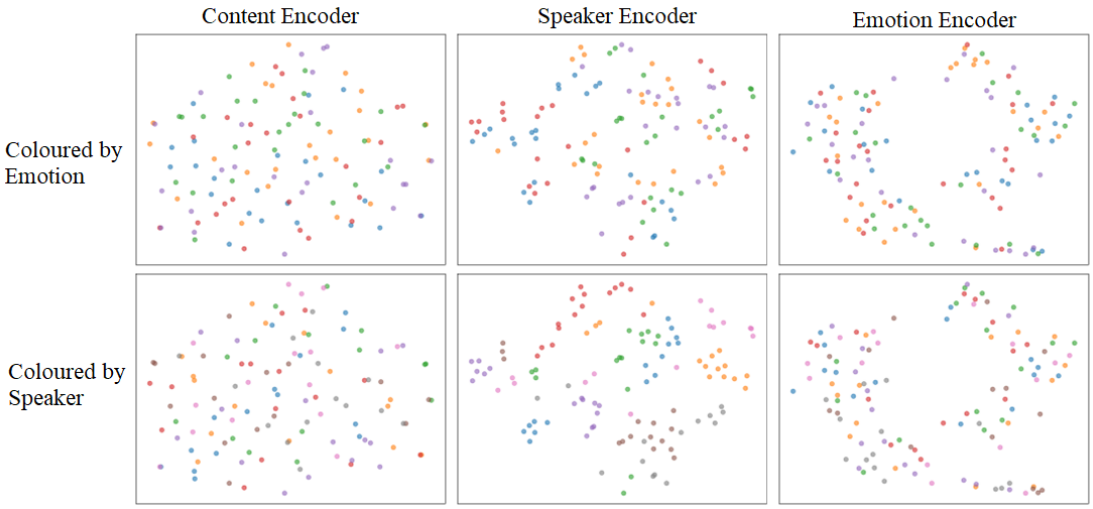

Figure 6.1: t-SNE plots of *mi*

All plots but one show a random point distribution. The only exception is the t-SNE plot of the speaker embedding when coloured by speaker labels. These results indicate that speaker information is only contained within the speaker encoder, whereas emotion information is not present in any of the encodings. We find that the content t-SNE plots look similar for every model in the grid, we so no longer display them.

We investigate the effect of adding an additional emotion classifier during training (*mi_emocls*):

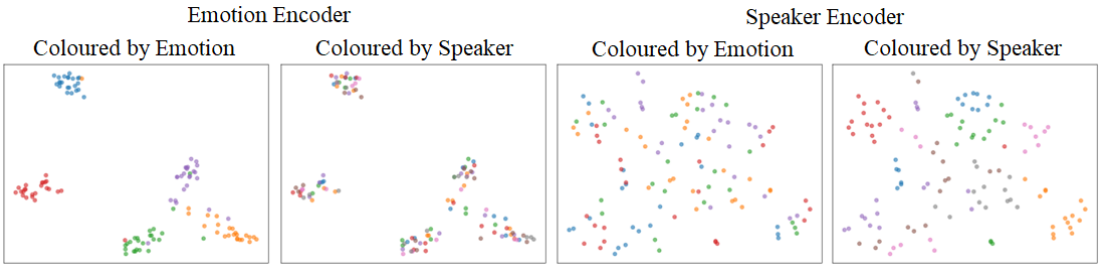

Figure 6.2: t-SNE plots of *mi_emocls*

The plots in figure 6.2 show that emotion information is now displayed within the emotion embeddings, while speaker information is still present in the speaker embeddings. We examine the rest of the t-SNE plots for mutual information based models together in figure 6.3:

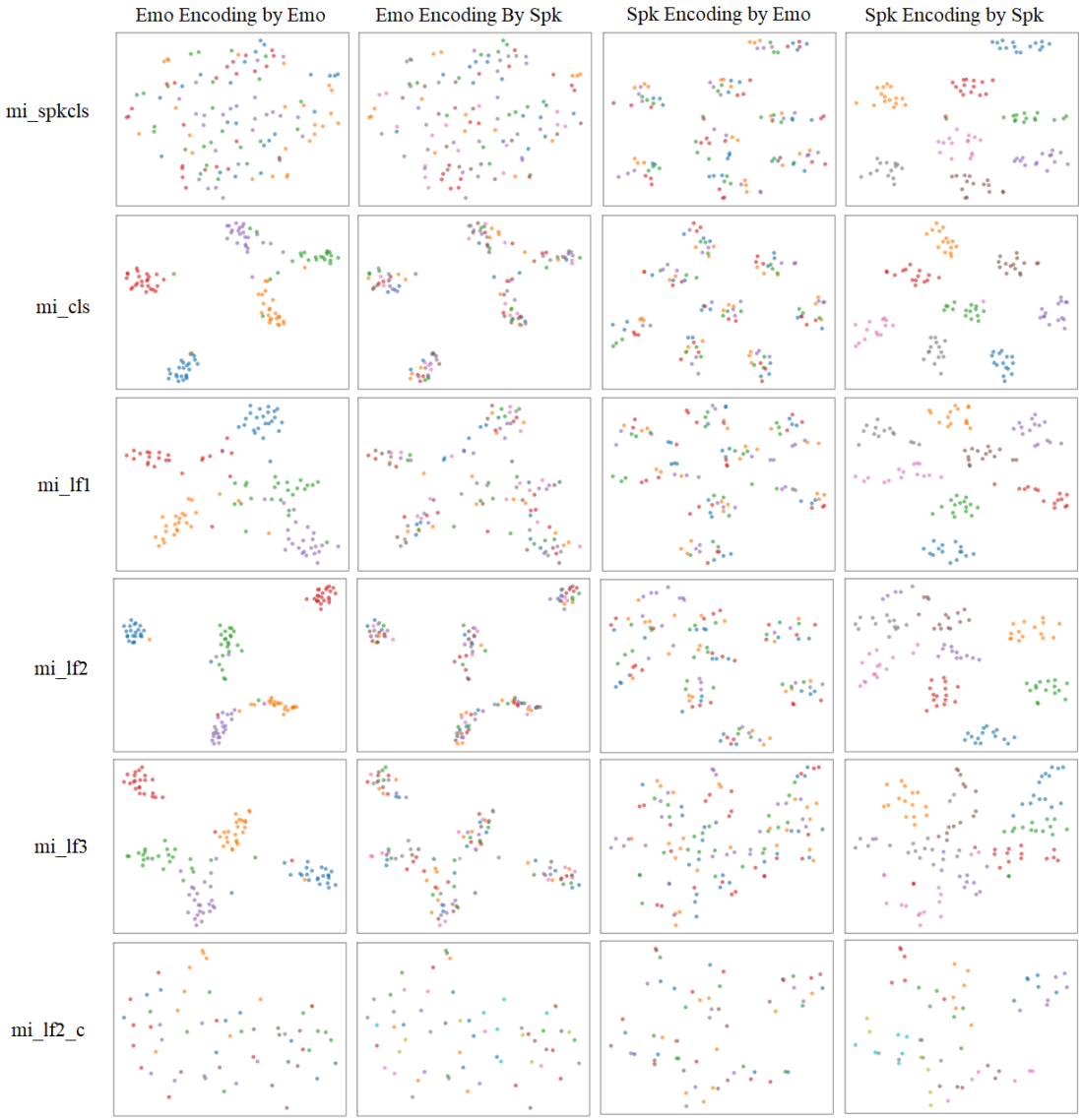

Figure 6.3: t-SNE plots for the rest of the MI grid

The plots of *mi_spkcls* look similar to those of *mi*, although the clustering in the speaker t-SNE labeled by speaker plot is stronger, signaling that adding a speaker classifier might force more speaker information to be retained by the encoder during training. The addition of an emotion classifier during training leads to better information retention for all models except mi_lf2_c. We infer therefore that the size of the training dataset has a significant effect on the performance of the encoders.

Next we compare the objective metrics for the entire mutual information sub-grid in table 6.2:

Model	$cls_{c\to e}$	$cls_{s\to e}$	$cls_{e\to e}$	$cls_{c\to s}$	$cls_{s\to s}$	$cls_{e\to s}$	D_{emo}	D_{spk}	E_{emo}	E_{spk}	MCD
mi	0.21	0.77	0.27	0.13	0.99	0.31	0.01	0.86	0.79	0.99	11.70
mi_emocls	0.22	0.72	0.80	0.14	0.99	0.60	0.99	0.84	0.92	0.99	11.75
mi_spkcls	0.20	0.70	0.67	0.14	0.99	0.72	0.13	0.94	0.85	0.99	11.85
mi_cls	0.21	0.71	0.80	0.15	0.99	0.63	0.91	0.99	0.92	0.99	11.86
mi_lf1	0.21	0.70	0.80	0.15	0.99	0.87	0.97	0.97	0.92	0.99	11.81
mi_lf2	0.22	0.79	0.92	0.13	0.99	0.48	0.91	0.98	0.92	0.99	11.80
mi_lf3	0.21	0.81	0.91	0.15	0.99	0.88	0.99	0.91	0.92	0.99	11.82
mi_lf2_c	0.20	0.39	0.51	0.01	0.69	0.03	0.82	0.57	0.85	0.99	17.42

Table 6.2: Objective Measurements of Mutual Information-based Model Grid

The results show that neural network classifiers cannot accurately predict emotion or speaker labels from content embeddings no matter the architecture or training data, with the average percentages showing randomness (∼20% with 5 emotion labels, ∼15% for 8 speaker labels and ∼1% for 81 speaker labels). Emotion information is always contained in speaker embeddings, as shown by 70% to 80% accuracies reported by $cls_{s\to e}$. The model trained on the CREMA-D dataset (mi_lf2_c) shows a lower percentage, but still not low enough to be considered random. The models trained on ESD show 70% to 90% accuracies when predicting emotion labels from the respective embedding, with the exception of those trained without an emotion classifiers. Notably, while mi shows a percentage that approaches random guess level, mi_spkcls has a 67% prediction accuracy, showing that the inclusion of a speaker classifier in training also forces more information to be learnt by the emotion encoder. The $cls_{e\to s}$ values show that the emotion encoder learns speaker information as well, though not as much as the vice-versa case. $cls_{s\to s}$ shows that significant speaker information is learnt in its respective encoder for every model. All model trained with an emotion encoder show high intra-encoder emotion disentanglement values. All models except mi_lf2_c show high intra-encoder speaker disentanglement values. We assume that's caused by the large number of different speakers present in CREMA-D. The speaker explicitness E_{spk} is in line with $cls_{s\to s}$, with the only exception being mi_lf2_c. The emotion explicitness E_{emo} is consistently higher than $cls_{e\to e}$, showing that the gradient boosted tree classifiers may perform better than neural network classifiers. The results show consistent MCD values across all models trained on the same data set, with the larger set leading to less distortion.

6.2 Group Center Loss Based Models

We examine the t-SNE plots of the entire sub-grid together in figure 6.4:

Figure 6.4: t-SNE plots for the GCL grid

The plots show that emotion and speaker information are correctly retained by their respective encoders for all 5 models and the clusters are tighter than their mutual information counterparts. Like before, we examine the objective measurements in table 6.3: Once again, the content encoders are unable to predict emotion or speaker

Model	$cls_{c\to e}$	$cls_{s\to e}$	$cls_{e\to e}$	$cls_{c\to s}$	$cls_{s\to s}$	$cls_{e\to s}$	D_{emo}	D_{spk}	E_{emo}	E_{spk}	MCD
gcl1	0.20	0.20	0.91	0.14	0.99	0.31	0.99	0.95	0.92	0.99	12.52
gcl05	0.21	0.20	0.91	0.14	0.99	0.35	0.99	0.94	0.91	0.99	12.57
gcl1_cls	0.21	0.20	0.92	0.14	0.99	0.31	0.99	0.96	0.92	0.99	12.44
gcl1_c	0.23	0.20	0.53	0.01	0.76	0.01	0.99	0.95	0.84	0.99	17.12
gcl1_cls_c	0.22	0.17	0.53	0.005	0.76	0.01	0.99	0.96	0.84	0.99	17.11

Table 6.3: Objective Measurements of Group Center-based Model Grid

information correctly, with $cls_{c\to e}$ and $cls_{c\to s}$ showing percentages just above random

levels. In contrast to mutual information-based models, group center-based models also show low accuracies when predicting emotion labels from speaker embeddings and vice versa, signifying better architecture level disentanglement. The explicitness of the neural network ($cls_{e\to e}$ and $cls_{s\to s}$) is significantly lower for CREMA-D-based models ($gcl1\_c, gcl1\_cls\_c$), implying that the size of the data set has a large impact on the effectiveness of the training. Intra-encoder disentanglement values are consistently high for both data sets and DCI explicitness is affected less to not at all by the different data sets. MCD values once again show larger distortion in models trained with less data, while being consistent with the values of the mutual information sub-grid. Considering that all models used the same loss components when training the content encoder, we can assume that MCD is only impacted by the data set and not by the disentanglement method.

6.3 Adversarial Models

We first look at the t-SNE plots of adversarial models trained without classifiers:

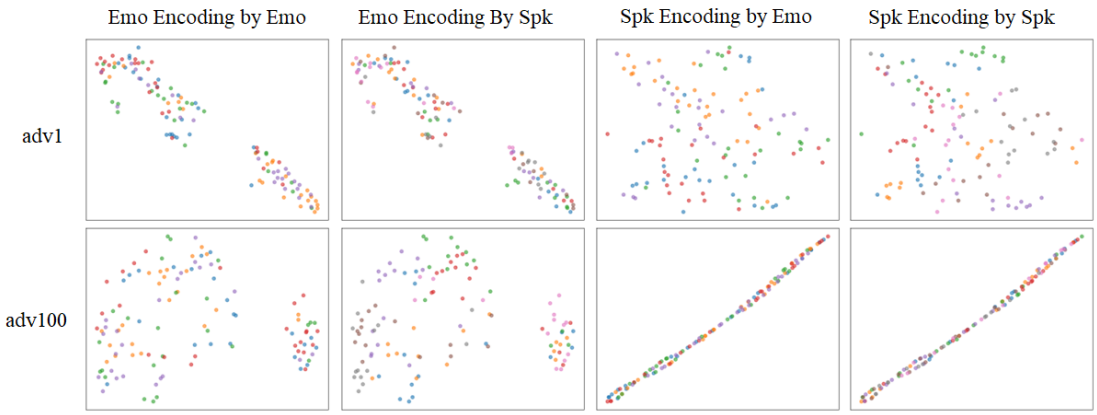

Figure 6.5: t-SNE plots of adversarial models without classifiers

The plots in figure 6.5 show that using adversarial disentanglement loss components causes the encoders to learn little to no information about emotion and speaker identity without classifier enhancement.

Figure 6.6: t-SNE plots of adversarial models with classifiers

With classifier loss components, we see in figure 6.6 that speaker information is now retained by all models in their respective encoders. Emotion information however, is retained less when increasing the scaling factor of the gradient reversal layer. We examine the objective measurements in table 6.4:

Model	$cls_{c\to e}$	$cls_{s\to e}$	$cls_{e\to e}$	$cls_{c\to s}$	$cls_{s\to s}$	$cls_{e\to s}$	D_{emo}	D_{spk}	E_{emo}	E_{spk}	MCD
adv1	0.22	0.75	0.35	0.15	0.93	0.30	0.04	0.69	0.80	0.97	12.21
adv100	0.19	0.20	0.28	0.12	0.12	0.51	0.14	0.09	0.80	0.87	19.78
adv1_cls	0.21	0.86	0.92	0.15	0.99	0.19	0.99	0.93	0.92	0.99	12.19
adv5_cls	0.20	0.79	0.92	0.14	0.99	0.61	0.88	0.88	0.92	0.99	12.19
adv10_cls	0.20	0.76	0.79	0.16	0.99	0.91	0.67	0.83	0.89	0.99	12.19
adv1_cls_c	0.23	0.38	0.53	0.02	0.76	0.04	0.97	0.52	0.85	0.99	16.70

Table 6.4: Objective Measurements of Adversarial Model Grid

The results show that the content encoders retain no speaker and emotion information. The $adv1$ and $adv100$ models show poor architecture level disentanglement and explicitness, with $cls_{e\to e}$ accuracies being low or similar to $cls_{s\to e}$. They also show poor intra-encoder disentanglement for both the emotion and speaker encoders. The rest of

the models trained on ESD show high emotion and speaker explicitness, with both the neural network classifiers and the tree classifiers having high accuracies. Intra-encoder disentanglement is high for all 3 models, although it decreases with the increase in scaling factor. The architecture level disentanglement metrics show that emotion and speaker information are both retained by both encoders. The models shows distortion values in line with the rest of the grid, with the exception of $adv100$, which has the lowest MCD of all models. This shows that scaling factors after a high enough threshold have an impact on every aspect of the model.

6.4 Results

We will briefly discuss the results of the experiments. We show that the StyleVC architecture successfully disentangles content information into its own encoder but needs additional classifier loss terms or entirely different disentanglement methods to separate emotion and speaker information. In terms of architecture level disentanglement, most models in the grid show that speaker information is meaningfully contained in the speaker encoders, with speaker embeddings being able to predict the respective labels with high accuracy. Most mutual information-based and adversarial models also show some level of speaker information encoded in the emotion encoder. Emotion information is poorly disentangled for mutual information-based and adversarial models, while group center loss-based models consistently show randomness when trying to predict emotion labels from speaker embeddings. DCI disentanglement values show high intra-encoder disentanglement across the full grid, with the exception of $mi, mi\_spkcls, adv100$, which have an overall poor performance. Lastly, Mel-cepstral distortion values show that the outputs of models trained on ESD always produce higher quality audio signals compared to those trained on CREMA-D.

CHAPTER 7

Conclusion

In this work, we successfully measure the effectiveness of three different disentanglement methods, the degree of information stored by the representations encoded via disentanglement, and the reconstruction quality of synthesized speech. We describe and utilize established metrics, as well as neural-networks-based predictors. We show that all three methods are capable of both intra-encoder and architecture level disentanglement. We also show that enhancing the base StyleVC architecture with classification losses as add-ons yields better results both objectively and subjectively. We summarize the differences in performance between disentanglement methods:

		Emotion		Speaker		
Method	Classifiers	E	D_{ienc}	E	D_{ienc}	D_{arch}
Mutual Information	no	poor	poor	good	good	poor
Mutual Information	yes	good	good	good	good	poor
Group Center	no	good	good	good	good	good
Group Center	yes	good	good	good	good	good
Adversarial	no	poor	poor	good	good	poor
Adversarial	yes	good	good	good	good	poor

Table 7.1: Summary of experiment results

where E is overall explicitness, D_{ienc} is intra-encoder disentanglement and D_{arch} is architecture level disentanglement. We can conclude that group center losses are the best match for the StlyeVC architecture for the task of disentangling speaker identity and emotion information, while content disentanglement is handled effectively by the unmodified loss components related to the task of speech reconstruction. We can also conclude that adding a classification term to the loss function has a positive impact on the information retention of every encoder.

We will briefly discuss limitations and future work. In terms of limitations, we acknowledge that the depth of the hyperparameter and model grid for each disentanglement grid can be further expanded. Exploring more expansive grids for every method can be a worthwhile consideration, especially in the context of optimization. Further tests with different architectures and data sets (or combinations thereof) would also provide important insights into the theme. Furthermore, emotional speech is, by its nature, a concept that cannot be perfectly defined formally. Therefore, subjective metrics can still provide great value when deciding on final architectures meant to be employed in downstream tasks such as speech synthesis or emotion elimination. A comprehensive study with subjects trying to identify replaced emotions or identify recreated speech could further enforce model choice. While our current focus is on disentanglement and model evaluation, downstream task performance is an area of research outside of the scope of this work that remains an important consideration for future investigation. A possible measurement is calculating the Mel-cepstral distortion between an original signal and one gained by swapping the emotion embedding of another to match it. This process can be repeated for every possible combination of emotions over multiple signals to determine which emotions are most compatible in speech synthesis and style change. In addition to the previously mentioned topics, we propose to apply and evaluate our presented methods to different domains, such as computer vision, in future work.

Overview of Generative AI Tools Used

We use the Perplexity client with the LLM model set to "Best". The areas where we use AI tools are:

- English to German translation of acknowledgements, ai tools and the abstract.
- Conversion of code to the pseudocode displayed in chapter 4.
- Formatting of the LaTeX tables in chapter 6.
- Generation of LaTeX formulas throughout the paper.

Übersicht verwendeter Hilfsmittel

Wir verwenden den Perplexity-Client mit dem LLM-Modell auf „Best“. Die Bereiche, in denen wir KI-Tools einsetzen, sind:

- Englisch-zu-Deutsch-Übersetzung der Danksagung, KI-Tools und der Kurzfassung.
- Umwandlung von Code in den im Kapitel 4 dargestellten Pseudocode.
- Formatierung der LaTeX-Tabellen in Kapitel 6.
- Generierung von LaTeX-Formeln im gesamten Paper.

List of Figures

List of Tables

List of Algorithms

Bibliography

[1] N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

[2] J. Allen. Short term spectral analysis, synthesis, and modification by discrete fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25(3):235–238, 1977.

[3] Diane Bouchacourt, Ryota Tomioka, and Sebastian Nowozin. Multi-level variational autoencoder: Learning disentangled representations from grouped observations. *CoRR*, abs/1705.08841, 2017.

[4] Houwei Cao, David G. Cooper, Michael K. Keutmann, Ruben C. Gur, Ani Nenkova, and Ragini Verma. Crema-d: Crowd-sourced emotional multimodal actors dataset. *IEEE Transactions on Affective Computing*, 5(4):377–390, 2014.

[5] Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. CLUB: A contrastive log-ratio upper bound of mutual information. *CoRR*, abs/2006.12013, 2020.

[6] Ju chieh Chou, Cheng chieh Yeh, and Hung yi Lee. One-shot voice conversion by separating speaker and content representations with instance normalization. In *Proceedings of Interspeech 2019*, pages 664–668, 2019.

[7] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics.

[8] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018.

[9] Junyoung Chung, Çaglar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *ArXiv*, abs/1412.3555, 2014.

[10] IBM Corporation and Microsoft Corporation. Multimedia programming interface and data specifications 1.0. Technical report, August 1991. Joint design document for OS/2 and Windows environments, specifies RIFF and multimedia standards.

[11] Chris Donahue, Julian J. McAuley, and Miller S. Puckette. Synthesizing audio with generative adversarial networks. *CoRR*, abs/1802.04208, 2018.

[12] Zongyang Du, Berrak Sisman, Kun Zhou, and Haizhou Li. Disentanglement of emotional style and speaker identity for expressive voice conversion. In *Proceedings of Interspeech 2022*. International Speech Communication Association, September 2022.

[13] Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

[14] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1180–1189, Lille, France, 07–09 Jul 2015. PMLR.

[15] Ünal Ege Gaznepoglu and Nils Peters. Why disentanglement-based speaker anonymization systems fail at preserving emotions? In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2025.

[16] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 87–102, Cham, 2016. Springer International Publishing.

[17] Irina Higgins, David Amos, David Pfau, Sébastien Racanière, Loïc Matthey, Danilo J. Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *CoRR*, abs/1812.02230, 2018.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, November 1997.

[19] Charles C. Holt. Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, 20(1):5–10, 2004.

[20] Roy Rudolf Huizen and Florentina Tatrin Kurniati. Feature extraction with mel scale separation method on noise audio recordings. *Indonesian Journal of Electrical Engineering and Computer Science*, 24(2):815, November 2021.

[21] Eric Jordan, Raphaël Terrisse, Valeria Lucarini, Motasem Alrahabi, Marie-Odile Krebs, Julien Desclés, and Christophe Lemey. Speech emotion recognition in mental health: Systematic review of voice-based applications. *JMIR Mental Health*, 12, 2025.

[22] Ł. Juszkiewicz. Speech emotion recognition system for social robots. *Journal of Automation Mobile Robotics and Intelligent Systems*, 7(4):59–65, 2013.

[23] R. Kubichek. Mel-cepstral distance measure for objective speech quality assessment. In *Proceedings of IEEE Pacific Rim Conference on Communications Computers and Signal Processing*, volume 1, pages 125–128 vol.1, 1993.

[24] Olivia Langner, Ron Dotsch, Gijs Bijlstra, Daniël HJ Wigboldus, Skyler T Hawk, and Ad van Knippenberg. Presentation and validation of the radboud faces database. *Cognition and Emotion*, 24(8):1377–1388, 2010.

[25] Maxime Leiber, Yosra Marnissi, Axel Barrau, and Mohammed El Badaoui. Differentiable short-time fourier transform. In *IEEE Statistical Signal Processing Workshop (SSP)*, 2023. arXiv:2308.02421.

[26] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *CoRR*, abs/1411.7766, 2014.

[27] Zhaojie Luo, Jinhui Chen, Toru Nakashika, Tetsuya Takiguchi, and Yasuo Ariki. Emotional voice conversion using neural networks with different temporal scales of f0 based on wavelet transform. In *Proceedings of the 9th ISCA Speech Synthesis Workshop*, September 2016.

[28] Chris Lytridis, Eleni Vrochidou, and Vassilis Kaburlasos. Emotional speech recognition toward modulating the behavior of a social robot. In *Proceedings of the 2018 JSME Conference on Robotics and Mechatronics*, June 2018.

[29] Yong Ma, Yuchong Zhang, Miroslav Bachinski, and Morten Fjeld. Emotion-aware voice assistants: Design, implementation, and preliminary insights. In *Proceedings of the Chinese CHI Conference on Human-Computer Interaction (CHCHI 2023)*. Association for Computing Machinery, February 2024.

[30] Samaneh Madanian, David Parry, Olayinka Adeleye, Christian Poellabauer, Farhaan Mirza, Shilpa Mathew, and Sandra Schneider. Automatic speech emotion recognition using machine learning: Digital transformation of mental health. In *Pacific Asia Conference on Information Systems (PACIS)*, 06 2022.

[31] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA, 2010. Omnipress.

[32] Emmy Noether. Der endlichkeitssatz der invarianten endlicher gruppen. *Mathematische Annalen*, 77:89–92, 1916.

[33] Hans B. Peek. The emergence of the compact disc. *IEEE Communications Magazine*, 48(1):10–17, 2010.

[34] Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, and Mark Hasegawa-Johnson. AutoVC: Zero-shot voice style transfer with only autoencoder loss. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5210–5219. PMLR, 09–15 Jun 2019.

[35] Navin Raj Prabhu, Nale Lehmann-Willenbrock, and Timo Gerkmann. In-the-wild speech emotion conversion using disentangled self-supervised representations and neural vocoder-based resynthesis. In *Speech Communication; 15th ITG Conference*, pages 176–180, 2023.

[36] Jeffrey Sable, Gabriele Gratton, and Monica Fabiani. Sound presentation rate is represented logarithmically in human cortex. *The European journal of neuroscience*, 17:2492–6, 07 2003.

[37] Vidhi Sareen and Seeja K.R. Speech emotion recognition using mel spectrogram and convolutional neural networks (cnn). *Procedia Computer Science*, 258:3693–3702, 2025. International Conference on Machine Learning and Data Engineering.

[38] Chaitanya Singla, Sukhdev Singh, Preeti Sharma, Nitin Mittal, and Fikreselam Gared. Emotion recognition for human–computer interaction using high-level descriptors. *Scientific Reports*, 14, 05 2024.

[39] Julius O. Smith. *Spectral Audio Signal Processing*. https://ccrma.stanford.edu/ jos/sasp/, accessed 09.2025. online book, 2011 edition.

[40] Jonathan Sterne. The mp3 as cultural artifact. *New Media & Society*, 8(5):825–842, 2006.

[41] S.S. Stevens, J. Volkman, and E.B. Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937.

[42] Andreas Triantafyllopoulos, Björn W. Schuller, Gökçe İymen, Metin Sezgin, Xiangheng He, Zijiang Yang, Panagiotis Tzirakis, Shuo Liu, Silvan Mertes, Elisabeth André, Ruibo Fu, and Jianhua Tao. An overview of affective speech synthesis and conversion in the deep learning era. *Proceedings of the IEEE*, 111(10):1355–1381, 2023.

[43] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[44] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[45] Lonce Wyse. Audio spectrogram representations for processing with convolutional neural networks. In *Proceedings of the First International Workshop on Deep Learning and Music, joint with IJCNN*, volume 1, pages 37–41, Anchorage, US, May 2017.

[46] Jiajian Xie, Shengyu Zhang, Mengze Li, chengfei lv, Zhou Zhao, and Fei Wu. Ecoface: Audio-visual emotional co-disentanglement speech-driven 3d talking face generation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[47] Kailai Yang, Tianlin Zhang, and Sophia Ananiadou. Disentangled variational autoencoder for emotion recognition in conversations. *IEEE Transactions on Affective Computing*, 15(2):508–518, April 2024.

[48] Jing Yao, Hongliang Liu, Eng Siong Chng, and Lei Xie. Easy: Emotion-aware speaker anonymization via factorized distillation. In *Proceedings of Interspeech 2025*, pages 3219–3223, 2025.

[49] Tingrong Yin. Music track recommendation using deep-cnn and mel spectrograms. *Mobile Networks and Applications*, 28:2130–2137, 2023.

[50] Jia Qi Yip, Dianwen Ng, Bin Ma, and Chng Eng Siong. Analysis of speech separation performance degradation on emotional speech mixtures. In *2023 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 2002–2007, 2023.

[51] Julian Zaidi, Jonathan Boilard, Ghyslain Gagnon, and Marc-André Carbonneau. Measuring disentanglement: A review of metrics. *CoRR*, abs/2012.09276, 2020.

[52] Boyan Zhang, Jared Leitner, and Samuel Thornton. Audio recognition using mel spectrograms and convolution neural networks. 2019.

[53] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[54] Kun Zhou, Berrak Sisman, Rui Liu, and Haizhou Li. Emotional voice conversion: Theory, databases and esd. *Speech Communication*, 137:1–18, 2022.