ircam Centre Pompidou

Master of Science
Télécom SudParis / CNAM
DATAPAC M1 / TRIED M2

End of studies internship report

Voice emotion transformation: Generating expressive F0 contours with RNN-LSTM sequence-to-sequence models

Author
Carl Robinson
(carl.robinson@gmail.com)

Laboratory
IRCAM - Analysis-Synthesis Team

Supervisor
Nicolas Obin - IRCAM
(nicolas.obin@ircam.fr)

Additional Supervision
Nesma Houmani - Télécom SudParis
Axel Roebel - IRCAM

September 10, 2018

TELECOM SudParis

le cnam

Université de Versailles St-Quentin-en-Yvelines

université PARIS-SACLAY

Abstract

Voice interfaces are becoming wildly popular and driving demand for more advanced speech synthesis systems. Current text-to-speech methods produce realistic human sounding voices, but lack the ability to express emotions that fit the context of the utterance. Emotional voice conversion is a post-processing technique that can generate expressive speech from a synthesised neutral voice, or indeed modify the emotion in any natural human voice.

This research aims to investigate the effectiveness of using a sequence-to-sequence (seq2seq) encoder-decoder based model to transform the fundamental frequency (F0) contours of the human voice. These transformed contours can then be applied to neutral source utterances to modify their prosody and render them emotionally expressive.

The model was constructed, and a qualitative evaluation was performed using an online survey framework. Participants listened to a selection of transformed and non-transformed voice samples, then selected the emotion they recognised from four options (happiness, sadness, anger, fear). Additionally, they judged how natural the voice sounded, and select a corresponding voice quality (bad, poor, fair, good, excellent).

It was discovered that conditioning the model on the position of the syllable in the phrase caused some emotion transformations to be much more easily recognised by participants. The participants' ability to recognise a transformed emotion was also found to be highly phrase dependent, while their level of French and the listening equipment they used were much less important factors.

This is highly simplified model of the human voice, capable of transforming only the fundamental frequency and the duration of voiced phonemes, while ignoring other important components such as voice quality and amplitude. Even so, this study has demonstrated that the model can produce convincing voice emotion results that rival those of previous studies in the field.

Keywords

Voice emotion transformation, voice conversion, speech, LSTM, sequence to sequence, seq2seq, tensorflow, deep learning, fundamental frequency, f0, signal processing, stft, phase vocoder

Acknowledgements

This project would not have been possible without the generous help and support of many talented people. I would like to thank the following people for giving me their time, for sharing their knowledge and ideas, and for their continued encouragement:

· Nicolas Obin

· Axel Roebel, Guillaume Doras and Rafael Ferro

· Sylvie Thiria, Cecile Mallet, Sonia Garcia, Nesma Houmani, and Jerome Boudy

· Eric Bolo and the Batvoice team

· Veronique Sieng

Contents

Chapter 1

Introduction

Voice interfaces are becoming wildly popular and driving demand for more advanced speech synthesis systems. Current text-to-speech methods produce realistic human sounding voices, but lack the ability to express emotions that fit the context of the utterance. Voice emotion transformation is a post-processing technique that can be used to generate expressive speech from a synthesised neutral voice, or indeed modify the emotion in any natural human voice.

1.1 Context

This internship took place from 26th February to 27th July 2018 under the supervision of Nicolas Obin, in the Sound Analysis and Synthesis team at IRCAM. This team is part of the UMR STMS (Sciences et Technologies de la Musique et du Son), which brings together CNRS, UPMC, the Ministry of Culture and IRCAM, and focuses on the study and modelling of sound and music phenomena for analysis, transformation and synthesis. This work is inspired by an earlier voice emotion transformation project, conducted at IRCAM [Veaux and Rodet, 2011]. A novel method is applied to the same data set used in the earlier project, and the results are compared.

1.2 Objectives

The emotion of a speaker is expressed in the prosodic component of speech, which can be represented by a sequence of fundamental frequency (F0) contours. This work concerns the transformation of these F0 contours, specifically from a neutral to an expressive form, in order to modify the prosody of an utterance. A sequence-to-sequence (seq2seq), encoder-decoder based model is used, comprising two linked recurrent neural networks of long-short-term-memory cells (RNN-LSTM). This learns the mapping between neutral and expressive syllable/phoneme contours during training, and then generates expressive contours during inference. An attention mechanism and problem-specific enhancements are applied to the standard seq2seq model to improve results.

Chapter 2 outlines the state of the art of in voice emotion transformation, and introduces the theoretical bases of the seq2seq architecture and the attention mechanism. Chapter 3 then details the main contributions made during the internship. Chapter 4 describes the evaluation procedure and the results obtained, and chapter 5 concludes with some final remarks.

Chapter 2

State of the art in voice emotion transformation

2.1 The importance of emotion in voice technology

As technology continues to advance and improve, devices become more complex to use. The more complex a technology is, the greater the need for a natural and intuitive interface. Voice-interfaces are widely seen as solution, and the current trend is to move from purely keyboard and touch-screen based interfaces to voice enabled interfaces.

If the goal is to allow humans to interact with machines in the most natural way possible, then voice interfaces must be emotion-enabled, i.e. they should be able to read, interpret and reproduce expressions of human emotion. Imbuing technology with the ability to relate and be relatable, will ultimately encourage acceptance by users, lead to greater use and engagement, and to a perception of enhanced utility and value.

2.2 Prosody, intonation and F0 contours

The sound of a human voice is changed as a consequence of the somatic (bodily) effects of emotional responses. Humans have learned to detect this in each other's voices because it provides a distinct advantage when communicating, improving comprehension of both explicit statements and unspoken belief and intent.

There are two main components to speech: a verbal component comprised of a sequence of words (studied in the field of linguistics); and a prosodic component, which involves emotion, emphasis, and sentence position. When speech is interpreted by a human the prosodic component is given greater importance, and takes priority over the verbal component if the two conflict. It follows that if computers are to communicate with humans in the most natural manner possible, they will need to be able to synthesise prosody convincingly.

Prosody expresses emotion, emphasises words, reveals the speaker's attitude, breaks a sentence into phrases, governs sentence rhythm, and controls the intonation, pitch or tune of the utterance. A unit of speech can be emphasised by adding (or removing) stress to a syllable in a word, or to a word in a phrase. Accent can also be added, by modifying the pitch/intonation of an element.

Intonation is the manipulation of the fundamental frequency (F0) of the voice for communicative or linguistic purposes, and is used to express both affective prosody (e.g. adding emotion) and augmentative prosody (e.g. putting emphasis on words for clarity). F0 contours are continuous, variable-length sequences of frequency values that represent the pitch of the speaker's voice.

To effect a change in the prosody of a speech recording (e.g. to add/subtract/enhance an emotion), these F0 contours can be extracted, transformed, then reapplied to the same recording to produced a modified version. The extraction and reapplication of F0 contours is still an active research area, but can be considered solved for our purposes, and we make use of tools developed at IRCAM to perform these operations. This work concerns the transformation of the F0 contour, specifically from a neutral to an expressive form.

2.3 Voice transformation systems

While in TTS the input is writing, and the output is speech, in voice transformation (VT) both the input and the output is speech. VT systems aim to change one or more aspects of a speech signal while preserving linguistic information [Mohammadi and Kain, 2017]. In this paper we are interested in using VT to change one emotion into another, specifically to address the shortcoming of TTS systems producing neutral sounding speech. Solving this problem would open up a range of new applications for TTS systems, such as dynamically generated video-game dialog.

The general schematic for a voice transformation system that converts neutral speech to emotive speech is illustrated in Figure 2.1, and described as followed:

Training phase

1. Input a dataset of audio segments comprising source neutral speech and target emotive speech
2. Perform speech analysis to identify elements such as words, syllables and phonemes, and create speech features
3. Encode speech features into mapping features that allow for the modification of speech properties
4. Phonetically align source and target features (if necessary)

5. Train mapping function to convert between the two sequences of representations to create the model

Conversion phase

1. Input data set of source neutral speech audio segments to convert into emotive speech audio
2. Perform speech analysis to identify elements such as words, syllables and phonemes, and create speech features
3. Encode speech features into mapping features that allow for the modification of speech properties
4. Convert features using the trained model
5. Create new emotive speech waveform from outputted features

Speech features are encoded using a vocoder, which can use a source & filter model (e.g. STRAIGHT) which outputs the spectral envelope and excitation signals separately. The alternative is to use a purely signal-based model (e.g. PSOLA), which produce higher quality but less modifiable representations of the signal.

Local mapping features can be calculated per frame of audio (e.g. every 10ms), such as the fundamental frequency (F0), spectral envelope, mel-frequency cepstrum coefficients (MFCC), line-spectral frequencies (LSF/LSP) and formants. Contextual mapping features at the syllable and phrase level, such as mean F0 and other linguistic information can be added. Finally, implicit features are also present when using models such as Hidden Markov Models (HMM) and RNN, which have state that implicitly model the speech.

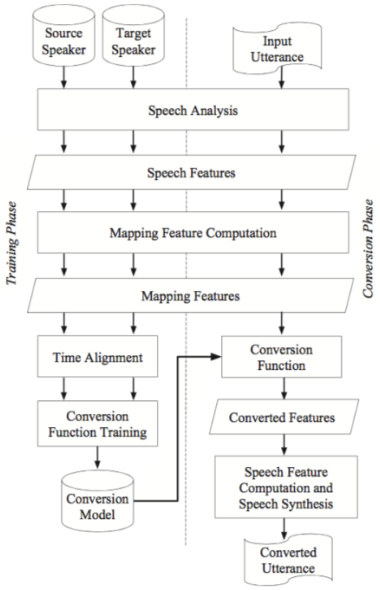

Figure 2.1: The general schematic for a voice transformation system

For time alignment of parallel speech databases, dynamic time-warping (DTW) is often used. More complicated processes exist for non-parallel datasets.

2.4 Prosody modelling

Prosody modelling is a complex problem that depends on linguistic and semantic information. For example, the emphasis that a speaker puts on certain speech units (such as words) can be quite different for another speaker saying the same thing, as it highly

depends on the context of the utterance. It is the pitch (F0) mapping feature and the choice of mapping function that interests us the most, as this is the key to generating natural sounding prosody. A variety of representations and functions have been used over the years, as shown in Figure 2.2.

Method	Level	Pitch representation	Other info	Mapping function
Mean and variance matching (Chappell and Hansen, 1998)	Frame-level	F_0 contour	–	Linear
Predicting from spectrum (En-Najjary et al., 2003)	Frame-level	F_0 contour	Spectrum	Weighted linear
Joint modeling with spectrum (En-Najjary et al., 2004; Hanzlíček and Matoušek, 2007; Xie et al., 2014b)	Frame-level	F_0 contour	Spectrum	Weighted linear
Histogram equalization (Wu et al., 2010)	Frame-level	F_0 contour	–	Histogram equalization
MSD-HMM (Yutani et al., 2009)	Frame-level	F_0 contour	Spectrum	Weighted linear
LSTM (Chen et al., 2016; Ming et al., 2016)	Frame-level	F_0 contour	Spectrum	LSTM
Syllable-based codebook (Rao et al., 2007)	Syllable-level	F_0 contour	Syllable boundary	Codebook mapping
Syllable-based MLLR (Lolive et al., 2008)	Syllable-level	F_0 contour	Syllable boundary	MLLR adaptation
Syllable-based CART (Helander and Nurminen, 2007a)	Syllable-level	DCT	Syllable boundary	CART
Syllable-based weighted linear (Veaux and Rodet, 2011)	Syllable-level	DCT	Syllable boundary	Weighted linear
Hierarchical modeling of F0 (Sanchez et al., 2014)	Utterance-level	Wavelet transform (Suni et al., 2013)	–	KPLS (Helander et al., 2012)
Contour codebook + DTW (Chappell and Hansen, 1998; Inanoglu, 2003)	Utterance-level	F_0 contour	–	Codebook mapping
Weighting contours (Türk and Arslan, 2003; Inanoglu, 2003)	Utterance-level	F_0 contour	–	Weighting codebooks
SHLF parametrization (Gillett and King, 2003)	Utterance-level	Patterson (Patterson, 2000)	–	Piecewise linear
OSV parametrization (Ceyssens et al., 2002)	Utterance-level	Offset, slope and variance	–	Linear

Figure 2.2: An overview of pitch mapping methods

We can model the pitch at different acoustic and linguistic levels (frame-level, syllable-level, and phrase-level). Various pitch representations have been used, such as the F0 contour, the discrete cosine transform (DCT) of the F0 contour, the Wavelet transformation of the F0 contour, and other compact parameterisations of the F0 contour.

2.4.1 Current challenges

The main challenges faced by researchers working in voice transformation are:

1. **Analysis/Synthesis issues** - converting to/from waveform to speech features is where much of the audio quality is lost. New vocoders are being developed to tackle this.
2. **Feature interpolation issues** - when representing spectral envelopes, various features are used, such as LSF and MFCC. Interpolating between two or more spectral representations can result in spectral representations that are not generated by the human vocal tract, thus producing unnatural sound.
3. **One-to-many issues** - when two very similar speech segments of the source audio correspond to very dissimilar target audio segments, the mapping function often over-smoothes the generated features.
4. **Prosodic mapping issues** - previous models often performed naive modifications of global statistics, such as average and standard deviation, which cannot effectively convert suprasegmental features. These models suffer from the absence of certain high-level features during conversion, which hugely affect human prosody. These features might be linguistic features (such as information about phonemes and syllables), or more abstract features (such as sarcasm and emotion). Therefore the

main challenge is to transform pitch contours by considering more context than one frame at a time Mohammadi and Kain [2017].

2.5 Active research areas

There are many new avenues and techniques that researchers are exploring, with a view to improving the quality of voice transformation. Here we present a selection of these with a brief description and references.

2.5.1 Pitch representation

It has been shown that outputting quantised F0 levels rather than interpolated continuous valued F0 contours avoids artificially interpolated F0 contours [Wang et al., 2017].

F0 segments can be represented with discrete cosine transform (DCT) coefficients at the syllable level, and multi-level dynamic features can be added to model the temporal correlation between syllables and to constrain the F0 contour at the phrase level [Veaux and Rodet, 2011].

A continuous wavelet transform (CWT) can be used to divide a continuous-time function into wavelets. Unlike Fourier transform, the continuous wavelet transform possesses the ability to construct a time-frequency representation of a signal that offers very good time and frequency localization. CWT decompositions of F0 and energy contours describe prosody patterns in different temporal scales and allow for effective prosody manipulation in speech synthesis.

As the F0 in an emotional voice changes over time here the continuous wavelet transform (CWT) is used to decompose F0 into different temporal scales. These can be well trained by NNs for prosody modeling in emotional voice conversion [Luo et al., 2017].

Here, phonetically-aware sparse representations of the fundamental frequency and energy contour are created using CWT. It is proposed that phonetically aware exemplars lead to a better estimation of the activation matrix, and possibly better conversion of prosody [Sisman et al., 2017].

2.5.2 Non-parallel unaligned data

Building a voice transformation system from non-parallel speech corpora is highly valuable in real application scenarios. In many speech datasets we do not have the source and target speech are not uttering the same phrases. Being able to simply sample a person's speech and transform across their entire range of emotional expression would be extremely useful. Here, a variational autoencoding GAN is used so that non-parallel

(unaligned) voice datasets can be used to train the network, also with improved conversion quality [Hsu et al., 2017].

A restricted Boltzmann machine (RBM) can be used for training with non-parallel data, and used to represent the distribution of the spectral features derived from a target speaker. linear transformation was employed to convert the spectral and delta features. A conversion function was obtained by maximizing the conditional probability density function with respect to the target RBM [Lee, 2017].

An average voice model is trained using other speakers' data, and i-vectors, a compact vector representing the identities of source and target speakers, are extracted independently [Sisman et al., 2017, Wu et al., 2016].

2.5.3 Mapping functions

Mapping functions are the architectures and methods used to model the relationship between the representational features of the source and target audio sequences.

Recurrent neural networks and long-short-term memory (RNN and LSTM)

Recurrent neural networks and long-short-term memory cell models are able to exploit the temporal dependencies in audio data much more effectively than deep neural networks. Deep bi-directional LSTM models have been created to model timbre and prosody [Ming et al., 2016], and also pitch transformation [Fan et al., 2014, Li et al., 2016]. LSTM-RNN-based models have been shown to produce more natural sounding voice synthesis, and perform the conversion at lower-latency, than HMM models [Zen, 2015, Zen and Sak, 2015].

LSTM-RNN-based SPSS systems can be optimised to improve upon HMM-based SPSS systems in speed, latency, disk footprint, and naturalness on modern mobile devices. This is achieved through quantising the LSTM-RNN weights to 8-bit integers to reduce disk footprint by 70%, using multi-frame inference to reduced CPU use by 40%, and using an e-contaminated Gaussian loss function rather than a squared loss function to increase the learning rate and improve naturalness [Zen et al., 2016].

It is possible to improve the conventional LSTM-RNN method for voice conversion by modelling the spectral (frequencies) and the temporal (pitch contour) features together in a two-dimensional time-frequency LSTM (TFLSTM). A structured output layer (SOL) can then be used to capture dependencies between the two [Li et al., 2017].

By modifying the RNN architecture to include multi-tier links, feedback loops at both the frame, syllable and phoneme levels, we can preserve the segmental and suprasegmental correlations between F0 contours and the unvoiced regions. Using dropout on the feedback F0 avoids exposure bias i.e. model is less dependent on the feedback data, and

uses the textual input more, which makes it more robust to generation errors on previous frames [Wang et al., 2017].

A standard RNN can be extended such that not only is the hidden state recurrent, but also the output values are recurrent i.e. we take into account previous output values on subsequent time steps [Zen and Sak, 2015].

A template-based approach can be used for automatic F0 generation, where per-syllable pitch contour templates are predicted by a recurrent neural network (RNN). This mitigates the over-smoothing problem and is able to reproduce pitch patterns observed in the data. The use of an RNN, paired with connectionist temporal classification (CTC), enables the prediction of structure in the pitch contour spanning the entire utterance. Separate LSTMs are used for predicting phone durations and the other acoustic features [Ronanki et al., 2016].

Sequence to sequence transcoders (seq2seq)

Sequence to Sequence (seq2seq) transcoder models use a multilayered Long Short-Term Memory (LSTM) to map an input sequence to a vector of a fixed dimensionality, and then another deep LSTM to decode the target sequence from this vector. After training the two networks with source and target sequences, we can perform conversion operations on new source sequences [Thang Luong, Eugene Brevdo,].

Seq2seq can be used to convert from sequences of text, and was originally designed for the purposes of machine translation [Sutskever et al., 2014]. It can be used to convert neutral speech to emotional speech with controlled intensity of emotions by converting sequences of F0 values.

An LSTM-based autoencoder sequence-to-sequence text-to-speech (STT) model has been shown to approach WaveNet in terms of quality while retaining low computational cost and latency [Van den oord et al., 2016, Wan et al., 2017]. At the time of writing, no paper that demonstrated a sequence-to-sequence being applied to the task of voice transformation could be found.

Generative adversarial networks (GAN)

A training framework for sequence-to-sequence voice conversion (SVC) has been developed that uses a similarity metric implicitly derived from a generative adversarial network (GAN). This helps ensure the acoustic-feature sequences generated from a converter are not over-smoothed, avoiding the buzzy-sounding speech commonly found in such networks [Kaneko et al., 2017]. A similar system uses a GAN without a sequence-to-sequence model to overcome the smoothing problems. It consists of two neural networks: a discriminator to distinguish natural and generated samples, and a generator to deceive the discriminator [Saito et al., 2018].

Chapter 3

Contributions

3.1 Problem statement

Given the above research, and in discussions with my project supervisor Prof. Nicolas Obin at IRCAM, a voice transformation system capable of converting neutral speech to emotive speech was built using a machine learning sequence-to sequence-architecture. Specifically, this is a transcoder that can learn the mapping between two sequences of pitch values, one from neutral speech and other from speech expressing a single emotion e.g., anger, sadness, joy, or fear. The model was built using Python and the Tensorflow machine learning framework, which facilitated development alongside the experienced developers at IRCAM, and the subsequent sharing of results and code amongst the team.

3.2 Data

3.2.1 The raw dataset

The model was trained on a portion of a parallel audio database of high-fidelity studio recordings, Olivia2006, developed in-house at IRCAM. The recordings are in raw 16-bit WAV format, at 48000 KHz.

The portion of the dataset used for training our models is made up as follows.

- One female actor, speaking only in French i.e. a language dependent model.
- 10 different phrases (p01-p10) spoken
- 8 different emotions (e01-e08) spoken for each phrase
  - JOY - Gentle
  - JOY - Explosive
  - FEAR - Contained
  - FEAR - Hysterical
  - SADNESS - Contained

  - SADNESS - Tearful
  - ANGER - Controlled
  - ANGER - Explosive

- 5 levels of increasing intensity (i01-i05), along with a 'neutral' level (i00), spoken for each emotion
- In total, this provides:
  - 80 recorded utterances of neutral speech (10*8*1), used as source data
  - 400 recorded utterances of expressive speech (10*8*5), used as target data

3.2.2 F0 extraction and alignment

AudioSculpt was used to extract the fundamental frequency (F0) over the entire phrase, and output as a MATLAB MAT file. The pitch values extracted from the waveform are continuous sequences of real values corresponding to the voiced sections of speech, separated by sections of silence corresponding to the noise (unvoiced areas) in the waveform. It was necessary to interpolate over these breaks in the contour in order to create an unbroken F0 signal which is then later divided by syllable or phoneme for use as input the model.

We used a a text-dependent approach, as we had a parallel dataset that was created expressly for the purpose of voice transformation by actors reading from scripts. That is to say we exploited both the original text of what is being said, and the audio of the phrases being spoken. In order to train the sequence-to-sequence model on a phoneme or syllable level, we needed to be able to segment the audio waveform into phonemes and syllables. To do this we made use of the transcription text and in-house tools developed at IRCAM, such as IrcamAlign [IRCAM, 2018b], AudioSculpt [IRCAM, 2018a] and WaveSurfer. In particular, IrcamAlign performed text-to-audio alignment which provided linguistic information about the text being pronounced. The sequence of words, syllables and phonemes, and crucially, the timestamp position of each of these elements within the sentence was extracted and added to the MAT file.

Spectrograms were also generated for manual data validation and inspection of the differences in F0 between utterances of the same phrase in different emotions/intensities. AudioSculpt can also superimpose the phoneme labels onto the relevant sections of the spectrogram, labelling the features and permitting further analyses (Figure 3.1). Note that the full spectrum data was not used as input to the model, only the F0 extracted from it.

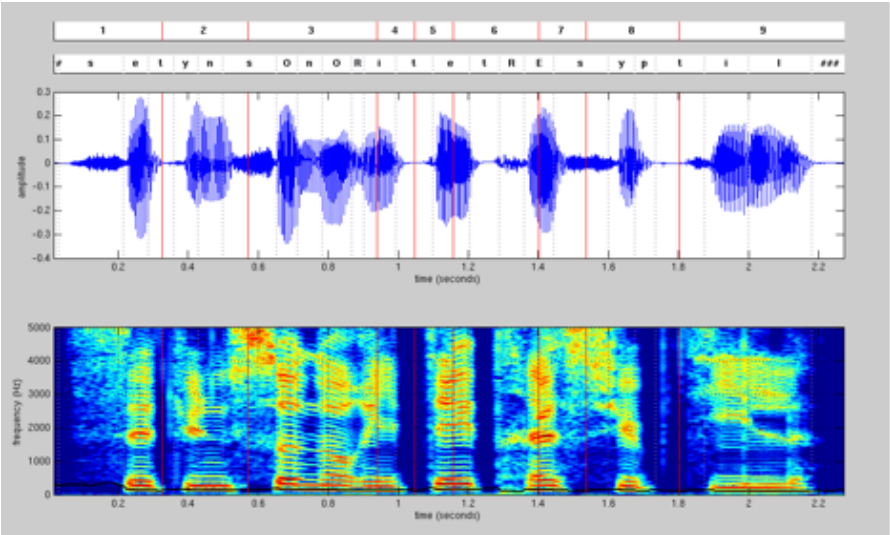

Figure 3.1: Spectrogram with aligned phonemes using AudioSculpt 3.0 and IrcamAlign

3.3 Methods

3.3.1 Conversion process overview

The voice emotion conversion process has three main steps: an extraction of the F0 contours from the neutral source WAV file; a transformation of those contours using the seq2seq model; an application of the transformed F0 contours back into the neutral source WAV file, which produces a new WAV file containing the desired expressive form of the utterance (Figure 3.2).

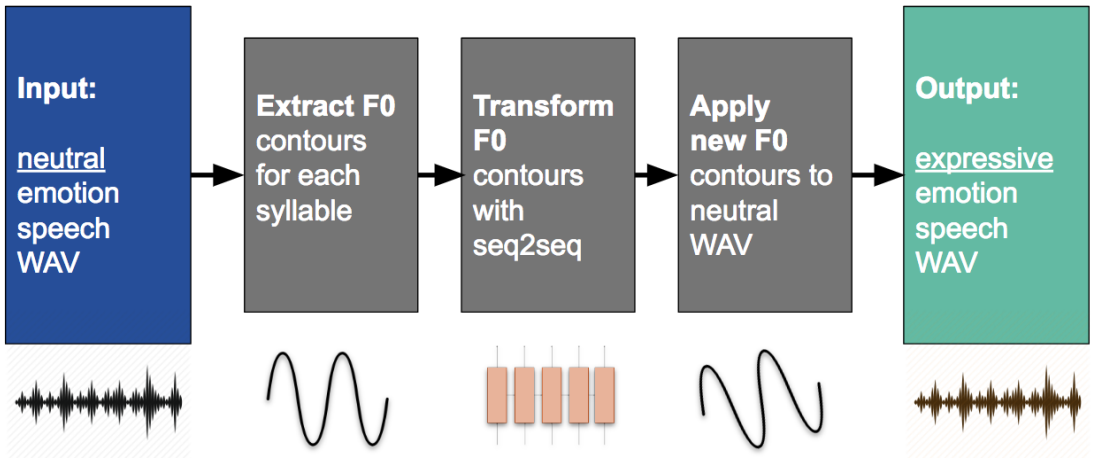

Figure 3.2: Overview of the WAV conversion process from neutral to expressive

Before this conversion can take place, the training data must be prepared and the model trained to learn the mapping.

3.3.2 Training data pre-processing: extraction of F0 contours

The F0 values were extracted as real-valued floats, however the seq2seq framework code that we adapted [Google, 2017] uses an embedding layer, as it was originally designed to perform machine translation of words. Therefore, a conversion from an infinite range of floats to a finite range of discrete integers was required. This was accomplished through simple rounding to the nearest integer, resulting in a slight loss of information in the process.

Using the timestamps in the MAT files, the continuous F0 sequences were split by syllable or by phoneme, and used to produce a series of CSV files, each containing the string of integers representing the F0 for a particular phrase/emotion/(syllable/phoneme) combination. These files were renamed to include the phrase, emotion, intensity and syllable/phoneme label, as well as the relative phoneme position in the phrase.

The CSV files were then assembled into collections of source/target pairs to build the training set. The i00 files were always used as the source data. Various combinations of target intensities were used, such as i01-i03, i03-i05, and i01-i05, as the database was produced by an actor, and the there was concern that the higher intensities suffered from 'over acting', with too much emotion being imparted into the utterance.

3.3.3 Sequence-to-sequence (seq2seq) and attention

An autoencoder is a variant of a neural network in which the output is the approximation of the input data. The network is designed so as to learn the low-dimensional latent representation of the data that best reconstructs the observed data. A transcoder is the same at this, except that the objective is no longer to approximate the input vector, but another data vector. For speech emotion conversion, the input and output vectors are sequences of pitch values calculated on each syllable/phoneme.

Also, by using recurrent architectures we can model the dynamics of long series of sequential data, and handle input and output vectors of variable lengths. Converting between source and target F0 sequences requires a transcoder capable of generating sequences of varying lengths, as the duration of syllables and phonemes varies according to the emotion and intensity being expressed, so the corresponding pitch sequences will have a different number of pitch values.

Furthermore, our database is very small in relation to the huge databases used by companies such as Google, often containing millions of utterances. so this influenced our selection of transformation method. LSTM cells favour memorisation of long distance correlations in sequences. They allow us to take full advantage of the information

contained in the sequences, and to produce a model that generalises as well as possible, given the small number of training examples.

It is for these reasons primarily that we chose an RNN-LSTM sequence-to-sequence encoder-decoder architecture to learn the mapping between the neutral and emotional pitch sequences.

Basic sequence-to-sequence (seq2seq) architecture

In the basic encoder-decoder architecture, the system reads the source sequence using an encoder, builds a vector, passes that to a decoder, which then processes it and generates a converted sequence.

In the general case, the input and output sequences are not the same length. It is important that the entire input sequence be read before the target can start to be predicted, in order for the long-term dependencies to be taken into account. The internal state that the RNN-LSTM encoder layer outputs is the context (the conditioning) of the decoder layer. Note that in the basic architecture, the output of the encoder is discarded, and only the internal state is considered useful. The decoder is trained to understand the general model of the target data, but requires an input in order to generate something. As an analogy, not giving the decoder a context would be like asking a human translator (who understands the general model of the target language) to perform a translation without giving him/her the source sentence to translate.

The decoder predicts the next element of a target sequence given the previous element of the target sequence. This is called teacher forcing. During training, the decoder is trained to turn a target sequence into the same sequence, offset by 1 timestep. The decoder learns to generate target(t+1) given target(t), conditioned on the input sequence (so it knows what to generate). In Figure 3.3 we use a written phrase (a sequence of characters), but the sequence could just as easily be a series of numbers representing the values of an F0 contour.

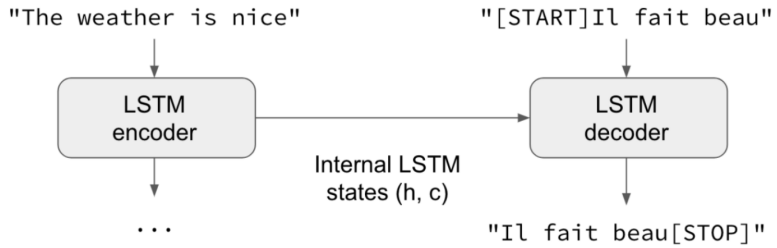

Figure 3.3: The encoder-decoder model

When we perform inference (conversion), we use the encoder to encode input sequence into hidden state vectors. In Figure 3.4 we first we start with target sequence of length

1 (just the start of sequence character, 'SOS'), and provide the decoder with the state vectors and the 1 char target sequence. It calculates prediction probabilities for all possible next characters/values, and we use argmax to select (sample) the next character (i.e. the first character/word in the translated target sequence/sentence), and append to the target sequence. If we were using a quantised F0 model the process would be the same. If using a continuous F0 model, we would engineer the deep LSTM-RNN network to simply output continuous values. We then (optionally) feed the output value/character back into the decoder in order to generate the next character in the sequence (teacher forcing), then repeat until the decoder generates the end of sequence character 'EOS'.

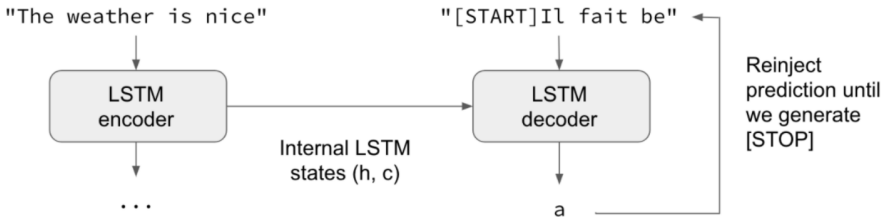

Figure 3.4: Inference of a target sequence

In our architecture, the 'EOS' character was automatically applied to the end of all source and target sequences that were fed into the model for training and inference.

Attention mechanism

Attention mechanisms are a way of augmenting the performance of the basic model presented above, and are the defacto standard in industry today (Thang Luong Eugene Brevdo n.d.). As illustrated in Figure 3.5, instead of discarding all of the hidden states computed in the encoder RNN, the attention mechanism allows the decoder to peek at them. This allows longer sentences to be handled, as it avoids the bottleneck of a fixed-size hidden state as input to the decoder.

3.3.4 Transformation of F0 contours with seq2seq

Software environment

The data pre/post-processing was all written in Python in Jupyter Notebooks, leveraging various open source libraries such as SciPy [sci] and Matplotlib [mat]. A library developed at IRCAM, as_pysrc was used for phase vocoder conversions of the STFT. This leverages an implementation of Griffin and Lim's algorithm [Griffin and Lim, 1984] to recover an audio signal given only the magnitude of its STFT.

The model was built using the Python programming language, as a fork of the seq2seq framework [Google, 2017]. This was one of the first sequence to sequence frameworks developed by Denny Britz at Google in 2017, and has already been superseded by more

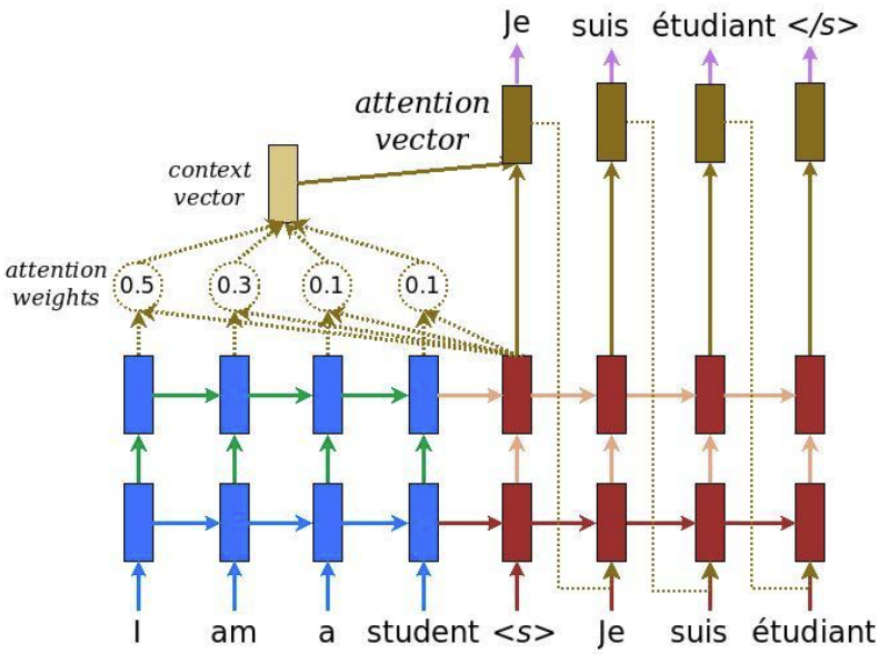

Figure 3.5: The attention mechanism, as applied to the seq2seq architecture

complete frameworks such as tensor2tensor [Tensorflow]. It was originally developed for machine translation purposes, mapping between relatively short sequences of words, so I adapted it for use with longer sequences of integer F0 values.

The final phoneme-to-phoneme model used for the experimental evaluation uses Tensorflow 1.8 libraries for the machine learning calculations. It took around 90 minutes to train 10000 steps on 1 x GeForce GTX 1080 Ti graphics card with a batch size of 256. A model such as this needed to be trained for each combination of model type, target emotion, and intensity range. As such models were often trained in parallel on separate GPUs to save time.

Encoder configuration

We made a number of modifications to the basic architecture in order to achieve the best results. The encoder was configured with two layers, each having 128 bidirectional LSTM cells to ensure the full F0 contours of each phoneme could be captured (we need one cell per F0 value). Bidirectional cells could be used as the full source and target data was available immediately, which would not be the case in a real-time system. Dropout on the input values was set at 0.8 to improve generalisation. Residual connections were not used in the encoder, in order to avoid the raw source input values being passed directly to the second layer, where the attention mechanism would reference them directly, spoiling the results. Peepholes in the LSTM cells were tried, but were found to also degrade the

results significantly.

Decoder configuration

The decoder was configured with three layers, each having 128 bidirectional LSTM cells. Dropout on the input values was again set at 0.8 to improve generalisation. Unlike the encoder, residual connections were used in the decoder, and were found to provide a slight improvement. Peepholes were not used. A cap on the maximum decoded sequence length was set to 128, to ensure the decoder did not generate unnaturally long F0 contours for the phonemes.

Attention mechanism

In my model I actually only used the attention mechanism [Bahdanau et al., 2014] to transfer context from the encoder to the decoder. Unlike the basic seq2seq architecture, after the entire source sequence had been read, I did not pass a context vector across, but instead, triggered the decoder to begin reading in the target sequence (during training) and referencing the (previously saved) outputs from each encoder step.

For longer sequences this was found to improve the results, which makes sense as it is unrealistic to expect a single context vector to contain sufficient information for every step of the encoding. This is especially the case for the steps that occur at the beginning of the sequence, the information from which is often greatly diminished or lost entirely by the time the final source elements have been read in.

Loss function and optimisation function

Cross entropy loss (log loss) indicates the distance between what the model believes the output distribution should be, and what the original distribution really is. It is defined in Equation 3.1.

$$H(y,p) = -\sum_i p_i \log(q_i) \tag{3.1}$$

where $p \in \{y, 1-y\}$ and $q \in \{\hat{y}, 1-\hat{y}\}$. The cost function is computed by taking the average of all the cross entropies over the whole sample.

The cross entropy loss is a widely used alternative to the squared error loss (RMSE). It is used when node activations can be understood as representing the probability that each hypothesis might be true, i.e. when the output is a probability distribution. Thus it is used as a loss function in neural networks which have softmax activations in the output layer. Our model has a softmax as the final layer, which outputs a probability for each element in the target embedding (each a unique F0 integer value, restricted to the range 50 to 550, easily encompassing the vocal range of a human).

Minimising cross entropy maximises the log likelihood. The ADAM optimiser was employed for this purpose, using a default epsilon of 0.00000001 and an initial learning rate of 0.001. This minimised the cross entropy over the target embedding of F0 integers.

It must be noted that the cross entropy loss value isn't indicative of anything other than how training is progressing; it's possible to have a low cross entropy value but the F0 values between neutral and expressive can be way off, and equally possible to have a high cross entropy value and still get good F0 results. Ideally, during training we would periodically generate predictions for the target F0, then calculate the true F0 value loss (difference) to gauge if the training is making a real difference to the quality of the F0 transformation. In our model we did not attempt this, as manual inspection of the final contours showed a near perfect matching was generated each time. The problem we faced, therefore, was not so much to generate accurate target F0 contours, but ensuring these contours would generalise well to unseen contexts (words/phrases).

3.3.5 Dataset configurations

A number of dataset configurations were tried over the course of development, in order to achieve the best results.

Syllable and phoneme level data

To start with, a basic model trained at the phrase level was built. However the F0 sequences were too long, due to the small frame size used during the Short Time Fourier Transformation (STFT) that AudioSculpt uses to extract the F0 values. This in turn increased the number of LSTM cells required, and caused the model to take too long to train. We quickly moved to using the extracted alignments to train at the syllable and phoneme levels, which involved much more reasonable sized F0 sequences of 5 to 100 elements each, and made training much quicker and more manageable.

Voiced phonemes only

When training at the syllable to syllable level, the results sounded poor as the model was being trained on F0 contours that included the unvoiced parts of a syllable. For example the words 'soldat' has two syllables, 'sol' and 'dat'. The first syllable 'sol' has three phonemes, the unvoiced phoneme 's' which is simply a hissing noise signal, and two voiced phonemes, the 'o' vowel sound, and the 'l' sound. The values interpolated across the 's' phoneme would (quite literally) introduce noise into the model. To avoid this, the data was split by phoneme as explained above, and the model trained on single voiced phonemes only.

Matching syllable/phoneme position in the phrase

There are often a number of identical syllables or phonemes in a phrase, and many more across the whole dataset. Initially, we produced source/target combinations of every

syllable with every other identical syllable across all target phrases, emotions and intensities. The reasoning behind this was to produce as large a number of identical syllable combinations as possible, to compensate for the small dataset size. However, this approach was flawed for a crucial reason; the context that each syllable appeared in was often completely different, as the source and target syllables, while the same syllable, came from different words in different phrases. This was an entirely unrealistic mapping that produced resulted in our model producing very poor predictions.

Then this technique was restricted to only syllables or phonemes in the same phrase, such that if a phrase contained two identical phonemes at different points, there would be 2*2=4 total source/target combinations. However, even this was unrealistic, and the results suffered.

Finally, to preserve the true context, the syllable and phoneme F0 files were tagged with their position in the phrase, and only files from the same position in the phrase were matched as source/target combinations. Despite the fact this made the training dataset much smaller, due to their being far fewer possible combinations, this adjustment alone led to a dramatic improvement in the quality of the resulting conversions.

Conditioning based on syllable position in the phrase

To add further context to the model, the F0 values for a phoneme were tagged with a prefix that indicated whether they belonged to the first, last or other syllable in the phrase. This caused the model to distinguish between these three types of contour, which allowed it to generate the important inflections often found at the start and the end of a phrase. This adjustment also made a significant difference to the quality of the results. This difference is quantified and analysed in the results section of this report.

This idea was further extended to conditioning at the sub-phrase level, where the syllables before and after each silence pause in the middle of the phrase were also tagged with the last/first prefixes. However, this had a negative effect on the results, with the resulting conversions sounding very unnatural.

Train / validation / test split

The complete dataset of source/target pairs of phoneme contours was first split by phrase, to ensure that the phrases used for training were not used for testing. Of the 10 phrases available, 6 were used for training, and 4 for testing i.e. a 60% training data / 40% test data split.

The training data was then split again, 85% for the training set / 15% for the validation set. This resulted in around 1100 phoneme contour pair CSV files used for training, and 170 used for validation.

3.3.6 Inference, data post-processing and WAV conversion

Inference and visualisation

Once the model had been trained for a particular emotion, it was used to produce the converted phoneme contours for the neutral test set phrases, for use in their conversion. Recall that only the model was only trained to transform voiced phonemes, and so only voiced phonemes can be supplied at inference time. Assembling the voiced phoneme F0 contours for a particular phrase creates a fragmented contour interspersed with gaps for the silences and unvoiced sections. These were visualised using Matplotlib, and inspected alongside the neutral contours for discontinuities and irregularities (Figure 3.6).

In Figure 3.6, the solid lines are the original neutral contours, while the dotted lines are the transformed expressive contours (in this case, joy). Like phonemes have the same colour, while black lines are unvoiced phonemes or periods of silence. Notice that the transformed contours are of a different length to their original counterparts, as desired, which is thanks to the seq2seq architecture being generative in nature.

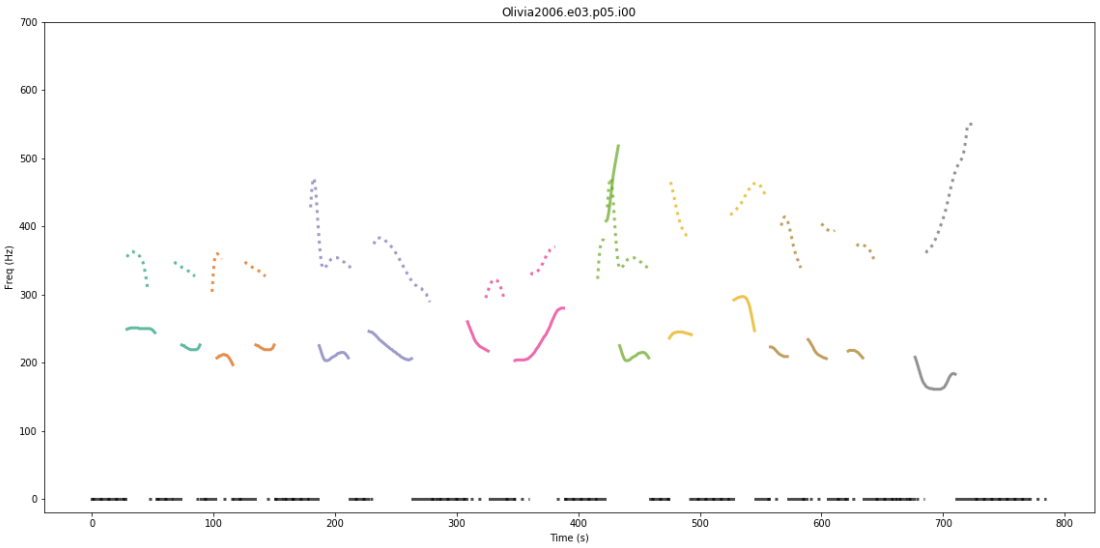

Figure 3.6: F0 contours voiced phonemes across an entire phrase.

Time stretching the source WAV

Each of these transformed phoneme contours is of a non-determinable length, as the decoder generates sequences of variable length. This is desirable, as a vowel phoneme expressed in a 'sad' manner may be longer in duration than the same phoneme expressed neutrally. In order to be able to apply the transformed contours to the neutral WAV file, each corresponding section of the neutral WAV file must be stretched to fit their new lengths. This was performed using the as_pysrc library, using the start and end

timestamps of the neutral phoneme contours, and calculating the new timestamps by accumulating the durations of the transformed voice contours with the durations of the original silences and unvoiced phonemes.

Interpolation across phoneme contours

We then linearly interpolated the fragmented phrase F0 contours over the silence and unvoiced sections, to eliminate the gaps. This was important, as we could then use the harmonicity of the neutral WAV to determine which parts of this F0 contour to actually use during the STFT conversion by the phase vocoder.

Harmonicity threshold

Harmonicity is the signal to noise ratio at a particular point in a sound file, and it changes throughout the file. The harmonicity ratio is high for areas of the file where the signal is strong and the noise is weak, such as in the voiced sections of speech. These are the areas that we wish to apply the interpolated F0 contour to, whereas we wish to avoid applying it to the low harmonicity areas, corresponding to high noise, silence, and crucially, the unvoiced phonemes.

Various values for the harmonicity threshold were used during the transformation, 0.3, 0.5 and 0.7. The correct level depended on the source phrase and target emotion, but in general a high value of 0.7 produced an output with the least audible distortion. A high level is the most conservative, applying the least amount of the interpolated F0 contour to the neutral file, and only to the sections most likely to be voiced.

Chapter 4

Experimental evaluation

To evaluate the results of the conversions, we considered quantitative and qualitative methods.

4.1 Quantitative evaluation

As there are multiple target F0 contours used for source combination of phrase/emotion/intensity/phonem there is no single ground-truth to compare generated contours to.

Additionally, the generated F0 contours are of variable length, making standard comparison methods such as root-mean-squared-error (RMSE) impossible to use. Other distance measures such as dynamic-time warping were considered, but the lack of a single ground truth would mean any distance measurement would be an unreliable indicator of success. Quantitative evaluation was therefore not attempted.

4.2 Qualitative evaluation

It was decided to perform a qualitative evaluation using an emotion recognition survey, to determine how easily people could recognise the emotions added to the neutral test WAV files. The assumption is that if the emotion is easily recognisable, then the conversion is successful.

4.2.1 Model and data

To generate the best sounding results for the evaluation, the phoneme-to-phoneme model with the above mentioned modifications was used. The data was taken from 4 of the 8 available emotions (e02, e04, e06, e08) corresponding to the strong form of each of the 4 main emotions, as these were found to result in transformations with more distinct expressivity. Also, the model was trained using different intensity ranges for each emotion; i01-i03, i03-i05, and i01-i05, to compare the results. The F0 conversions were applied using a harmonicity value of 0.7, and the best sounding results were selected from across

all those generated. These files were then normalised to all have the same volume level, and converted to 160 Kbps MP3 format for use online.

Three sets of 32 files were manually selected for: 32 from the syllable-position conditioned model; 32 from the non-conditioned model; and 32 from the original non-transformed expressive samples provided by the actor (for use as a control). Each set of 32 files comprised 8 samples from each of the 4 emotions. Additionally, care was taken to ensure each emotion set contained 2 examples for each of the 4 test phrases (p01, p03, p05, p10), to ensure they were represented equally and to reduce the bias from the wording of the samples influencing the participant's choice of emotion. A total of 96 files were available for evaluation by participants.

4.2.2 Online survey

IRCAM have developed an online evaluation framework for receiving feedback on the results of audio experiments. This is a survey tool with audio playback facilities, which allows anyone with internet access to listen to the audio clips and provide responses to evaluation questions. Using this tool, a custom online survey was constructed (Figure 4.1), which allowed participants to hear a voice sample, then select an emotion from four options (happiness, sadness, anger, fear). Additionally, they would judge how natural the voice sounded, and select a corresponding voice quality (bad, poor, fair, good, excellent).

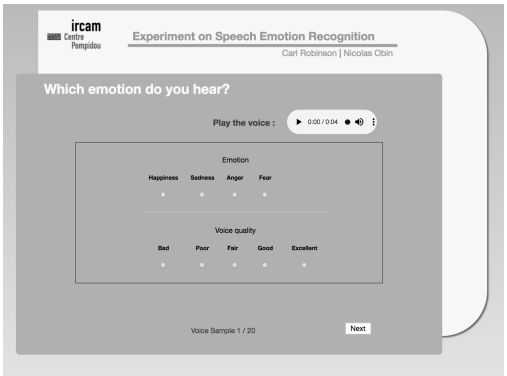

Figure 4.1: Participants selected one of four possible emotions: happiness; sadness; anger; fear.

The survey asks participants to identify the emotion for 20 files, selected at random for each participant from the pool of 96 files. At the end of the survey, the participants indicate their level of French (native, non-native, cannot speak French), and the equipment they used for the evaluation (headphones, earphones, speakers). The survey link was then distributed widely amongst fellow researchers at IRCAM and fellow masters degree students, as well as the wider research community using mailing lists and social media channels.

4.3 Results

The survey was completed by **87 participants**, who provided a sizeable **1734 responses** to each of the questions. Here we present column charts that represent the confusion tables for ease of comparison. We split the response data across a number factors: the

model type, the participant's french level, equipment used for the exercise, and the phrase of the sample audio.

4.3.1 Model type

Here we divide the responses by model type, to compare and contrast the effect of a model conditioned on the position of the syllable in the phrase, versus one with no conditioning. We also include the responses to the original (non-transformed) expressive samples, which serve as a baseline for the respondents' ability to identify emotion in the human voice.

We can see that the conditioned model performs best for anger and fear, whereas the non-conditioned model is better for joy and sadness. This is interesting as anger and fear are opposite emotions on Plutchik's Wheel of Emotions [Plutchik, 2001], as are Joy and Sadness, so it may be the case that the pairs share similar features.

4.3.2 Conditional model

The conditional model successfully transforms neutral samples into ones expressing anger, with over 55% of samples correctly identified as angry. Fear and sadness are easily confused for one another, but rarely confused with joy or anger. Finally, joy is the worst performing emotion, with participants identifying these samples as joy, fear and sadness in equal measure (Figure 4.2).

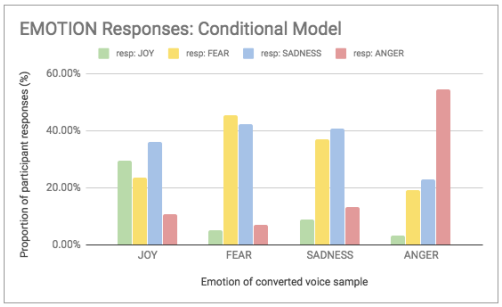

Figure 4.2: Emotion responses for the phrase conditional model

4.3.3 Non-conditional model

The non-conditional model somewhat successfully transforms neutral samples into ones expressing joy and sadness; joy is correctly identified 38% of the time, but is confused with sadness just as much. Sadness is correctly identified in 48% of responses, and confused with fear 32% of the time (Figure 4.3).

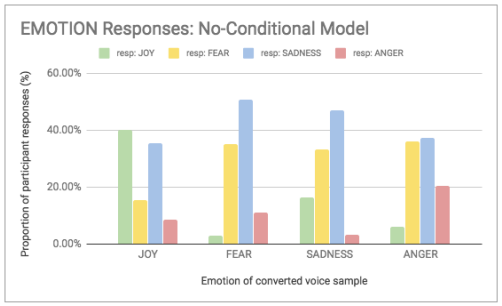

Figure 4.3: Emotion responses for the non-conditional model

4.3.4 Original non-transformed samples

The responses to the original (non-transformed) expressive samples are quite

telling; joy and anger have near perfect scores, but fear and sadness suffer from problems.

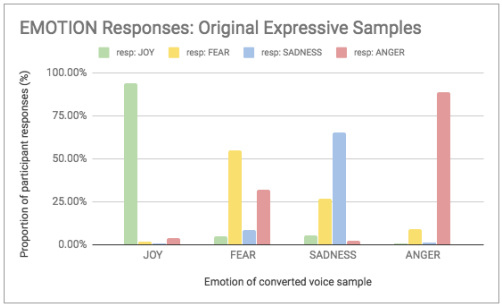

Figure 4.4: Emotion responses for the original (non-transformed) expressive samples

As shown in Figure 4.4, fear is interpreted as anger 33% of the time, and listening to the samples you can hear why. During the research and development phase, we took the decision to only use the strong form of each of the four emotions for the evaluation experiment, to reduce the number of models that needed training (to save time), and because we were getting more convincing test conversions with the strong versions. This means that hysterical fear was used in the experiment, but contained fear was not. Hysterical fear sounds more like shouting during an emergency, whereas restrained fear is the weak and trembling type of fear that is more commonly associated with the label. It is likely that users were expected to hear contained fear, and as there is a loss of nuance in the voice quality due to the limitations of the model, causing the hysterical fear to lose its 'fearful' characteristics, the results were interpreted as anger.

Sadness is interpreted as fear 26% of the time (Figure 4.4). While less serious, this serves to highlight that humans often have a hard time identifying the emotion others' voices, even when that person is an actor expressly trying to convey a single emotion in their voice. Emotions are ambiguous, and their interpretation depends on many factors, including the speaker, the listener, the context, and the linguistic content of the utterance.

4.3.5 Difference between transformed and original

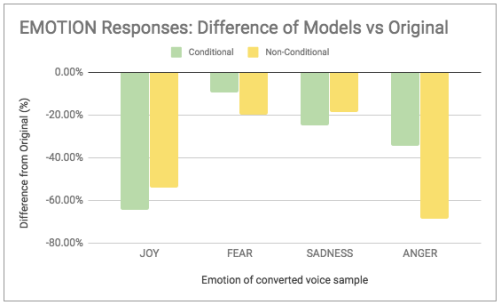

Figure 4.5: Difference in emotion responses between the original (non-transformed) expressive samples and the conditional and non-conditional models

It's important to calculate the difference in recognition rates between the the original samples and the transformed files, for each of the four emotions, so that we can take into account the participants' abilities to discern each emotion.

It can be seen that both models perform equally poorly for joy, each having around 60% fewer correct responses than the original samples (Figure 4.5). This suggests that the model poorly represents the features responsible for expressing joy in the voice.

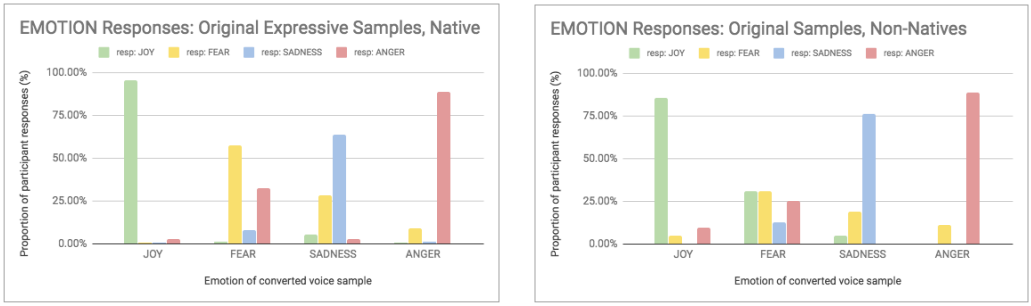

Figure 4.7: Native and non-native french speakers' emotion responses for original samples

Anger is the most easily recognisable emotion out of the four original samples, so it is a high bar to meet. The conditional model achieves good results, within 30% of the baseline, which is a reasonable result. However, it must be concluded that an effective expression of anger requires more than just a change in pitch, with voice quality, pacing and amplitude all playing significant factors.

It can be seen however, that anger is expressed twice as effectively with the conditional model as it is with the non-conditional model, suggesting that the conditioning of start and end syllables more successfully captures the pertinent features of the frequency changes in an angry voice.

Fear and sadness both perform equally well for both models, with recognition rates approaching parity with those from the original samples. This is a successful result, and may suggest that the expressions of these two emotions rely more heavily on the F0 than joy and anger.

4.3.6 Participant's French level

The large majority of participants were native French speakers (Figure 4.6), which was deliberately sought so as to avoid cultural bias in the results; expression of emotions can be culturally specific.

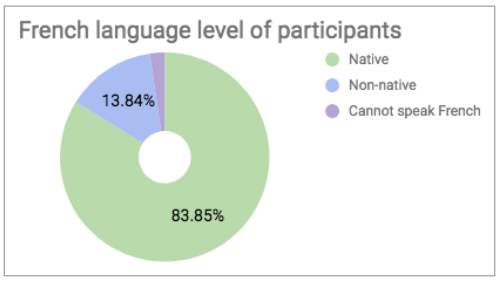

Figure 4.6: French language level of participants

Using the original non-transformed samples, as they are the least ambiguous, very little difference can be seen between native and non-native participants' ability to correctly identify the emotion (Figure 4.7). Note: here the non-native group includes participants who could not speak French at all, of which there were very few.

With the conditional model results, the non-native speakers were slightly better at distinguishing between fear and sadness, but this difference is too small to be considered significant. This is especially true given the relatively small sample size of non-natives (85 responses) compared to natives (494 responses). These findings are important, as they allowed us to continue our analysis of the results on the entire population, not just one of the native/non-native cohorts.

4.3.7 Equipment used for the exercise

Half of all participants used headphones when conducting the evaluation survey, while the other half were split equally between earphones and speakers (Figure 4.8). It is assumed that headphones and earphones provide a superior listening experience as they also provide some background noise isolation, however given that many responses are from people working in audio research, this may not be the case, as these people own high quality speakers and often arrange their listening environments for optimal sound fidelity.

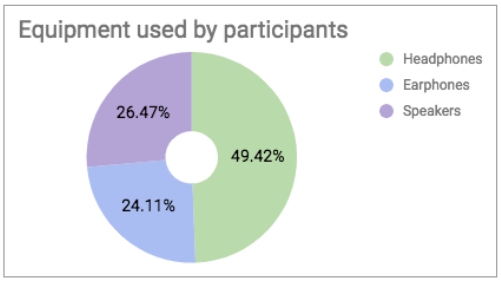

Figure 4.8: Listening equipment used by participants

With the original non-transformed samples, very little difference in the emotion detection performance (Figure 4.9) or the perceived quality of the voice (Figure 4.10) can be seen across listening devices. This suggests that the emotion in the actors voice is clear enough to be heard, regardless of whether they used headphones, earphones or speakers.

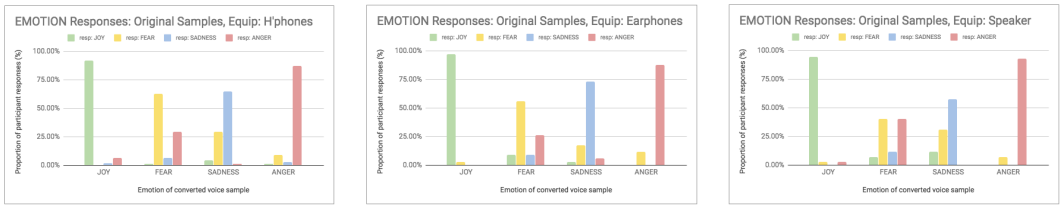

Figure 4.9: Emotion responses for original samples by participants using headphones, earphones and speakers to conduct the evaluation survey

Surprisingly then, a significant difference can be observed between fear and sadness on the conditional model results (Figures 4.11 and 4.12). For example, sadness is correctly identified the majority of the time with headphones and earphones, but is confused for fear over 60% of the time when speakers are used. This would suggest that a small change in the listening environment can significantly amplify the uncertainty between these often-confused emotions, by somehow attenuating the subtleties that distinguish them.

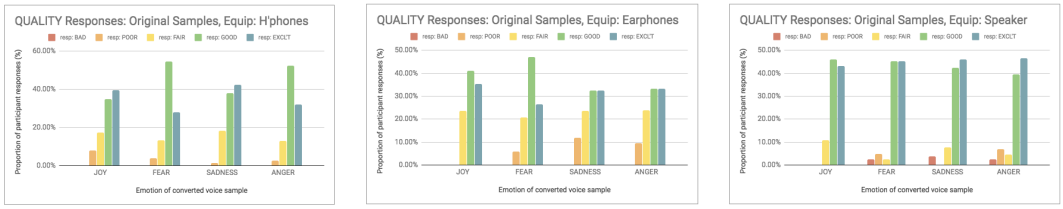

Figure 4.10: Quality responses for original samples by participants using headphones, earphones and speakers to conduct the evaluation survey

While joy and anger are impacted somewhat less by the choice of listening device, it can be concluded that headphones or earphones provide the best listening environment, and should be insisted upon for future experiments.

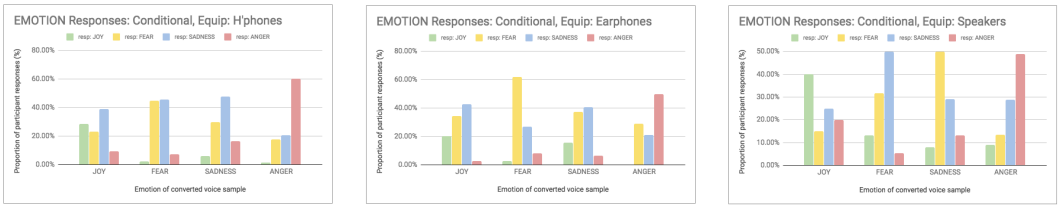

Figure 4.11: Emotion responses for conditional model transformations, by participants using headphones, earphones and speakers to conduct the evaluation survey

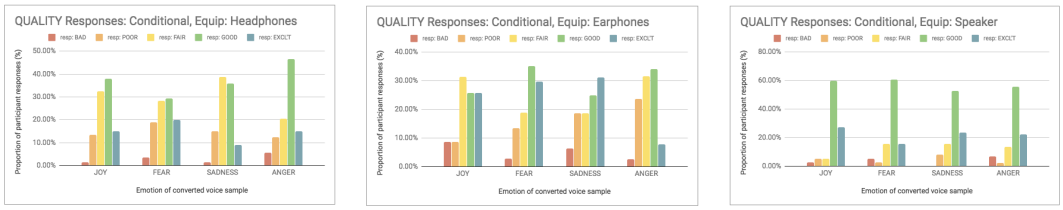

Figure 4.12: Quality responses for conditional model transformations by participants using headphones, earphones and speakers to conduct the evaluation survey

4.3.8 Phrase of sample audio

The results clearly show that the emotion selected by the participant is highly phrase dependent. If we compare the responses across the four phrases for the joy samples generated by the conditional model, we see that participants correctly identify joy over 40% of the time for phrases 1 and 5, over 20% of the time for phrase 3, and only 11% of the time for phrase 10 (Figure 4.13).

This could be due to a number of factors. Firstly, the linguistic content of the phrase could be influencing the listener. Phrase 1 is 'c'est un soldat à cheveux gris', which trans-

lates to 'it's a soldier with grey hair', while phrase 10 is 'je me demande, où se trouve cet endroit?', which translates to '"I wonder, where is this place?'. Neither phrase has anything particularly emotive in its content.

The real problem is that the original expressive samples of phrases 1 and 10 are read in different tones by the actor, with phrase 10 read in a more demanding forceful tone, while phrase 1 is spoken more neutrally. These characteristics will not entirely disappear after the F0 conversion process, which goes towards explaining the difference in the listeners' responses.

The solution to this problem is clearly a larger training database that contains many more phrases. Here we just use 6 phrases for training and 4 for testing, which introduces a large bias into the results. The training database could also be designed to be emotion dependent, to only include phrases that are appropriate for the target emotion i.e. there are things you would (almost) never say in a happy voice, so these could be discarded when training the model for joy.

The difference is much less for fear and sadness, with the conditional model almost reaching parity for fear. This is an encouraging result, as it shows that the model can generate realistic sounding results for both of these emotions.

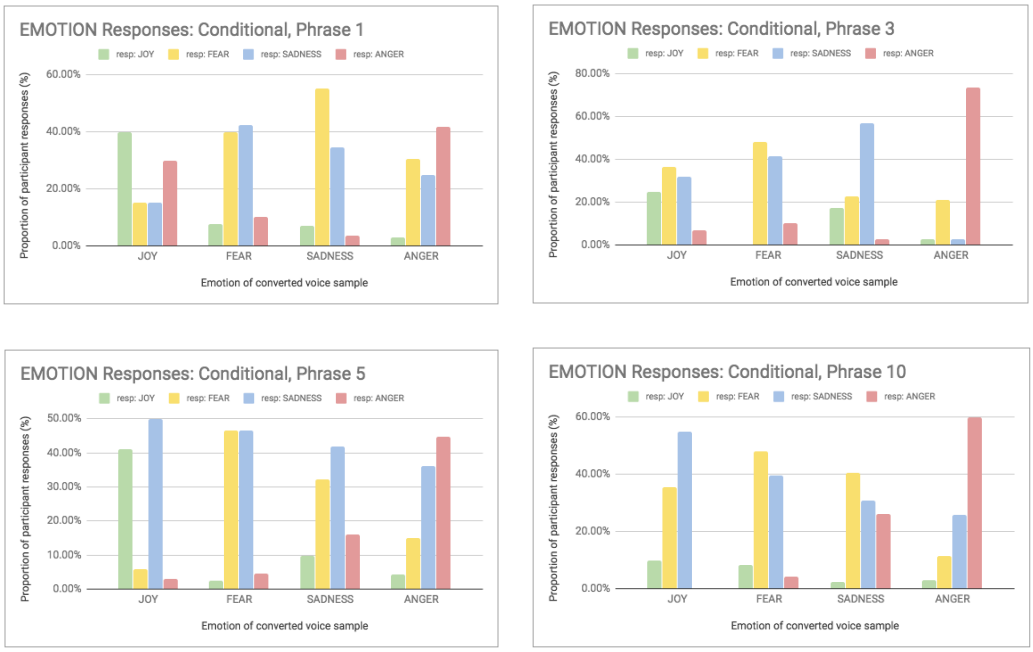

Figure 4.13: Emotion responses for conditional model by test phrase

Chapter 5

Conclusions

5.1 Comparison with Veaux & Rodet 2011

In 2011, Christophe Veaux and Xavier Rodet presented a conversion method for F0 contours trained on the same parallel database of acted emotional speech, Olivia2006, as used in this paper.

Target	**Perceived Emotion**			
	Joy	Fear	Sadness	Anger
Joy	**64,4**	6.8	20,3	8.5
Fear	9.4	**55,4**	6.4	28.7
Sadness	19,3	6.1	**73,4**	1.2
Anger	7.1	31.2	1.5	**60,1**

Figure 5.1: Perceptual classification scores, from Veaux & Rodet 2011

In their work, the F0 segments were represented by discrete cosine transform (DCT) coefficients at the syllable level, and multi-level dynamic features were added to model the temporal correlation between syllables and to constrain the F0 contour at the phrase level. Gaussian mixture models (GMM) were used to map the prosodic features between neutral and expressive speech, and the converted F0 contour was generated under the dynamic features constraints. Their experimental evaluation produced the results shown in Figure 5.1. We can draw many interesting comparisons between the two studies.

Joy is converted much more successfully by their model, with a 64% success rate compared to 40% in the best of our models. This is likely due to their use of multi-level features that mapped the relationship between syllables, which our model did not. Joy is the emotion with the greatest of frequency fluctuations, and this with this component missing for our model, much of the crucial information is lost.

Fear has almost identical results to the responses we obtained from the original samples, with around one-third of respondents confusing it for anger, and 55% correctly identifying it. This suggests that their conversion method failed to effectively remove the anger from the voice, whereas in our model, fear was most often confused with sadness, subjectively a much more similar sounding emotion.

Sadness has the highest recognition rate in both studies, with 73% for Veaux 2011 and 48% for our conditional model. Unlike our model however, their sadness conversion was most likely confused with joy, whereas ours with with fear.

Finally, anger has a 60% recognition rate in their study, compared to 55% in our conditional model, results which are very similar. Anger is typically the most easily recognisable emotion of the four in the literature.

Overall, our work reached comparable results in three of the four main emotions. One advantage of our model is that the sequence-to-sequence architecture is generative, and therefore the length of the predicted contours is determined by the model. In contrast, the method used in their paper forced the duration of the output contours by using the durations of the target data, which would not be possible in real-world setting using unseen data. By using the state-of-the-art machine learning techniques, our model is easily extensible in a number of ways, and with further work, the results could be made to reach parity with of Veaux 2011.

5.2 Shortcomings and future work

This work has a number of limitations which could be addressed in future work.

5.2.1 Small homogenous dataset

The dataset used for training and testing the model is very small, and was made even smaller by the selection of a subset of emotions and intensities, and the train/validation/test split. The smaller the dataset, the fewer examples of each phoneme there are, and the less generalisable the model will be. A second dataset expressing the same emotions is available, Phillippe2010, and future work could merge this with Olivia2006. However, as the genders of the speakers are different, the vocal frequency ranges are too, and care would have to be taken to normalise the frequency values before training a model with them.

5.2.2 Single speaker, single gender

Using only the Olivia2006 dataset means the model has only been trained on a single speaker, and a single gender (female). In addition to the dataset being small, both of these attributes suggest the model is unlikely to generalise well to other speakers. A larger, more varied dataset should be used for training, with new phrases from speakers not seen during training used as test examples.

5.2.3 Frequency only

This model only learns the mapping between F0 contours of voiced syllables, meaning it can only transform the pitch, and to some degree, the pacing of the source file. It does not take into account the other aspects of expression of emotion in the voice, such as volume of the voice, or the stress in the voice, which are important elements. Additionally, the duration of all silences and unvoiced syllables is not modelled, so pacing of the transformed file is not as accurate as it could be.

5.2.4 Phonemes mapped in isolation, single level only

This seq2seq model treats each phoneme in isolation, independent from those that came before or after it in the phrase. Clearly this lack of memory capability means a lot of the context for each F0 contour is lost. The order of the syllables and phonemes could be taken into account with a multi-tier architecture, as has been done in other projects. Both syllable-level and phrase-level transcoders could be layered into the architecture of both encoder and decoder, in order to capture the dependencies between phonemes in a syllable, and between syllables in a phrase.

5.2.5 Speaker identity is not modelled and preserved

This model sometimes has the effect of changing the character of the voice, as the pitch is raised too high across the whole phrase. While this may succeed in imbuing the phrase with the target emotion, it also changes the voice so much as to sound like another person, which is likely an undesirable trait in a production-level system. The model could be extended to include speaker identity features, in order to preserve them in the generated output.

5.2.6 Linguistic information is not leveraged

It may also be possible to manually add some information to the dataset, such as the syllables or words should be stressed, in order to further enhance the generated expressivity. A vocabulary of often-stressed words or word-types could be provided, so that the information contained in the text could be searched and exploited. The length of phrases and the position of the word/syllable/phoneme in the phrase is another linguistic feature that is likely to affect the prosody of the utterance.

5.2.7 Parallel database required

This model is trained with parallel data i.e. a speaker utters the each sentence in multiple emotions. However, the construction of such constraining parallel databases is time-consuming and expensive, and not practical for many real-world applications. Ideally we could learn the F0 mapping using 'on-the-fly' data, so the model could learn from all recorded speech from multiple speakers, with each sentence uttered only once. This may be achievable by clustering the linguistic context in which the pitch contours are

observed, and training separate mappings between the phonemes contained within each cluster.

5.3 Final remarks

In conclusion, we can say that adding conditioning to the model was an important innovation that helped improve results overall. We see that despite some of the original emotions being ambiguous to participants, the model was able to extract pertinent features for each, and generate an F0 contour suitable for converting the phrase convincingly. The lack of a large and varied database, the absence of inter-syllable correlations, and mapping only the frequency across voiced phonemes, are all obstacles to successfully training a model that can convert between a wide range of emotions and generalise to multiple speakers. Overall, this work has highlighted the capabilities and limitations of a sequence-to-sequence model when applied to F0, and it is hoped will serve as a basis for further research into this complex problem of voice emotion conversion.

Bibliography

Matplotlib: Python plotting - matplotlib 2.2.2 documentation. URL https://matplotlib.org/.

Scipy.org. URL https://www.scipy.org/.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. URL http://arxiv.org/abs/1409.0473.

Yuchen Fan, Yao Qian, Feng-Long Xie, and Frank K Soong. TTS synthesis with bidirectional LSTM based recurrent neural networks. *Interspeech*, 2014.

Google. google/seq2seq, Apr 2017. URL https://github.com/google/seq2seq/.

D. Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243, 1984. doi: 10.1109/tassp.1984.1164317.

Chin-Cheng Hsu, Hsin-Te Hwang, Yi-Chiao Wu, Yu Tsao, and Hsin-Min Wang. Voice conversion from unaligned corpora using variational autoencoding wasserstein generative adversarial networks. In *Interspeech 2017*, 2017.

IRCAM. AudioSculpt. http://forumnet.ircam.fr/product/audiosculpt-en/, 2018a. Accessed: 2018-3-26.

IRCAM. IrcamAlign | ircam anasynth. http://anasynth.ircam.fr/home/english/software/ircamali 2018b. Accessed: 2018-3-26.

Takuhiro Kaneko, Hirokazu Kameoka, Kaoru Hiramatsu, and Kunio Kashino. Sequence-to-Sequence voice conversion with similarity metric learned using generative adversarial networks. In *Interspeech 2017*, 2017.

Ki-Seung Lee. Restricted boltzmann Machine-Based voice conversion for nonparallel corpus. *IEEE Signal Process. Lett.*, 24(8):1103–1107, August 2017.

Runnan Li, Zhiyong Wu, Helen Meng, and Lianhong Cai. DBLSTM-based multi-task learning for pitch transformation in voice conversion. In *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, 2016.

Runnan Li, Zhiyong Wu, Yishuang Ning, Lifa Sun, Helen Meng, and Lianhong Cai. Spectro-Temporal modelling with Time-Frequency LSTM and structured output layer for voice conversion. In *Interspeech 2017*, 2017.

Zhaojie Luo, Jinhui Chen, Tetsuya Takiguchi, and Yasuo Ariki. Emotional voice conversion using neural networks with arbitrary scales F0 based on wavelet transform. *EURASIP Journal on Audio, Speech, and Music Processing*, 2017(1), 2017.

Huaiping Ming, Dongyan Huang, Lei Xie, Jie Wu, Minghui Dong, and Haizhou Li. Deep bidirectional LSTM modeling of timbre and prosody for emotional voice conversion. In *Interspeech 2016*, 2016.

Seyed Hamidreza Mohammadi and Alexander Kain. An overview of voice conversion systems. *Speech Commun.*, 88:65–82, 2017.

Robert Plutchik. The nature of emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice. *American Scientist*, 89(4):344–350, 2001. ISSN 00030996. URL `http://www.jstor.org/stable/27857503`.

Srikanth Ronanki, Gustav Eje Henter, Zhizheng Wu, and Simon King. A Template-Based approach for speech synthesis intonation generation using LSTMs. In *Interspeech 2016*, 2016.

Yuki Saito, Shinnosuke Takamichi, and Hiroshi Saruwatari. Statistical parametric speech synthesis incorporating generative adversarial networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(1):84–96, 2018.

Berrak Sisman, Haizhou Li, and Kay Chen Tan. Transformation of prosody in voice conversion. In *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1537–1546. IEEE, December 2017.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. September 2014.

Tensorflow. tensorflow/tensor2tensor. URL `https://github.com/tensorflow/tensor2tensor`.

Thang Luong, Eugene Brevdo,. Neural machine translation (seq2seq) tutorial | TensorFlow. `https://www.tensorflow.org/tutorials/seq2seq`. Accessed: 2018-3-23.

Aaron Van den oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. September 2016.

Christophe Veaux and Xavier Rodet. Intonation conversion from neutral to expressive speech. In *INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication Association, Florence, Italy, August 27-31, 2011*, pages 2765–2768, January 2011.

Vincent Wan, Yannis Agiomyrgiannakis, Hanna Silen, and Jakub Vít. Google's Next-Generation Real-Time Unit-Selection synthesizer using Sequence-to-Sequence LSTM-Based autoencoders. In *Interspeech 2017*, 2017.

Xin Wang, Shinji Takaki, and Junichi Yamagishi. An RNN-Based quantized F0 model with Multi-Tier feedback links for Text-to-Speech synthesis. In *Interspeech 2017*, 2017.

Jie Wu, Zhizheng Wu, and Lei Xie. On the use of i-vectors and average voice model for voice conversion without parallel data. In *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, pages 1–6. IEEE, December 2016.

Heiga Zen. Statistical parametric speech synthesis: from HMM to LSTM-RNN. 2015.

Heiga Zen and Hasim Sak. Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

Heiga Zen, Yannis Agiomyrgiannakis, Niels Egberts, Fergus Henderson, and Przemysław Szczepaniak. Fast, compact, and high quality LSTM-RNN based statistical parametric speech synthesizers for mobile devices. In *Interspeech 2016*, 2016.

List of Figures