Improvements of Silent Speech Interface Algorithms

PhD Thesis

Amin Honarmandi Shandiz
Supervisor: Dr. László Tóth

Doctoral School of Computer Science
Department of Computer Algorithms and Artificial Intelligence
Faculty of Science and Informatics
University of Szeged

Szeged
2023

Preface

Speech is a vital mode of communication, but for some individuals, speaking out loud may not be an option. Silent speech interfaces are a promising technology that allows for speech generation from articulatory signals, enabling individuals who are unable to speak to communicate. The focus of this topic is on the development and improvement of silent speech interfaces, which involves generating speech from data gathered from the movement of the tongue, lips, and jaw.

To explore this topic, I have planned a comprehensive agenda that covers various aspects of silent speech interface development. The agenda includes the following topics:

Preparing Data: This topic covers the collection and processing of data from the articulatory movements of a speaker. It includes data acquisition techniques such as Electromagnetic Articulography (EMA), Ultrasound Imaging, and Magnetic Resonance Imaging (MRI).

Different Model Implementations: This topic covers various models that have been implemented for generating speech from articulatory signals, such as deep neural networks, support vector machines, and Hidden Markov Models.

New Evaluation Metrics: This topic covers the development of new evaluation metrics for assessing the performance of silent speech interfaces. These metrics aim to provide a more accurate measure of speech quality and intelligibility, as traditional metrics may not be suitable for evaluating speech generated from articulatory signals.

Model Improvement: This topic covers various techniques for improving the performance of silent speech interfaces, such as regularization, transfer learning, and data augmentation.

Generalization of the Model for Unseen Data: This topic covers the development of models that can generalize well to new data and unseen speakers, which is essential for practical applications of silent speech interfaces.

Throughout this topic, I have relied on extensive research and consultations with experts in the field. Their insights and guidance have been invaluable in developing a comprehensive understanding of silent speech interfaces.

I am grateful to my professor and colleagues who have provided invaluable support and guidance throughout the research process. Their feedback and insights have been instrumental in shaping this topic, and I am grateful for their contributions. Last

but not least, I would like to express my gratitude to my family, friends, colleagues, and anyone who played a significant role in helping me reach this point in my life with their support.

”As one stage comes to an end, another stage begins.”
Amin Honarmandi Shandiz, August 2023

Contents

List of Figures

List of Tables

Abbreviations

3D CNN 3D Convolutional Neural Networks 11

AI Artificial Intelligence 11

ANN Artificial neural networks 23

CGANs Conditional Generative Adversarial Networks 45

CNN Convolutional Neural Networks 7

DCT Discrete Cosine Transform 21

DFT discrete Fourier transform 20

DNN Deep Neural Networks 7

EEG Electroencephalogram 7

ELTE Eötvös Loránd University 10

EMA Electromagnetic articulography 7

EMG Electromyography 8

FBANK Filterbank 19

fMRI functional MRI 8

fNIRS functional near-infrared spectroscopy 42

GAN Generative Adversarial Network 41

GRU gated recurrent unit 32

LDA Linear Discriminant Analysis 19

LPC Linear Predictive Coding 22

LSP	Line Spectral Pair representation 34
LSTM	Long Short-Term Memory 11
MFCC	Mel-frequency Cepstral Coefficients 19
MFGG	Mel-Generalized Cepstral Coefficients 34
MRI	Magnetic Resonance Imaging 8
NLP	Natural Language Processing 25
PCA	Principal Component Analysis 19
PLP	Perceptual Linear Prediction 19
RNN	Recurrent Neural Networks 7
SSI	Silent Speech Interface 7
STFT	Short-Time Fourier Transform 19
TDNN	time-delay neural networks 32
TTS	Text-to-speech 25
UDI	Ultrasound Direct Imaging 9
VAD	voice activity detection 12

Chapter 1

Introduction

A silent speech interface is a technology that allows individuals to communicate without speaking out loud. It works by detecting the movements of the tongue, lips, jaw, and other facial muscles involved in speech production (Articulatory signals) and then translating those movements into speech (Acoustic signals) or text. Silent speech interfaces are primarily designed to assist people who are unable to speak or have difficulty speaking due to medical conditions such as neurological disorders, tracheostomies, or other speech impairments. By using silent speech interfaces, these individuals can communicate more effectively and independently, without the need for a human intermediary or traditional assistive devices such as speech-generating devices.

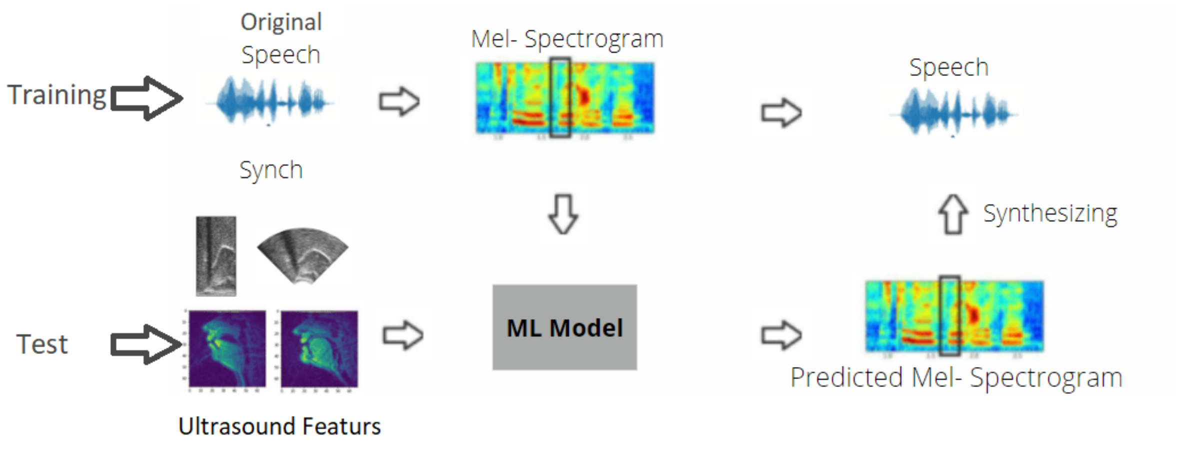
As you can see in figure 1.1 the Silent Speech Interface (SSI) process typically includes several components that are involved in converting silent speech into recognizable audio signals. The process begins with the acquisition of silent speech using sensors such as ultrasound, Electromagnetic articulography (EMA), or Electroencephalogram (EEG). The acquired data is then preprocessed to remove noise and irrelevant information, and feature extraction techniques are applied to identify relevant articulatory features.

The extracted features are then used to train machine learning models that can map the articulatory features to corresponding acoustic features. The machine learning models used in this process could be based on different architectures such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), or Deep Neural Networks (DNN).

After training, the machine learning models are used to predict the acoustic features of the silent speech signals. Finally, the predicted acoustic features are synthesized into audible speech using speech synthesis techniques such as concatenative synthesis or parametric synthesis.

The figure of the SSI process gives a clear visual representation of the various steps involved in the conversion of silent speech into recognizable audio signals, see fig 1.1. It highlights the importance of each component in the process and how they work together to achieve the final output. This can be useful for researchers and engineers working in the field of SSI to better understand the process and develop improved methods for converting silent speech into speech signals.

Figure 1.1: *The figure shows the process of speech synthesis from ultrasound tongue images using a silent speech interface(the original speech used only during the training).*



SSI research has utilized various recording techniques, including EEG, Electromyography (EMG), Magnetic Resonance Imaging (MRI), ultrasound, and others, to develop technologies that enable communication without the need for vocalization.

The earliest implementation of a SSI used EMG to detect the electrical signals generated by the muscles involved in speech production. In the early 1970s [99], researchers began exploring the possibility of using EMG to develop a device that could generate speech from muscle movement, without the need for vocalization.

Around the same time, the EEG was introduced as a tool for SSI research, as it can provide real-time measurement of brain activity with high temporal resolution. EEG has since been used to study the neural mechanisms underlying speech production and to develop SSI technologies that can translate brain signals into speech.

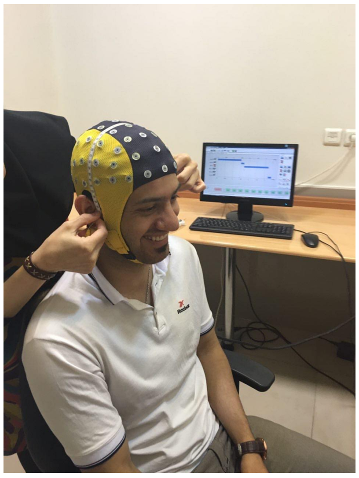
Advancements in MRI technology in the following decades led to the development of functional MRI (fMRI) for studying brain activity during speech production. FMRI has since been used to study the neural pathways involved in speech production and to develop new SSI technologies that can decode brain signals into speech.

Ultrasound was introduced as a tool for SSI research in the 1980s, with researchers using it to image the tongue, lips, and jaw during speech production. Ultrasound imaging was later combined with EMG to create the EMA technique, which allowed researchers to track the movement of the tongue and other articulators during speech production.

Some recording devices such as EEG, EMG were one of the earliest implementations in SSI because they could provide real-time measurement of brain activity with high temporal resolution. This means that EEG can capture changes in brain activity that occur in a matter of milliseconds, making it a useful tool for detecting neural activity associated with speech production in real-time. Additionally, EEG is non-invasive, portable, and relatively inexpensive compared to other imaging techniques such as FMRI, which makes it a more accessible option for researchers and clinicians who want to develop SSI applications. Therefore, EEG has been an attractive option for researchers interested in developing SSI applications for individuals with communication difficulties.

Figure 1.2 shows an individual seated and wearing a cap with multiple electrodes attached to their scalp. These electrodes are designed to detect and record the electrical activity of the individual's brain as they attempt to produce speech without vocalizing. This electrical activity can then be analyzed and used to generate speech output through a computer interface, providing a means of communication for individuals who are unable to speak or have difficulty speaking.

Figure 1.2: *The figure depicts a person who is undergoing EEG recording for silent speech interface (SSI).*



While EEG has the advantage of providing real-time measurement with high temporal resolution, it has limitations in terms of spatial resolution and specificity. EEG signals are influenced by the activity of large groups of neurons, so it can be difficult to determine the exact location of neural activity using EEG alone. This is where other imaging techniques such as MRI and Ultrasound Direct Imaging (UDI) can be useful. MRI provides excellent spatial resolution, allowing researchers to locate the exact areas of the brain that are active during speech production. UDI can also provide detailed imaging of the structures involved in speech production, such as the tongue, lips, and jaw. Combining these imaging techniques with EEG can provide a more complete picture of neural activity during speech production, and can help improve the accuracy and reliability of SSI applications. Therefore, different imaging techniques are used in SSI research to address the limitations of each technique and to obtain a more comprehensive understanding of the neural mechanisms underlying speech production.

In explanation of different ways of recording articulatory signals, UDI is just one of several recording devices used for SSI development, and each device has its own strengths and limitations. Compared to other recording devices, such as EEG, EMG, and MRI, UDI has several advantages for SSI development.

Firstly, ultrasound provides direct imaging of the articulators, which are the tongue, lips, and jaw involved in speech production. In contrast, EEG and EMG measure electrical activity in the brain and muscles, respectively, which indirectly infer the movements of the articulators. Ultrasound's direct imaging provides a more accurate and detailed representation of the movements of the articulators during speech, which can help improve the accuracy of SSI systems.

Secondly, ultrasound has high temporal and spatial resolution, which means it can capture movements in real-time and with high precision. EEG and EMG have high temporal resolution but lower spatial resolution compared to ultrasound. MRI has high spatial resolution but lower temporal resolution compared to ultrasound. Ultrasound's high temporal and spatial resolution allows for precise tracking of articulator movements, which is important for accurate speech synthesis.

Thirdly, ultrasound is a non-invasive and safe method for recording speech production. EEG and EMG are also non-invasive but may require electrodes to be attached to the skin or scalp, which can cause discomfort or irritation. MRI is safe but requires the participant to lie still in a confined space for an extended period, which can be uncomfortable for some individuals and not good for practical use.

Lastly, ultrasound is portable and cost-effective compared to other recording devices, which makes it more accessible for SSI research and development.

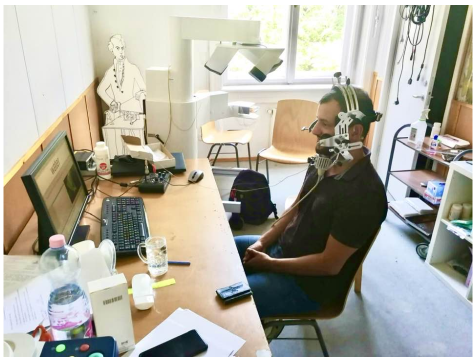
Overall, while each recording device has its own strengths and limitations, ultrasound's direct imaging of the articulators, high temporal and spatial resolution, non-invasive nature, cost-effectiveness and portability make it a better way and more relevant nowadays for SSI development, especially in clinical settings and with a wider range of populations.

Figure 1.3 shows a person sitting in a room at Eötvös Loránd University (ELTE) in Budapest, Hungary. The person sits on a chair while an ultrasound probe is placed under the jaw to record tongue movements, and the probe is connected to a computer via a cable. The computer is running specialized software to capture and analyze the ultrasound signals produced by the person's vocal tract during silent speech.

Ultrasound is a non-invasive and safe technique that uses high-frequency sound waves to create images of internal structures in the body. In the case of a SSI, ultrasound is used to capture the movement of the tongue, lips, and jaw, which are important in generating speech. By placing the ultrasound probe on the throat, the researcher can visualize the shape and movement of the vocal tract during silent speech, and use this information to generate speech sounds.

The use of ultrasound for SSI has several advantages. It provides high-resolution images of the vocal tract and can capture detailed movements that are not easily visible with other techniques. Additionally, it is safe and non-invasive, making it suitable for use with a wide range of individuals, including children and those with medical conditions.

Figure 1.3: *A person using an ultrasound probe to record their vocal tract movements during silent speech, which is a useful technique for developing SSI technology.*



In my thesis, I focused on exploring the advantages of ultrasound recordings for SSI and its relation to Artificial Intelligence (AI). I found that previous methods for SSI using ultrasound recordings suffered from high error rates and long training times beside lack of generalization ability over unseed data, which made them less practical for real-world applications.

To address these challenges, I implemented several new models and techniques. Firstly, I developed new models that significantly improved the synthesized data error rate, which is critical to generate good quality speech. To achieve this, I proposed a novel approach using 3D Convolutional Neural Networks (3D CNN).

Specifically, I first converted the 2D ultrasound data into a 3D format by using a sliding window approach. This involved breaking the data down into small windows and then sliding the window along the time axis, with a certain amount of overlap between each window and the previous one. By doing this, I was able to create a 3D representation of the data that could be fed into a 3D CNN and use the technique of parallel processing.

Next, I compared the performance of this 3D CNN approach with other commonly used methods, such as RNN like Long Short-Term Memory (LSTM) networks. After conducting extensive experiments, I found that the 3D-CNN approach outperformed the other methods, resulting in significantly improved synthesized data error rates.

Secondly, I improved the training model time by implementing a new technique that optimizes the use of hardware resources. This resulted in a faster and more efficient training process. To do that, I propose a new combination of ConvLSTM and 3D-CNN, By using a combination network of ConvLSTM and Conv3D, I was able to decrease the model training time. ConvLSTM is a type of RNN that is effective for processing sequential data, while Conv3D is a type of CNN that is efficient in processing volumetric data. By combining the two networks, I was able to effectively process both sequential and volumetric data. Moreover, I decreased the number of layers and the number of units in each layer, which helped to reduce the model's complexity and therefore its training time. As a result, the training process became faster and more efficient, without compromising the accuracy of the model.

Additionally, I introduced a new technique for preprocessing the data that improves the quality of the recorded signals and makes it easier to analyze them. When recording speech signals for SSI, there may be periods of silence or background noise that can interfere with the analysis. Traditionally, researchers have used algorithms to remove these silent or noisy portions of the speech before analyzing the remaining signal. However, this can be challenging, as it is difficult to distinguish between silent portions of speech and silence due to background noise.

A better approach is to use voice activity detection (VAD) algorithms to identify which portions of the speech signal actually contain speech. Therefore, to solve this challenge we divide the speech signal into frames and analyze each frame separately, VAD algorithms can accurately detect the presence of speech and remove any silent portions within the frames. This makes it easier to analyze the remaining signal and improves the overall quality of the recorded signals.

In contrast, when the entire speech signal is given as input to a silence remover, it may not be able to accurately detect and remove all silent portions of the speech, leading to a loss of important information in the recorded signals. Therefore, using VAD algorithms over the speech signals in frame level is a more effective way to improve the quality of the recorded signals and make them easier to analyze for SSI.

I also developed a new method to generalize the data for unseen speakers, which is crucial for SSI to be useful in real-world settings where the speaker is not known in advance. By using x-vector features of the speakers in the speaker recognition method and embedding them as features in the ultrasound-to-speech model, the model can better capture the unique characteristics of each speaker's voice. This can help improve the generalization ability of the model to unseen data from different speakers, as the model can better differentiate between speakers and adapt to their unique voice characteristics. The x-vector features can also help reduce the effect of inter-speaker variability and make the model more robust to variations in voice quality, accent, and speaking style. As a result, the model can perform better on unseen data, which is important for the practical application of SSI systems in real-world scenarios where there is a wide range of speakers and speaking conditions.

Finally, another challenge was improving the accuracy and generalization ability of neural silent speech interface models. The traditional methods for speech recognition rely on audio signals, but silent speech interface models use ultrasound signals, which have a different structure and pose unique challenges for processing and analysis.

To address this challenge, we proposed a novel method of using adversarial training to improve the accuracy and generalization ability of neural silent speech interface models. Adversarial training involves using two neural networks: a generator network that generates synthesized speech signals from the ultrasound data, and a discriminator network that tries to distinguish between the synthesized speech signals and real speech signals. The two networks are trained simultaneously, with the goal of improving the accuracy and generalization ability of the generator network.

We found that our proposed method improved the accuracy and generalization ability of the neural silent speech interface models, as compared to traditional methods. This has important implications for the development of more accurate and reliable silent speech interface systems, which can be used in a variety of applications such as speech therapy, human-computer interaction, and assistive technology.

1.1 Speech Production and Articulatory Features

1.1.1 Speech Production

Speech production is the process of producing speech sounds using various parts of the human body, such as the lungs, vocal cords, tongue, lips, and teeth. The process involves a series of coordinated movements and adjustments of these articulators to produce different sounds and speech patterns.

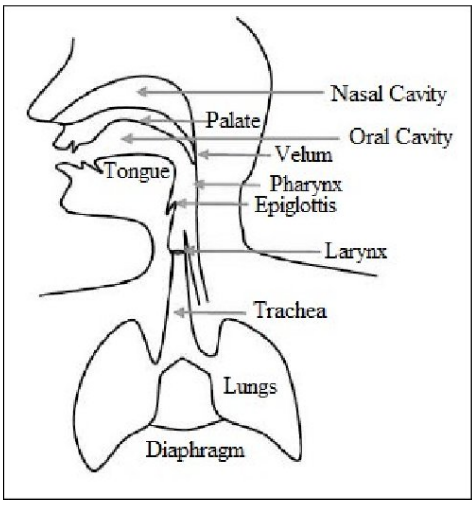
The production of speech involves three main stages: the initiation of an idea or thought, the formulation of that idea into a linguistic structure, and the actual articulation or production of the speech sounds. During the initiation stage, the speaker decides what they want to say and how they want to say it. In the formulation stage, the speaker selects the appropriate words, phrases, and grammatical structures to convey their message. Finally, during the articulation stage, the speaker produces the sounds of the language using their articulators.

Speech production is a complex process that involves the coordination of multiple systems in the human body, including the respiratory, phonatory, and articulatory systems. Any disruptions in these systems can affect speech production and result in speech disorders or difficulties. Understanding the mechanisms of speech production is important for developing interventions and treatments for speech disorders, as well as for developing technologies such as speech recognition and synthesis systems.

Figure 1.4 titled ”speech production anatomy” depicts the different structures and organs involved in the production of speech sounds. At the bottom of the figure, we see the lungs, which provide the air pressure required for speech. The air passes through the trachea and into the larynx, where the vocal cords are located. The vocal cords vibrate when the air passes through them, producing sound.

Moving further up the figure, we see the pharynx and the oral cavity, which shape the sound waves produced by the vocal cords into specific speech sounds. The tongue, teeth, and lips play a critical role in shaping the sound waves, creating different consonant and vowel sounds. The nasal cavity is also shown, which allows for the production of nasal sounds.

Figure 1.4: *This figure provides a comprehensive view of the complex process involved in producing speech and highlights the crucial role played by different structures and organs in this process [55].*



Finally, the motor cortex is responsible for controlling the movement and coordination of the muscles involved in speech production. It sends signals to the articulators, such as the tongue and lips, to produce specific speech sounds. The auditory cortex, on the other hand, processes the sound of our own voice, which enables us to monitor and adjust our speech as required. Therefore, the brain plays a crucial role in speech production, not only in controlling the physical movements necessary for speech but also in providing feedback to ensure that we speak accurately.

Figure 1.5 compares the speech production process in humans with that of a speech coder.

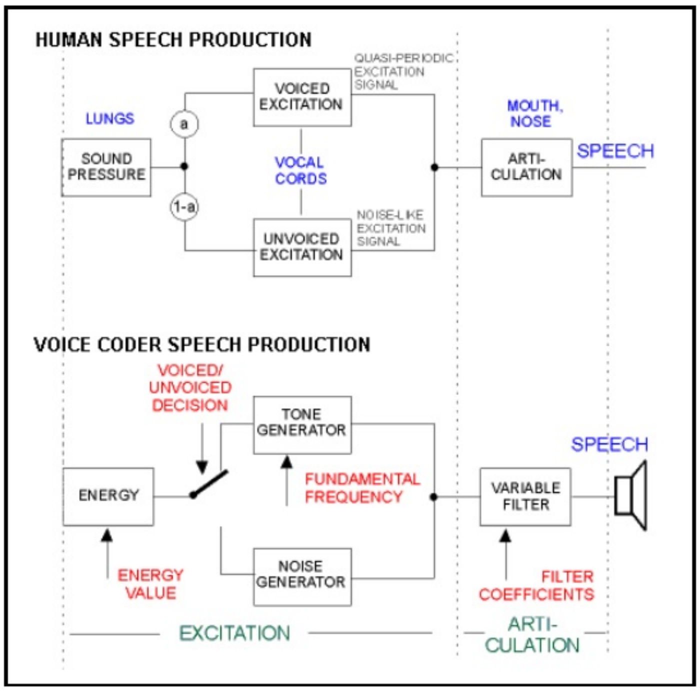
It shows a block diagram of the speech production process in humans, which involves several anatomical components such as the lungs, vocal cords, throat, and mouth. The air from the lungs is passed through the vocal cords, which vibrate to produce a source signal. This signal is then modified by the vocal tract to produce the final speech signal.

The figure also shows a block diagram of a speech coder, which consists of an excitation model and an articulatory model, as described in the previous figure. The excitation model generates a waveform that represents the sound source for speech production, while the articulatory model transforms the excitation waveform into speech.

The comparison between the two block diagrams illustrates that while the human speech production process is complex and involves multiple anatomical components, a speech coder can model the process using only two main components: an excitation model and an articulatory model.

The figure also shows several sub-blocks within the excitation and articulatory

Figure 1.5: *General voice or speech coder: Human vs. Voice Coder Speech Production [10].*



models. These sub-blocks may include signal analysis, quantization, coding, and decoding stages. These stages are used to process the speech signal and reduce the amount of data needed to transmit or store it.

1.1.2 Articulatory-To-Acoustic Conversion

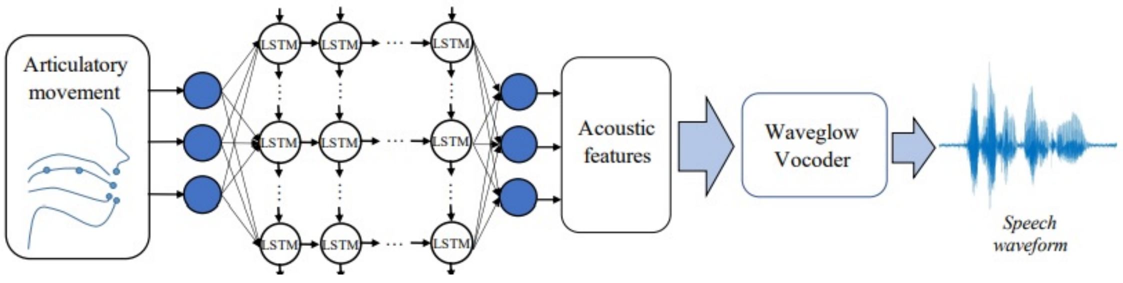
Articulatory-to-acoustic conversion is the process of transforming articulatory features (which represent the movements of the articulators in the vocal tract during speech production) into acoustic features (which represent the acoustic properties of the speech signal). This conversion can be challenging due to the complex and nonlinear relationship between articulatory and acoustic features.

Deep learning, specifically neural networks, has been used to tackle this problem. One approach is to train a neural network to map articulatory features to acoustic features directly, without the need for intermediate steps such as acoustic modeling or speech synthesis. This can be achieved using a variety of neural network architectures such as feedforward neural networks, recurrent neural networks, and convolutional neural networks.

WaveGlow is a neural vocoder that used for speech synthesis that uses a flow-based generative model to produce high-quality audio waveforms. It has also been used for articulatory-to-acoustic conversion, where the articulatory features are used to generate the speech waveform directly [75].

In this approach, the articulatory features are first transformed into a latent representation using an encoder network. The latent representation is then used as input to the WaveGlow model, which generates the corresponding acoustic waveform during training. The resulting acoustic waveform is compared to the waveform using a loss function, and the parameters of the encoder network and WaveGlow model are optimized to minimize this loss.

Figure 1.6: *Process of transforming articulatory features to acoustic features using deep learning methods for speech synthesis [12].*



The figure 1.6 is an example of a flowchart that illustrates the process of transforming articulatory features (such as tongue and lip positions) to acoustic features (such as spectral envelopes and prosody) using deep learning methods LSTM for speech synthesis. It shows the various stages involved in the conversion process, such as data acquisition and feature extraction until synthesizing the speech [12].

Articulatory Features

Articulatory features describe the different movements and configurations made by the speech organs during speech production. These features describes the position and movement of the tongue, lips, velum, jaw, and vocal folds. The use of articulatory features in speech processing and recognition is an active area of research, as it can provide valuable information about the articulation patterns of a speaker [20]. Articulatory features are important in the context of articulatory-to-acoustic models that aim to synthesize speech by mapping articulatory movements to acoustic features. These models rely on capturing the relationship between the movements of the vocal tract and the resulting acoustic signals produced by them. The use of articulatory features in these models allows for a more direct mapping between the physical processes involved in speech production and the resulting acoustic signal. Articulatory-to-acoustic models have been developed in recent years, and they have shown promising results in synthesizing natural-sounding speech. One example is the work of [39], which proposes an articulatory-to-acoustic mapping model that uses a combination of deep neural networks and articulatory features to synthesize speech from articulatory data. The model is based on a three-stage architecture that first predicts articulatory features from raw acoustic data, then maps these features to a compact representation, and finally generates a high-quality acoustic signal from this representation.

The use of articulatory features in articulatory-to-acoustic models can help improve the accuracy and naturalness of synthesized speech. These models have the potential to be used in various applications, such as speech synthesis for individuals with speech impairments, and the development of virtual assistants and chatbots that can produce natural-sounding speech.

Acoustic featurs

Acoustic signals refer to the sound waves produced by speech and other sources, which can be captured and analyzed using digital signal processing techniques. Acoustic features, on the other hand, refer to the measurable characteristics of the sound signal that are used to distinguish between different sounds or analyze their properties.

In speech analysis, some common acoustic features are pitch, formants, spectral features, and energy. Pitch refers to the perceived frequency of the voice, which can provide information about the speaker's gender, age, and emotional state. Formants are the resonant frequencies of the vocal tract, which can be used to distinguish between different vowel sounds. Spectral features, such as spectral centroid, spectral flatness, and spectral roll-off, provide information about the spectral content of the signal. Energy, on the other hand, refers to the intensity or loudness of the signal and can be used to detect changes in emphasis or emotion.

In addition to speech analysis, acoustic features are used in a variety of applications such as music analysis, speaker identification, and emotion recognition. For example, in music analysis, spectral features are often used to detect different instruments or identify musical genres. In speaker identification, pitch and formants can be used to distinguish between different speakers. In emotion recognition, energy and spectral features can be used to detect changes in a speaker's emotional state.

1.2 Feature Extraction for Speech Processing

Feature extraction is a critical step that involves transforming raw speech data into a set of meaningful features that can be used for analysis and modeling. The main goal of feature extraction is to capture the relevant information from the speech signal that is important for the intended application, such as speech recognition, speaker identification, or emotion recognition.

The process of feature extraction involves several steps, including pre-processing, feature selection, and feature transformation. Pre-processing involves filtering and normalization of the speech signal to remove noise and other artifacts. Feature selection involves choosing a subset of features that are most relevant for the application at hand, based on their discriminative power and computational complexity. Feature transformation involves transforming the selected features into a new representation that is more suitable for the intended analysis or modeling approach.

Some commonly used features in speech processing include Mel-frequency Cepstral Coefficients (MFCC), Linear Predictive Coding (LPC) coefficients, pitch, formants, and energy. MFCC are widely used in speech recognition and speaker identification, as they capture the spectral envelope of the speech signal and are relatively robust to noise and other distortions. LPC coefficients are used to model the vocal tract characteristics of speech and can be used for speech synthesis and compression. Pitch and formants are useful for identifying the prosodic and phonetic characteristics of speech, respectively. Energy is often used to detect changes in loudness and stress in speech [110].

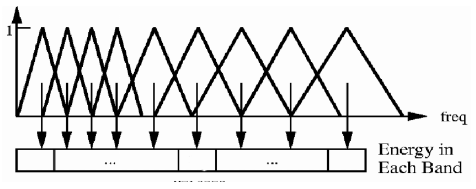
1.2.1 Filterbanks

Filterbank (FBANK) features are calculated using a Filterbank, which is a set of bandpass filters that simulate the human auditory system's frequency response. The Filterbanks is typically applied to the magnitude of the Short-Time Fourier Transform (STFT) of the speech signal, resulting in a series of Filter outputs that represent the energy in each frequency band over time. The Filterbanks are outputs are then typically transformed using a logarithmic compression to approximate the human perception of loudness.

One of the main challenges with using FBANKs features as features for SSI models is that they are highly correlated, which can lead to collinearity issues and suboptimal performance in the model. To overcome this, various methods have been proposed, such as Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA), to reduce the dimensionality and decorrelate the features before using them in the model. Additionally, other feature extraction techniques, such as MFCC or Perceptual Linear Prediction (PLP), have been developed that capture similar spectral information as Filterbank features but with improved performance in SSI tasks.

Figure1.7 shows Filterbank features in speech feature extraction which contains of the process of filtering a speech signal using a bank of triangular filters on the mel-scale, resulting in a set of Filterbank coefficients that represent the power of the signal within each frequency band, which can be further processed and used as features for speech recognition or other speech-related tasks.

Figure 1.7: *The horizontal axis of the figure represents the frequency in hertz (Hz), and the vertical axis represents the amplitude. The lower frequency range is shown on the left side of the graph, and the higher frequency range is shown on the right side of the graph. The power spectrum of the speech signal is represented by a series of bars that indicate the energy level at each frequency [110].*



The equation for computing the Filterbank features for a given speech signal x(t) is:

$$F_m = \sum_{k=0}^{N-1} |X(k)|^2 H_m(k) \tag{1.1}$$

where F_m is the FBANK output for the m^{th} filter, N is the length of FFT, X_k is the discrete Fourier transform (DFT) of the input signal $X(t)$, and $H_m(k)$ is the Filter weight for the m^{th} filter at frequency bin k. The Filter weights can be computed using a set of triangular filters that approximate the Mel scale, as shown in the figure 1.7.

Once the Filterbank output is computed, it is typically logarithmically compressed to obtain a more compact representation:

$$f_m = \log(f_m) \tag{1.2}$$

The resulting filterbank features f_m are then used as input to various speech processing tasks.

1.2.2 Mel-spectrogram

Mel-spectrograms (see Fig 1.8) are derived from the traditional spectrogram, which represents the spectrum of a signal as a function of time. However, instead of using a linear frequency scale, Mel-spectrogram features use a non-linear Mel-frequency scale that better approximates the human auditory system's response to sound.

To calculate Mel-spectrogram features, the speech signal is first divided into small frames of equal duration (typically 20-30 ms), and for each frame, the Fourier transform is computed to obtain the power spectrum. The power spectrum is then transformed into the Mel-scale using a Mel-filterbank, which consists of a set of triangular filters that are uniformly spaced in the Mel-frequency scale. The output of each filter is then summed, resulting in a set of Mel-spectrogram coefficients that represent the spectral content of the speech signal in different frequency bands, see Fig 1.7.

The number of Mel-spectrogram features depends on the number of filters used in the Mel-filterbank, which is typically between 20 and 40. Mel-spectrogram features have several advantages over other types of speech features, including their robustness to noise and their ability to capture important spectral information in different frequency bands. They are commonly used as input features in speech recognition and other speech-related tasks.

1.2.3 Mel-Frequency Cepstral Coefficient

Mel-Frequency Cepstral Coefficients are commonly used acoustic features in speech recognition systems. MFCCs are calculated by first applying a mel-scale filterbank to the power spectrum of the speech signal. The resulting mel-frequency spectrum is then transformed into the cepstral domain using the Discrete Cosine Transform (DCT). The first several coefficients, typically ranging from 12 to 40, are used as features for speech recognition.

MFCCs have several advantages as speech features. First, they are robust to variations in speech production caused by different speakers, speaking rates, and speaking styles. Second, they can capture both the spectral and temporal characteristics of speech. Third, they are computationally efficient to compute and have been widely used in many speech recognition systems.

Figure 1.8: *An example of a Mel-spectrogram for a speech signal [35].*

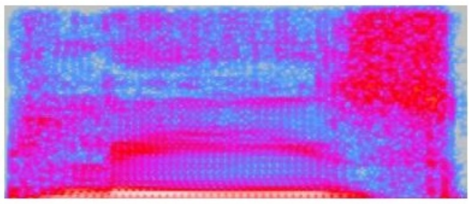


Figure 1.8 shows a visualization of the Mel-spectrogram for a speech signal. The MFCC are computed using a series of processing steps, including framing, windowing, Fourier transform, Mel filtering, logarithmic compression, and discrete cosine transform. The resulting MFCC are typically arranged in a matrix or image format, where each column corresponds to a different frame of the signal and each row corresponds to a different MFCC coefficient. The image provides a way to visualize the frequency and temporal variations in the MFCC and can be used as a feature representation for various speech processing tasks such as speech recognition, speaker identification, and emotion recognition.

WaveGlow

WaveGlow is a neural vocoder that generates high-quality speech waveforms from acoustic features, such as Mel-spectrograms or other spectral representations, using a flow-based generative model that enables efficient parallel generation and high-fidelity synthesis of speech.

1.2.4 Linear Predictive Coding

Linear Predictive Coding (LPC) is a widely used technique for speech signal processing that involves modeling the speech signal as a linear combination of past samples. The coefficients obtained from this linear prediction model are known as Linear Predictive Coding coefficients.

Linear Predictive Coding is based on the assumption that speech signals are highly correlated and can be predicted using a linear combination of past samples. The Linear Predictive Coding coefficients are obtained by minimizing the prediction error between the original speech signal and the predicted signal.

The Linear Predictive Coding coefficients are commonly used in speech coding, speech synthesis, and speech recognition applications. They can be calculated using the following formula:

$$a_k = \frac{\sum_{i=k+1}^{p} r_i a_{k-i}}{\sum_{i=0}^{p} r_i} \tag{1.3}$$

where a_k is the kth Linear Predictive Coding coefficient, p is the order of the Linear Predictive Coding model, r_i is the ith autocorrelation coefficient, and $a_\mathrm{k\text{-}i}$ is the k-ith Linear Predictive Coding coefficient. The Linear Predictive Coding coefficients can be calculated recursively using the Levinson-Durbin algorithm, which is an efficient algorithm for solving the set of linear equations that arise in Linear Predictive Coding analysis.

There are several visual representations of Linear Predictive Coding coefficients, including plots of the Linear Predictive Coding spectra and Linear Predictive Coding cepstra. These representations provide insight into the frequency content and spectral envelope of the speech signal [10].

1.2.5 Gammatone filter features

Gammatone filter features are another type of speech feature extraction method that is based on modeling the human auditory system. They are calculated by filtering the speech signal with a bank of gammatone filters, which are modeled after the tuning of the auditory system's hair cells. The output of each filter is then rectified and low-pass filtered, and the resulting signals are then used as features.

The number of features in gammatone filter features can vary depending on the number of filters used. Typically, a bank of 40-80 filters is used, resulting in a corresponding number of features.

One advantage of gammatone filter features is that they are designed to better capture the spectral characteristics of speech, particularly in the higher frequency ranges. This can be useful in tasks such as speaker recognition or speech recognition in noisy environments.

Gammatone filter features can be more computationally expensive to compute than other feature extraction methods, such as MFCCs. Additionally, they may require more data to train CNN models effectively.

1.3 Artificial Neural Networks

Artificial neural networks (ANN) are a type of machine learning model inspired by the structure and function of biological neurons in the human brain [9]. ANN consist of layers of interconnected nodes, or neurons, each performing a simple computation on its inputs and passing the result to the next layer. The connections between neurons have weights that are adjusted during training to optimize the model's performance on a given task.

1.3.1 Ativation Functions

In artificial neural networks, an activation function is applied to the output of each neuron in a layer to introduce non-linearity into the network. It allows the network to learn complex patterns and relationships in the data. There are several commonly used activation functions in ANNs. Here are a few examples:

The sigmoid function is defined as:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{1.4}$$

The ReLU function is defined as:

$$f(x) = max(0, x) \tag{1.5}$$

The tanh function is defined as:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{1.6}$$

This function is commonly used in ANNs as an activation function for hidden layers due to its smooth and symmetric S-shaped curve, which allows for effective gradient descent during backpropagation.

ANN consist a large number of interconnected processing nodes (artificial neurons) organized into layers. The most common types of ANNs include feedforward neural networks, recurrent neural networks, convolutional neural networks, and deep neural networks.

1.3.2 Convolutional Neural Networks

CNN stands for Convolutional Neural Network, which is a type of artificial neural network commonly used for image and video analysis. CNN is good for image processing because of its ability to automatically learn and extract hierarchical features from the input image, making it highly effective in handling complex image structures and patterns.

In the case of ultrasound articulatory to acoustic SSI, CNNs are well-suited because they can effectively extract relevant features from the ultrasound images to produce accurate acoustic predictions. The use of CNNs in this field has shown promising results in improving the accuracy and robustness of SSI models, making the networks more effective in various applications such as speech recognition, language translation, and human-computer interaction.

1.3.3 Long Short Term Memory

Long Short Term Memory networks are a type of Recurrent Neural Networks designed to overcome the problem of vanishing gradients in traditional RNNs, which can make it difficult to learn long-term dependencies in sequential data. LSTMs use a gating mechanism to selectively retain or forget information over time, allowing them to effectively model sequences with long-term dependencies.

In the context of ultrasound SSI models, LSTMs have been shown to be effective in modeling the complex relationships between articulatory movements and acoustic signals over time. By leveraging the memory and gating mechanisms of LSTMs, these models can capture the dynamics of speech production and generate accurate predictions of acoustic signals from ultrasound images.

LSTMs have been widely applied in various sequential data processing tasks, such as speech recognition, language translation, and time series prediction. They are especially useful in tasks where long-term dependencies are important, and where the input data may have varying lengths or irregular time frames.

1.3.4 Residual networks

Residual networks, also known as ResNets, are a type of deep neural network architecture that was introduced to address the problem of vanishing gradients in very deep networks. They are composed of residual blocks, which contain skip connections that allow for the gradient to flow directly from one layer to another.

In ultrasound SSI models, ResNets can be particularly useful for improving model performance because they allow for the training of deeper networks without suffering from the vanishing gradient problem. This can result in better feature representation and improved model accuracy.

ResNets have been widely used in various applications, such as image classification, object detection, and speech recognition. They are particularly effective in sequential data processing tasks, where the input data has a temporal or spatial order, such as audio and video processing, language modeling, and speech synthesis.

1.4 Text To Speech

Text-to-speech (TTS) is a technology that converts written text into spoken words. TTS systems use Natural Language Processing (NLP) and machine learning techniques to generate speech that is indistinguishable from human speech. TTS systems have many applications, including accessibility for the visually impaired, language learning, and automated customer service.

Tacotron2 is a deep learning-based TTS system developed by Google. It uses a neural network to generate speech from text inputs. The network is trained on a large dataset of paired text and speech examples, and it learns to map input text to a sequence of acoustic features that can be used to synthesize speech.

Tacotron2 is particularly well-suited for speech synthesis in difficult-to-model domains, such as ultrasound SSI. In these domains, the acoustic properties of speech are highly variable and can be difficult to model using traditional rule-based approaches. Tacotron2 is able to learn these complex patterns from data and generate high-quality speech.

One of the key advantages of Tacotron2 is its ability to generate speech with natural-sounding intonation and prosody. This is achieved through the use of an attention mechanism that allows the network to selectively attend to different parts of the input text as it generates speech. The attention mechanism ensures that the generated speech is closely aligned with the input text, and that it reflects the appropriate intonation and prosody.

It is able to produce speech with natural-sounding intonation and prosody.

1.5 Datasets

As a part of our research, we used an already existing Hungarian dataset for speech production research. The dataset contained parallel ultrasound and speech recordings from a Hungarian female subject reading sentences aloud, which were collected using specialized equipment. The whole duration of the recordings was around half an hour, from which 310 sentences were used for training, 41 for development and 87 for testing, respectively.

To compare our implementation methods for more languages, we decided to add an extra English dataset to our research. For this, we used the TAL corpus, which contains parallel recordings of speech, tongue ultrasound, and lip videos from 81 speakers. We selected the TAL1 subset of the corpus [82], which includes recordings of a single trained native English speaker. The recording conditions for the English dataset were very similar to those of the Hungarian dataset. After dividing the data into train, validation, and test sets, we obtained 1015, 50, and 24 utterances, respectively.

We applied the same preprocessing steps to the English Data, as we did for the Hungarian dataset to ensure consistency in our research. By adding the English dataset to our research, we were able to compare our implementation methods across multiple languages and draw more comprehensive conclusions.

1.6 Summary by chapters

The Thesis is organised as follows:

In **Chapter 2.**, we addresses the challenge of accurately predicting speech from ultrasound data using deep learning methods. The proposed method utilizes 3D convolutional neural networks to learn spatiotemporal features from the ultrasound data, which are then used to predict speech. The performance of the proposed method is evaluated on a Hungarian dataset consisting of ultrasound and speech recordings from a Hungarian speaker, and compared with other methods in the literature. The results demonstrate that the proposed method outperforms other methods in terms of accuracy, and is able to accurately predict speech from ultrasound data.

To improve the results of this approach, we added more combined networks with an LSTM implementation, which was published in a Hungarian conference. The combined networks consist of a 3D CNN and a Long Short-Term Memory network. The addition of the LSTM network allows the proposed approach to capture the temporal dependencies in the ultrasound data, which further improves the accuracy of speech decoding.

Chapter 3. presents a method to improve the performance of neural models used for silent speech interfaces by incorporating adversarial training. The main challenge in SSIs is the limited availability of training data, which leads to overfitting and poor generalization. To address this issue, we proposed an adversarial training approach that incorporates a discriminator network to guide the generator network towards producing more realistic and diverse speech representations.

The proposed method involves training a generator network to generate realistic speech signals from silent articulation data and a discriminator network to distinguish between generated speech signals and real speech signals. The generator and discriminator networks are trained in an adversarial manner, where the generator is trained to fool the discriminator, and the discriminator is trained to accurately classify the generated and real speech signals. This adversarial training process improves the ability of the generator network to generate more realistic speech signals, which in turn improves the performance of the silent speech interface model.

The performance of the proposed method is evaluated on two datasets including Hungarian and English and compared with several baseline methods and the results were also avaluated with several metrics. The results show that the adversarial training approach significantly improves the performance of the silent speech interface model, compared to the best baseline method. Additionally, the proposed method is shown to be robust to variations in the input data, such as different speakers and speaking styles, which makes it a promising approach for real-world applications of silent speech interfaces.

Chapter 4. : The objective of this chapter is to improve the accuracy and robustness of ultrasound-based silent speech interfaces by integrating speaker information into the model to enhance the model's generalization capability for unseen speakers. To achieve this, a method is proposed for learning speaker embeddings using a deep neural network that can be utilized as a feature representation in the SSI model. The proposed approach involves a multi-task neural network that jointly learns speaker embeddings and SSI transcription from ultrasound data. The network includes a classification model that extracts features from the ultrasound data for each speaker and a speaker embedding network that maps the extracted features and ultrasound data to speech. For unseen speakers, the model predicts the features based on the already-trained model and then embeds these features into the unseen speaker's ultrasound data to forecast the speech features. The performance of the proposed method is evaluated on an English dataset with one speaker and multiple speakers, and the results demonstrate that integrating speaker embeddings improves the accuracy and robustness of the SSI model, especially for speaker-independent models.

In **Chapter 5.**, the challenge of preprocessing in Silent Speech Interfaces addressed by implementing a new Voice Activity Detection (VAD) method. Previous literature removed the silent parts of the speech by applying VAD on the whole utterance before preparing spectrogram as ground truth features. However, this implementation caused some misalignment between the spectrogram and ultrasound frames as the decreased signal was used as output data, which could not align with the ultrasound frames properly. To address this issue, we suggest splitting each utterance with time equal to the ultrasound time duration and implementing VAD on each speech frame instead of the whole speech. By doing this, we remove the respective silence part from both the ultrasound and the speech signal, avoiding misalignment issues.

To measure the performance of the detected frames as voice or silent, we classified the ultrasound labeled by VAD as silent or speech. The results were promising, indicating that the extracted features were useful in achieving better performance in the model. Additionally, we compared the method of applying silence removing to the whole speech versus in the frame level, and the results showed that the latter method was more effective in improving the model's performance. In conclusion, implementing VAD on each speech frame instead of the whole speech improved the accuracy of the SSI model, resulting in better alignment between spectrogram and ultrasound frames, leading to more reliable features for the model.

In **Chapter 6.**, we present a novel approach to deal with the challenge of high computational cost in deep learning models, which requires large amounts of data to achieve good performance. To address this issue, we suggest using a combination of Convolutional Neural Networks and Recurrent Neural Networks called ConvLSTM, along with a 3D Convolutional Network that can extract the sequential and volumetric information from the data.

By combining these two models, we aim to decrease the number of layers and parameters needed to train the model while still achieving high performance. The proposed method is evaluated on a Hungarian dataset and compared against previous high-performance models.

The chapter explains the implementation of the ConvLSTM model and demonstrates its ability to extract spatial and temporal features from ultrasound tongue videos. The results show that the proposed method outperforms the previous state-of-the-art models, achieving better accuracy and efficiency.

Overall, the method presents a promising approach to improving the performance of deep learning models while reducing the computational cost, which could be valuable for future research in the field of image processing and deep learning.

Chapter 7., focuses on addressing the challenges of speaker-specific models and poor performance in cross-session scenarios. We proposed the integration of a spatial transformer network (STN) module into deep networks to enable quick speaker and session adaptation of ultrasound tongue imaging-based silent speech interface (SSI) models.

The chapter begins by highlighting the sensitivity of current tongue ultrasound-based SSI systems to changing speakers and slight displacements of the recording device. These factors hinder quick switches between users and lead to poor performance when the recording equipment is dismounted and re-mounted. To mitigate these issues, we extend our previous deep networks with an STN module capable of performing an affine transformation on the input images.

We conducted experiments to evaluate the effectiveness of adapting the STN module and the linear output layer for spectral estimation. The results showed that allowing only the adaptation of the STN module could reduce the error rate. The experiments were performed on both 2D and 3D input blocks, with similar tendencies observed in both cases.

Considering that the STN module and the output layer together comprise only 10% of the weights of the full network, and the improvements were consistent for cross-session and cross-speaker setups, the proposed method offers a promising approach for quicker adaptation of the ultrasound tongue imaging (UTI)-processing workflow.

Overall, Chapter 7 demonstrates the potential of the STN module in addressing the challenges of speaker-specific models and poor cross-session performance in tongue ultrasound-based SSI systems. By incorporating the STN module, the proposed method allows for quicker adaptation and shows significant improvements in reducing error rates, making it a valuable contribution to the field of silent speech interfaces.

Chapter 2

3D Convolutional Neural Networks for Developing Silent Speech Interfaces Utilizing Ultrasound

Over the past few years, there has been a growing interest in converting articulatory movements into acoustic signals. In Chapter 1 we discussed the increasing interest in converting articulatory movements into acoustic signals to create ”Silent Speech Interfaces” (SSI). These interfaces record soundless articulatory movements and use them to automatically generate speech, making them useful for those who are speaking-impaired but can still move their articulators. SSIs can also be used in noisy environments or military applications.

The standard way to transform movement recordings into speech is through a two-step process involving recognition and synthesis, as described in [88]. First, a speech recognition system converts the biosignal into text, which is then used by a text-to-speech synthesis system to generate speech. However, this method has some drawbacks, including a significant delay between input and output and errors introduced by the speech recognizer. Furthermore, this approach results in the loss of prosodic information that can be estimated from the articulatory signal, such as energy and pitch, as noted in [31].

More recent systems prefer direct synthesis, where speech is generated directly from articulatory data without an intermediate step. As deep neural networks have become increasingly dominant in speech technology, recent studies have focused on using deep learning to solve the articulatory-to-acoustic conversion problem, regardless of the recording technique employed [15, 25, 31, 45, 47, 53]. DNN technology has been applied to various aspects of speech technology, including speech recognition [33], speech synthesis [62], and language modeling [111].

It appears that the main challenge leading to poor results in the literature’s implementation of SSI models was the failure to consider the type of data being used to implement the actual DNN model for extracting better features. Many of these studies utilized DNN technology for articulatory-to-acoustic conversion without regard to the recording technique used. Consequently, the results were not satisfactory, indicating the need to explore more effective strategies that account for the type of data being used in the SSI model.

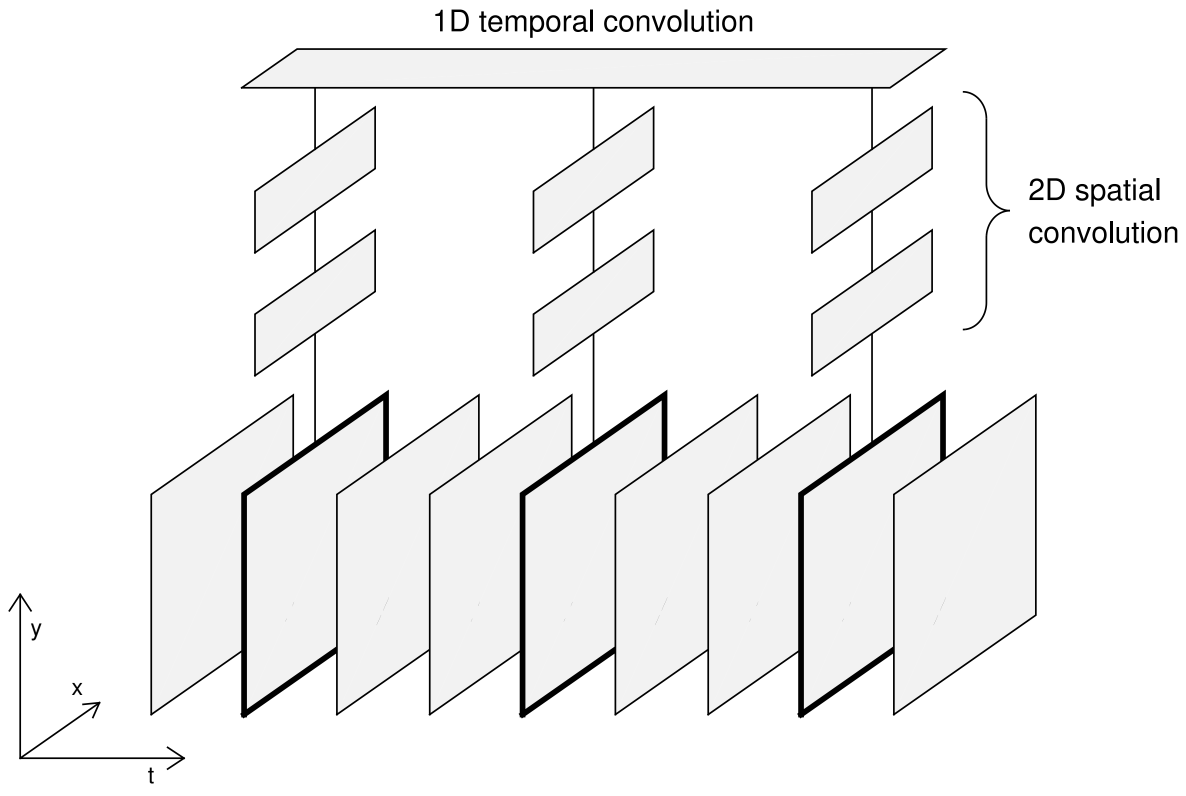
2.1 Introduction

In this dissertation, we utilize deep neural networks to convert ultrasound video of tongue movements into speech. Previous studies used fully connected neural networks for this purpose, but as we were working with a sequence of images, we applied convolutional neural networks that are popular for image recognition [15, 45, 47, 53, 56]. Our input here was a video sequence that contains time trajectory information of tongue movements. We explore different network structures for processing time sequences, including the stacking of a 2D CNN and a recurrent neural network like long short-term memory and extending the 2D CNN to 3D by adding time as an extra dimension [25, 48, 52, 53, 63, 108]. We follow the latter approach and investigate the applicability of a special 3D CNN model called the (2+1)D CNN [104] for ultrasound-based direct speech synthesis. Our finding suggests that our 3D CNN model achieves a lower error rate, is smaller, and has a faster training than the CNN+LSTM model. Therefore, for ultrasound video-based speech synthesis interface systems, 3D CNNs are a feasible alternative to a recurrent neural model.

2.2 Convolutional Neural Networks for Video Processing

Deep neural networks are a type of artificial neural network that use multiple layers of processing nodes to learn and represent complex relationships between inputs and outputs. Convolutional neural networks are a specific type of DNN that are particularly effective for image recognition tasks. Ever since the invention of 'Alexnet', CNNs have remained the leading technology in the recognition of still images [56]. These standard CNNs apply the convolution along the two spatial axes, that is, in two dimensions (2D). However, there are several tasks where the input is a video, and handling the video as a sequence (instead of simply processing separate frames) is vital for obtaining good recognition results. The best example is human gait recognition, but we can talk about action recognition in general [48, 119, 120]. In these cases, the sequence of video frames forms a three-dimensional data array, with the temporal axis being the third dimension in addition to the two spatial dimention (see Fig 2.1).

Figure 2.1: *Illustration of how the (2+1)D CNN operates. The video frames (at the bottom) are first processed by layers that perform 2D spatial convolution, then their outputs are combined by 1D temporal convolution. The model is allowed to skip video frames by changing the stride parameter of the temporal convolution.*



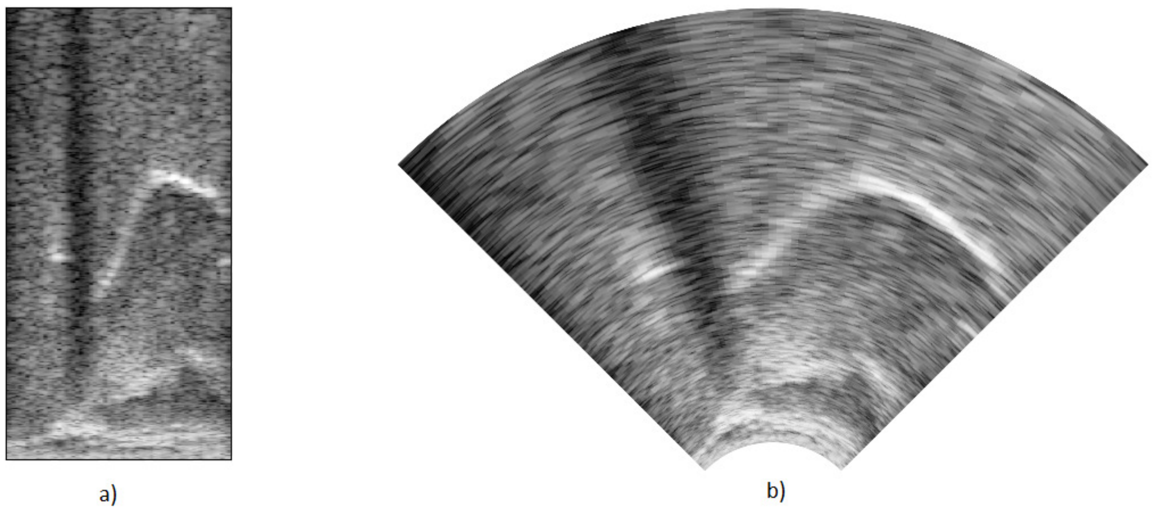
In the context of ultrasound-based direct speech synthesis, previous studies have applied CNNs to convert ultrasound video of tongue movements into speech. However, these studies did not always consider the type of input data, and simply implemented CNN models without adjusting them to the specific characteristics of the ultrasound data. Additionally, the ground truth labels used in these studies, such as frames of spectrograms, may have contained some information from previous or later frames.

For the processing of sequences, recurrent neural structures such as the LSTM are the most powerful tool [34]. However, the training of these networks is known to be slow and problematic, which led to the invention of simplified models, such as the gated recurrent unit (GRU) [13] or the quasi-recurrent neural network [11]. Alternatively, several convolutional network structures have been proposed that handle time sequences without recurrent connections. In speech recognition, time-delay neural networks (TDNN) have proved very successful [74, 101], but we can also mention the feedforward sequential memory network [115]. As regards video processing, several modified CNN structures have been proposed to handle the temporal sequence of video frames [48, 119, 120]. Unfortunately, the standard 2D convolution may be extended to 3D in many possible ways, giving a lot of choices for optimization. Tran et al. performed an experimental comparison of several 3D variants, and they got the best results when they decomposed the spatial and temporal convolution steps [104]. The model they called '(2+1)D convolution' first performs a 2D convolution along the spatial axes, and then a 1D convolution along the time axis (see Fig. 2.1). By changing the stride parameter of the 1D convolution, the model can skip several video frames, thus covering a wider time context without increasing the number of processed frames. Interestingly, a very similar network structure proved very efficient in speech recognition as well [101]. This model allows for flexibility in window length and overlapping sliding windows, which allows for more precise extraction of input data. Stacking several such processing blocks on top of each other is also possible, resulting in a very deep network [104]. Here, we are going to experiment with a similar (2+1)D network structure for ultrasound-based SSI systems. The performance of the new model was compared to other CNN-based models, such as (CNN+BiLSTM), (CNN+LSTM), 2D CNN, and FCN (for the details, see sec. 2.4).

2.3 Data Acquisition and Signal Preprocessing

The ultrasound recordings were collected from a Hungarian female subject (42 years old, with normal speaking abilities) while she was reading sentences aloud. Her tongue movement was recorded in a midsagittal orientation – placing the ultrasonic imaging probe under the jaw – using a "Micro" ultrasound system by Articulate Instruments Ltd. The transducer was fixed using a stabilization headset. The 2-4 Mhz / 64 element 20 mm radius convex ultrasound transducer produced 82 images per second. The speech signal was recorded in parallel with an Audio-Technica ATR 3350 omnidirectional condenser microphone placed at a distance of 20 cm from the lips. The ultrasound and the audio signals were synchronized using the software tool provided with the equipment. Altogether 438 sentences (approximately half an hour) were recorded from the subject, which was divided into train, development and test sets in a 310-41-87 ratio. We should add that the same dataset was used in several earlier studies [15, 31]. The ultrasound probe records 946 samples along each of its 64 scan lines. The recorded data can be converted to conventional ultrasound images using the software tools provided. However, due to its irregular shape, this image is harder to process by computers, while it contains no extra information compared to the original scan data. Hence, we worked with the original 964x64 data items, which were downsampled to 128x64 pixels. Fig. 2.2 shows an example of the data samples arranged as a rectangular image, and the standard ultrasound-style display generated from it. The intensity range of the data was min-max normalized to the [-1, 1] interval before feeding it to the network.

Figure 2.2: *Example of displaying the ultrasound recordings as a) a rectangular image of raw data samples b) an anatomically correct image, obtained by interpolation.*



The speech signal was recorded with a sampling rate of 11025 Hz, and then processed by a vocoder from the SPTK toolkit (http://sp-tk.sourceforge.net). The vocoder represented the speech signals by 12 Mel-Generalized Cepstral Coefficients (MFGG) converted to a Line Spectral Pair representation (LSP), with the signal's gain being the 13th parameter. These 13 coefficients served as the training targets in the DNN modeling experiments, as the speech signal can be reasonably well reconstructed from these parameters. Although perfect reconstruction would require the estimation of the pitch (F0 parameter) as well, here we ignored this component during the experiments. To facilitate training, each of the 13 targets were standardized to zero mean and unit variance.

2.4 Experimental Setup

We implemented our neural networks using the Keras framework with a Tensorflow backend [22]. We created five different models: a fully connected network that processes a single frame, a 2D convolutional network that still operates on a single frame (2D CNN), a 3D convolutional network that takes a sequence of frames as input (3D CNN), a network that combines the results of the 2D CNN layers processing the frames using an LSTM (CNN+LSTM) for comparison, and a bidirectional version of the same network (CNN+BiLSTM). To ensure that they were comparable in terms of the number of parameters, we set each network to have approximately 4.3 million trainable parameters. We used the Adam optimizer with a batch size of 100 for training. The mean squared error was used as the training loss function.

2D CNN	3D CNN
Conv2D(30, (13,13), strides=(2,2))	Conv**3D**(30, (**5**,13,13), strides=(**s**, 2,2))
Dropout(0.2)	Dropout(0.2)
Conv2D(60, (13,13), strides=(2,2))	Conv**3D**(60, (**1**,13,13), strides=(**1**,2,2))
Dropout(0.2)	Dropout(0.2)
MaxPooling2D(pool_size=(2,2))	MaxPooling**3D**(pool_size=(**1**,2,2))
Conv2D(90, (13,13), strides=(2,1))	Conv**3D**(90, (**1**,13,13), strides=(**1**,2,1))
Dropout(0.2)	Dropout(0.2)
Conv2D(150, (13,13), strides=(2,2))	Conv**3D**(**85**, (**1**,13,13), strides=(**1**,2,2))
Dropout(0.2)	Dropout(0.2)
MaxPooling2D(pool_size=(2,2))	MaxPooling**3D**(pool_size=(**1**,2,2))
Flatten()	Flatten()
Dense(1000)	Dense(1000)
Dropout(0.2)	Dropout(0.2)
Dense(13, activation='linear')	Dense(13, activation='linear')

Table 2.1: *The layers of the 2D and 3D CNNs in the Keras implementation, along with their most important parameters. The differences are highlighted in bold.*

Fully Connected Network (FCN): The simplest network structure is achieved when using fully connected (in Keras, ”Dense”) layers. In our case, the network was built with five hidden layers, each containing 430 neurons, while the output layer consisted of 80 linear activation neurons, corresponding to the Mel-spectral target vector. The network input is a single frame, i.e., 8192 pixels (128x64). The hidden layers used the Swish activation function [79], and after each hidden layer, a dropout layer with a dropout probability of 0.2 followed.

Convolutional Network (2D CNN): Similar to the previous network, this network also processes a single frame, however, each of the four layers below the top Dense layer performs spatial convolution on the data. The detailed configuration of the layers can be seen in Table 2.1, We searched for the best meta-parameters through experimentation, and for the hidden layers also used the swish activation function in this case.

3D Convolutional Network (3D CNN): In this network, instead of 2D convolution, we used 3D convolution, which allows for processing a short sequence of images instead of a single image. The specific network structure (see Table 2.1) processes five images that are s distance apart from each other, where s is the stride parameter along the temporal axis of the convolution. Following the (2+1)D convolution concept presented in Section 2.2, the network first processes each of the five images separately, and then performs the temporal convolutional step. In Table 2.1), we highlighted in bold the modifications that were necessary compared to the original 2D CNN network. Note that reducing the size of the top convolutional layer was necessary to keep the number of parameters in the two networks approximately the same.

Convolutional LSTM Neural Network (CNN+LSTM): The fully connected and 2D CNN networks process a single frame, so it can be assumed that they will not be worthy competitors for the 3D CNN network operating on a sequence of images. For processing sequences, the LSTM network is recommended, and since it is a sequence of images, it is worth combining the feedback network with the 2D CNN network that processes the images. The bottom four layers of the CNN+LSTM model we used were the same as the bottom four layers of the 2D CNN network; however, we replaced the top hidden layer with a Dense layer that was switched to an LSTM layer. To preserve the number of parameters, we added 500 LSTM neurons to this layer, and the input to the network consisted of 21 consecutive frames.

Convolutional Bidirection LSTM Neural Network(CNN+BiLSTM): If we do not insist on real-time processing, the LSTM layer can operate not only in a forward direction (from left to right) but also in a backward direction (from right to left) in time. It is also common to create a forward and a backward layer and combine their outputs. Our model called CNN+BiLSTM is such a bidirectional network, which differs from the previous CNN+LSTM model only in the bidirectionalization of the LSTM layer. To preserve the number of parameters, we had to reduce the size of the LSTM layer to 320 in this model.

2.5 Results

There are several options for evaluating the performance of our networks. In the simplest case, we can compare their performance by simple objective metrics, such as the value of the target function optimized during training (the MSE function in our case). Unfortunately, these metrics do not perfectly correlate with the users' subjective sense of quality of the synthesized speech. Hence, many authors apply subjective listening tests such as the MUSHRA method [50]. This kind of evaluation is tedious, as it requires averaging the scores of a lot of human subjects. As an interesting shortcut, Kimura et al. applied a set of commercial speech recognizers to substitute the human listeners in the listening tests [53]. In this paper, we will simply apply objective measures, namely the mean squared error and the (mean) R^2 score, which are simple and popular methods for evaluating the performance of neural networks on regression tasks.
As we have seen in the theoretical overview, the 3D convolutional network has a very important meta-parameter, the s parameter. In the first experiment, we examined its effect on the value of the error function. This parameter determines how much

information the network receives about the input time period: the distance between the two outer frames can be determined by the formula $w = 4 * s + 1$. For example, with $s = 5$, the size of the time frame covered by the network is $w = 21$ frames. Considering the video sampling rate of 82 frames/sec, this corresponds to approximately a quarter of a second, roughly half the length of a syllable.

Figure 2.3: *MSE rates of the 3D CNN on the development set for various s stride values. For comparison, the MSE attained by the 2D CNN is also shown (leftmost column).*

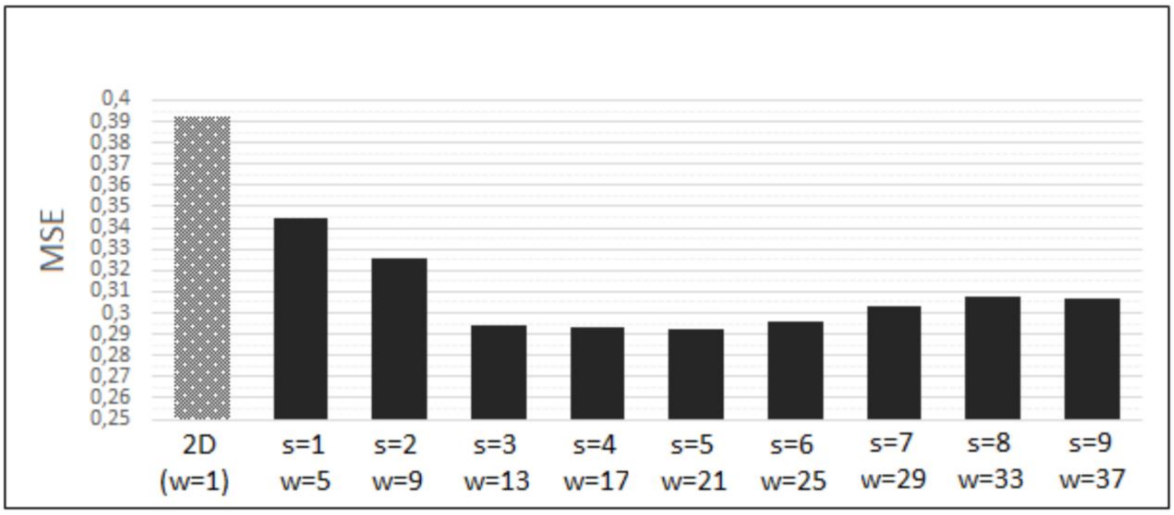


Fig. 2.3 shows the MSE values obtained with the 3D CNN network for different values of the s parameter. For comparison, we also included the error of the 2D CNN network (which only processes a single frame). It can be seen that by taking into account not only the current frame but also its neighborhood, significant error reduction can be achieved. Even by using only 2 immediate neighbors (s=1), we obtain better results, but the improvement is much greater with s values between 3 and 6. Therefore, considering a wider context is important, even at the cost of skipping frames. Based on the above result, we fixed the value of the s parameter to 5.

In the next experiment, we compared five different network structures, summarizing the MSE and R^2 values obtained on the validation and test sets in Table 2.2 (in the case of R^2, a higher value indicates a better model). It can be seen that, among the FC and 2D CNN networks that process a single frame, the convolutional network is clearly better, but much better results can be achieved with network variants that process sequences of frames instead of a single frame (3D CNN and CNN+LSTM networks).

Network Type	Val		Test	
	MSE	Mean R^2	MSE	Mean R^2
FCN	0.410	0.600	0.419	0.585
2D CNN	0.392	0.617	0.401	0.603
3D CNN (s=5)	0.292	0.714	0.293	0.710
CNN + LSTM [50]	0.303	0.701	0.269	0.709
CNN + BiLSTM	0.301	0.706	0.296	0.707

Table 2.2: *MSE and R^2 values obtained with different network architectures on the validation and test sets.*

To compare the 3D CNN and LSTM-based networks, we set the input of the LSTM networks to 21 frames, as this proved to be optimal for the 3D CNN. As shown in table 2.1, the CNN+LSTM model clearly outperformed the single-frame models, but could not surpass the accuracy of the 3D CNN network. Even the bidirectional LSTM layer did not change this: while it usually brings a small improvement in other tasks, here we got essentially the same result as with the unidirectional network. We considered the possibility that the optimal window size might be different for CNN+LSTM models, so we tried to change the 21-frame input size, but we did not get significantly better results with other values. Based on the obtained error values, it seems that subsampling the frames is just as effective in helping information fusion as processing all frames with the LSTM's more sophisticated feedback and internal memory techniques. However, because of its recurrent nature, the LSTM cannot skip frames, although this might be necessary in this case. Due to the preservation of all frames, the training of the CNN+LSTM network took about 70% longer than the training of the 3D CNN network, despite having the same number of parameters. Interestingly, the memory requirements of the LSTM network were also higher, presumably due to the preservation of all input frames.

The authors of [15] applied a fully connected network on the same data set. They obtained slightly better results than those given by our FCN, presumably due to the fact that their network had about 4 times as many parameters. More interestingly, they attempted to include more neighboring frames in the processing, simply by concatenating the corresponding image data. Feature selection was applied to alleviate the problems caused by the large size of the images (~ 8000 pixel per image), these simple methods failed to significantly reduce the error rate. Our current experiments show that the frames should be placed farther apart ($3 \leq s \leq 8$) for optimal performance. Moreover, instead of reducing the input size by feature selection, it seems to be more efficient to send the frames through several neural layers, with a relatively narrow 'bottleneck' layer on top.

Finally, we note that Moliner and Csapó also attempted to combine 2D CNN and LSTM networks on a similar task [50]. However, their results are not directly comparable to ours because they used a different model and therefore used different target values during training. Additionally, the network they used was much larger, with more than four times the number of parameters compared to our networks. They also did not observe a significant difference in performance between unidirectional and bidirectional LSTM variations. Parallel to our work, Saha and colleagues also attempted to process language-audio-video and identified a similar 3D convolutional network structure as optimal. They achieved significantly better results with the 3D CNN network than with the CNN+LSTM combination [85].

	STOI	PESQ	MCD
FCN	0,661	1,562	4,602
2D CNN	0,658	1,551	4,569
3D CNN (s=5)	0,743	1,831	4,161
CNN + LSTM	0,742	1,792	4,139
CNN + BiLSTM	0,736	1,789	4,152

Table 2.3: *Comparison of our five models with objective quality metrics for speech.*

Various objective metrics have been proposed for evaluating sound quality. These attempt to take into account the main characteristics of human hearing, and therefore provide a more accurate estimate of sound quality than the MSE error function optimized during training. In Table 2.3 above, we evaluated three such metrics on the five test sets synthesized by the models: STOI (Short-Term Objective Intelligibility, [97]), PESQ (Perceptual Evaluation of Speech Quality,[2]), and MCD (Mel-Cepstral Distance, [57]). Higher values indicate better quality for the first two, while lower values indicate better quality for the last one. The numbers obtained show a clear qualitative leap between the models that process only one frame (FC and 2D CNN) and those that convert sequences of frames (3D CNN, LSTM, and BiLSTM). Two metrics favored the 3D CNN, and one favored the LSTM, but the difference between their performance according to all three metrics was minimal, so we cannot declare a clear winner.

2.6 Summary

Based on the better results achieved in the experiments, we have decided to choose 3D CNN as the preferred implementation for later model improvements. The 3D CNN network performed equivalently or slightly better than the CNN+LSTM network, while requiring less training time. While both models have many meta-parameters, further in-depth measurements would be required to claim the superiority of the 3D CNN. However, we can conclude that the 3D CNN network is a competitive alternative to combined CNN+LSTM networks when building a silent speech interface based on visual speech input. Therefore, this method will be preferred for future improvements to the model.

The author of this PhD thesis has contributed in two main aspects presented in this chapter:

II/1. Implementing the neural networks used in the experiments to restore speech signals from articulatory recordings. Specifically, they implemented a 3D convolutional neural network with different window length and compared it with different variations of CNN and combination with CNN+LSTM and BiLSTM networks.

II/2. Calculating the performance of the models using objective metrics such as STOI, PESQ, and MCD. The results obtained from these metrics were analyzed to compare the performance of the different network architectures.

It should be noted that the results of this chapter have been published in two different publications, indicating the relevance and significance of the research [60, 102].

Chapter 3

Utilizing adversarial training to improve Deep Neural Network models

In deep neural network based ultrasound image SSI projects, conventional loss functions for 3D CNNs (3D convolutional neural networks) are often used to optimize the performance of the models. These loss functions can be simple mathematical formulas that measure the difference between the predicted output of the model and the actual ground truth. The most commonly used loss function for image processing tasks is the mean squared error.

However, conventional loss functions for 3D CNNs in ultrasound image SSI projects face several challenges. For example, conventional loss functions do not always capture the perceptual quality of the output spectrogram images. Additionally, conventional loss functions may result in blurry spectrogram images or spectrogram images with loss of fine details. Therefore, there is a need to use alternative loss functions that can capture the perceptual quality of the spectrogram images.

Perceptual loss functions are one type of alternative loss functions that can improve the performance of ultrasound image SSI models. Perceptual loss functions are originally designed to measure the similarity between two images in terms of their perceptual qualities, such as texture, contrast, and sharpness, instead of just measuring the pixel-wise differences between the images. Perceptual loss functions are commonly used in image style transfer and image generation tasks.

Adversarial networks, such as Generative Adversarial Network (GAN), can also be used to improve the conventional loss function for ultrasound image SSI models. GANs consist of two neural networks, a generator network and a discriminator network, that are trained together in a game-like setting. The generator network generates images that try to mimic the ground truth images, while the discriminator network tries to differentiate between the generated images and the real ground truth images. Through this process, the generator network learns to produce images that are more realistic and similar to the ground truth images.

In the context of ultrasound image SSI projects, adversarial networks can be used to improve the perceptual quality of the data predicted by the model. By using adversarial training, the SSI model can learn to generate spectrogram images that are not only pixel-wise similar to the ground truth spectrogram images, but also have better perceptual quality, such as sharper edges, better contrast, and more realistic texture.

In the experimental part, we used adversarial networks to improve the conventional loss function for ultrasound image SSI models. The experiments were run on two Hungarian and English datasets, and different evaluation metrics were used to evaluate the results. The results of the experiments showed that using adversarial networks can improve the perceptual quality of the images, as measured by the evaluation metrics.

3.1 Problem Description and Literature Overview

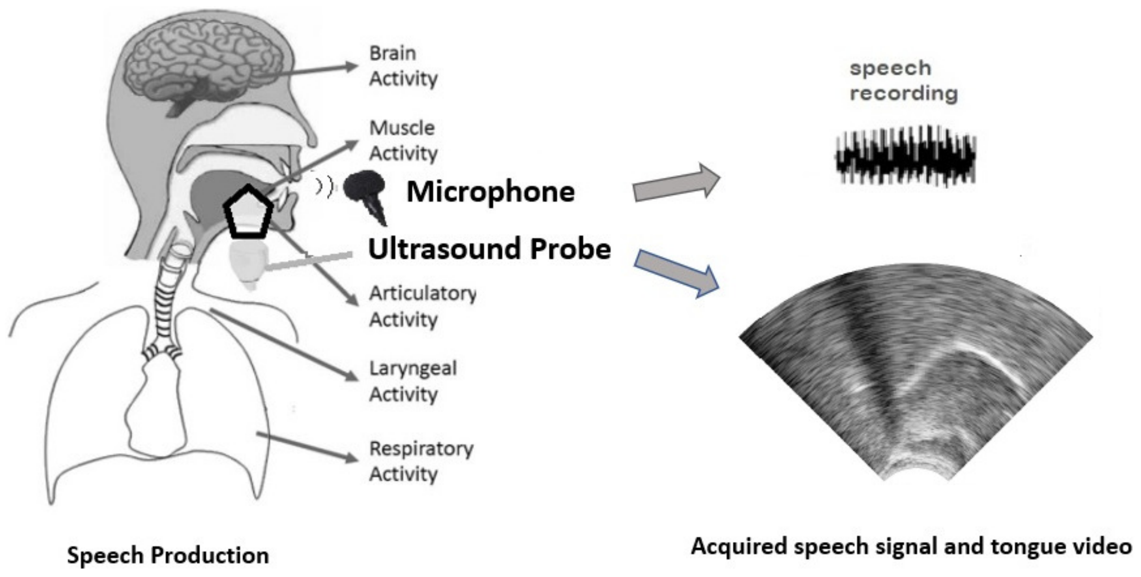
As we discussed in Chapter1, the process of human speech production involves complex motor processes that include speech planning in the brain, respiratory, laryngeal, and articulatory movements. Electromagnetic Articulography, Ultrasound Tongue Imaging, Permanent Magnetic Articulography, and surface electromyography are methods used to record articulatory movements. Silent Speech Interfaces reconstruct the speech signal from these signals, and their applications include aiding impaired individuals and situations where speaking loudly is not feasible. Brain-computer interfaces like functional Magnetic Resonance Imaging or functional functional near-infrared spectroscopy (fNIRS) can be used to record brain activity in the cases of imagined speech and inner speech where no articulatory movement is involved.

There are two approaches to articulatory-to-acoustic conversion, including the two-step approach, which first converts the signal to text and then to speech [23, 40, 52, 106], and the direct approach, which converts the articulatory signal to a speech signal without any intermediate steps. The direct approach, which typically uses a deep neural network, has become more popular with the development of powerful deep learning techniques [25, 45, 86, 98]. As we follow the second approach using ultrasound tongue imaging videos as the input, and we will experiment with a special DNN configuration to perform the conversion.

3.2 Silent Speech Interfaces Built on Ultrasound Tongue Imaging and Neural Vocoders

Here, we worked with an ultrasound-based data acquisition technique. In this framework, an ultrasound probe is placed under the chin of the subject, so it can record images of the tongue movement in a midsagittal orientation (see Fig 3.1). The device applied was the ”Micro” ultrasound system produced by Articulate Instruments Ltd. To support DNN training, the subjects were asked to speak loud, and their speech was recorded in parallel with the ultrasound output (cf. Fig 3.1 again). The ultrasound and speech recordings served as the training inputs and training targets for the DNN, respectively. As the input and target signals were synchronized, we applied conventional network structures which perform the conversion in a time-synchronized, frame-by-frame manner.

Figure 3.1: *Illustration of the data acquisition process.*



Training a network to generate speech signals directly would require a huge amount of training data. Hence, we opted for a two-step solution, which was motivated by speech synthesis. In text-to-speech systems, a popular approach is to first convert the text to a spectrogram, then convert the spectrogram to a speech signal. Nowadays, both steps can be implemented by neural networks [75]. To adjust this approach to our task, only the first network needs to be modified, as our input is an ultrasound video and not a text. This way, we can apply large, pre-trained networks for the second task, while our network has to estimate only a dense spectral representation instead of the speech waveform itself. Several neural vocoders are available for the speech generation task [29], and we chose to use the WaveGlow model [75]. As WaveGlow requires a sequence of 80-dimensional mel-scaled spectral vectors as input, the task of our network was to estimate such a spectral vector form each frame of the video. The next section discusses the details of this step.

3.3 Generative Adversarial Networks

Generative Adversarial Network is a type of neural network that consists of two networks: a generator network and a discriminator network. The generator network is responsible for generating synthetic data while the discriminator network's job is to distinguish between the synthetic data and the real data. The two networks are trained together in a process known as adversarial training, where the generator tries to produce more realistic data, and the discriminator tries to improve its ability to differentiate between the real and fake data [27]. This system is called adversarial, as the generator and the discriminator work against each other.

Figure 3.2: *A typical GAN implementation consists of two neural networks (generator and discriminator) trained in an adversarial manner, with noise vectors as input to the generator and real/fake labels as input to the discriminator.*

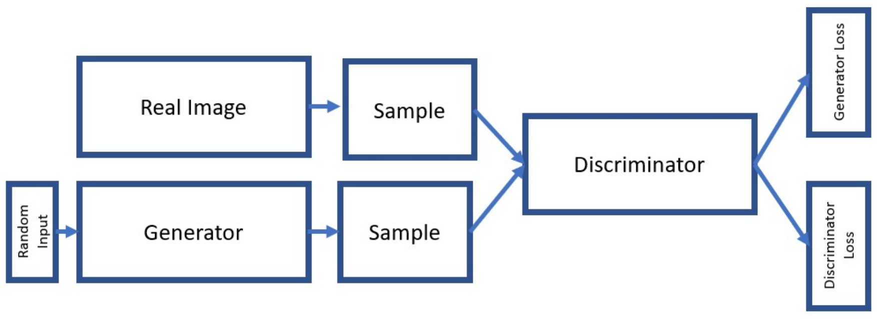


Figure 3.2 shows the implementation of a GAN (for image generation). It consists of two main components, the generator network and the discriminator network. The generator network takes a random noise vector as input and generates a synthetic image as output. The discriminator network takes the synthetic image generated by the generator network or a real image from the dataset as input and outputs a probability value that indicates whether the input image is real or fake. The two networks are trained in an adversarial manner, where the generator tries to generate more realistic images to fool the discriminator, and the discriminator tries to improve its ability to differentiate between the real and fake images. During the training process, the generator learns to generate more realistic images, and the discriminator learns to accurately classify the images as real or fake. Once the training is complete,

the generator can be used to generate new images that are similar to the real images in the dataset. This approach has been used in a wide range of applications, including image generation, speech synthesis, and natural language processing.

3.3.1 Conditional Generative Adversarial Networks

Conditional Generative Adversarial Networks (CGANs) are a type of GAN that add additional information (see fig 3.3), or conditions, to the generator and discriminator [68]. This extra information can be in the form of labels, attributes, or any other relevant data that can be used to guide the generation process, as it can be seen as y vector in fig3.3. CGANs have been successfully applied to a variety of tasks, such as image-to-image translation, text-to-image generation, and style transfer.

Figure 3.3: *Conditional Generative Adversarial Networks [68]*

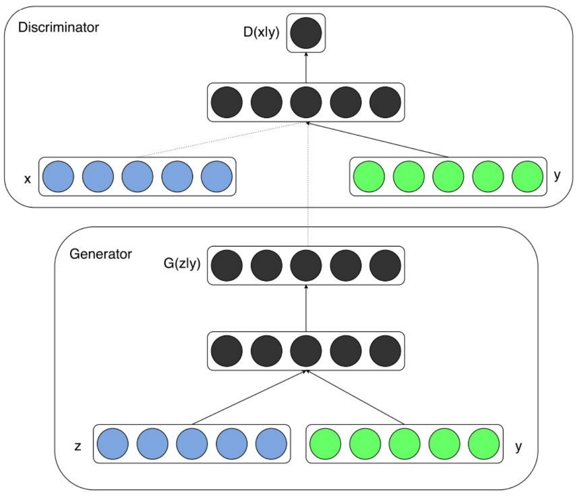


Image-to-image translation is a popular application of cGANs, where the goal is to translate an image from one domain to another [42]. For example, translating a grayscale image to a colored image, or translating a sketch to a realistic image. cGANs can also be used for text-to-image generation, where the goal is to generate an image from a textual description.

Another popular application of cGANs is style transfer, which involves transferring the style of one image onto another image while preserving the content. For example, taking a photograph and applying the style of a famous painting to create a new, stylized image.

Overall, cGANs are a powerful extension of GANs that allow for more precise control over the generation process and have many practical applications in fields such as computer vision, natural language processing, and creative arts.

3.4 Generative Adversarial Networks for Articulatory-to-Acoustic Mapping

As we use WaveGlow for the speech synthesis step, the task for our DNN was to convert each ultrasound video frame (64x128 pixels) into a mel-spectral vector (with 80 components). In chapter 2 we found that the best results can be achieved by using a short sequence of video frames as input, rather than just a single frame [102]. We got the lowest error rate with a special 3D convolutional neural network structure, and other authors reported similar findings [86]. Here, we apply the same 3D CNN structure that was found the best in chapter 2. The input of this network consists of a 25-frame block of the ultrasound video, and the output is a mel-spectral vector.

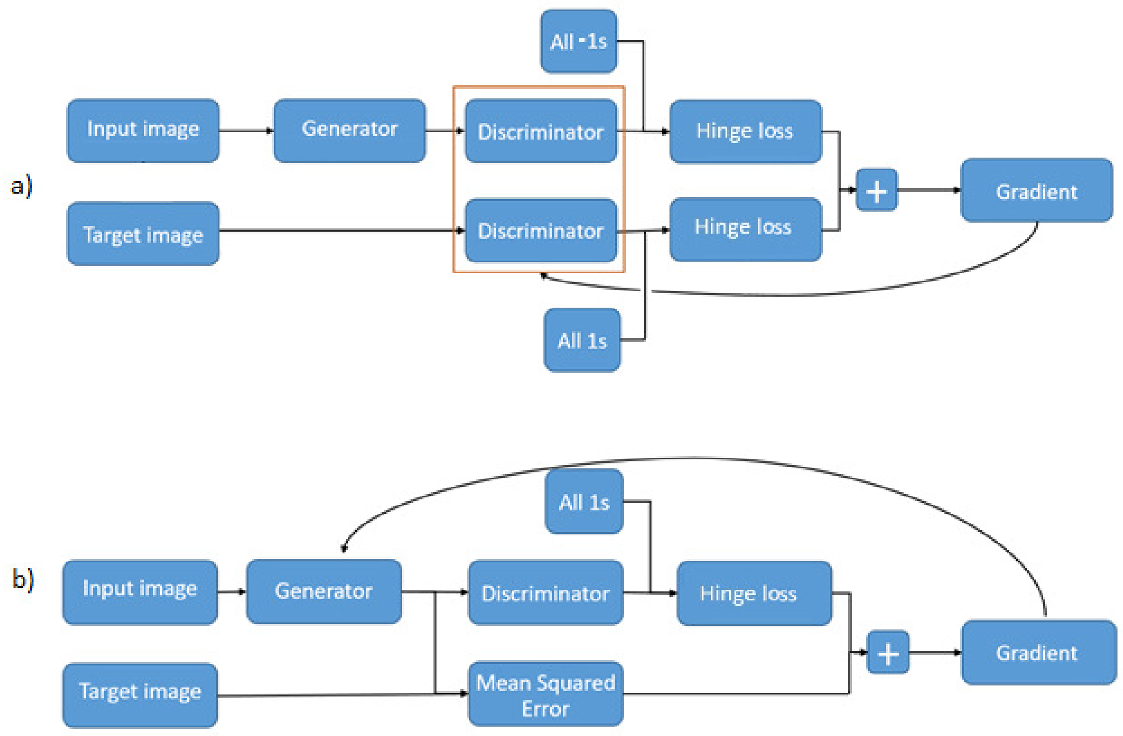
A crucial parameter of DNN training is the loss function which formalizes the difference between the training targets and the DNN output. In regression tasks, the simplest solution is to use general loss functions such as the mean-squared error. However, such mathematically motivated loss functions may not conform with our perception. When the network output is a speech signal, such as in speech enhancement, one may try to apply objective speech quality metrics as the loss function [66, 121]. Here, we explore another method where a second, discriminator network is trained to judge the quality of the output signal. This approach leads to the subject of Generative Adversarial Networks.

As we discussed in section 3.3.1 when the goal is not data generation but data transformation, we can modify the original GAN formulas by adding the input to be transformed as a 'condition' of the data distribution, and hence this model is called the conditional GAN or CGAN [68]. GANs and CGANs were shows to produce very good quality images in broad types of vision tasks like image generation [27], image translation [42, 122], and text-to-image synthesis [80]. In speech technology, the GAN-based approach is the most successful in speech enhancement [116] and voice conversion [51].

In our case, the role of the generator was played by the network that converts the ultrasound data to mel-spectral data. For this purpose we used the same network as in chapter 2. As the baseline system, we trained this network conventionally, using the MSE loss function. Then we created a discriminator which was trained to discriminate real mel-spectrograms from those produced by the generator. As was explained earlier, the generator was trained in an adversarial manner, so it was optimized to create spectrograms that cannot be discriminated from real spectrograms by the discriminator.

The generator and the discriminator was trained in parallel, using a two-step process [116]. As shown in Fig. 3.4a, the first step trains the discriminator on two types of images: on real spectrograms (lower data path), and on spectrograms created by the generator (upper path). These serve as positive and negative training examples,

Figure 3.4: *The error calculation (forward arrows) and weight update (backward arrows) training steps for the discriminator (upper image) and the generator (lower image) networks of the GAN.*



denoted by the target labels of 1 and -1 in the figure. For the training, we applied the hinge loss function, which is frequently used as a GAN objective [58], and optimized it using the Adam optimizer with a learning rate of 0.0002. In this step, only the weights of the discriminator are updated, while the generator weights are frozen.

Fig. 3.4b shows the other training step that updates the generator weights (now the discriminator weights are frozen). We combine two error functions to calculate the loss, and hence the gradient. First, we compare the generator output and the target spectrograms using the MSE loss (lower path). The second loss value is obtained from the discriminator (upper path). Notice that now the discriminator target labels are flipped, as we wish to train the generator in an adversarial manner, to create outputs that look like a real spectrogram.

3.5 Experimental Setup

The goal of the experiments was to compare the performance of the 3D CNN generator network trained conventionally, using the MSE loss function, with the training scheme that applies the GAN-style discriminator network. In the experiments we evaluated our models on two data sets – one of them being recorded from a Hungarian speaker and the other from an English speaker. Here, we shall present the technical details of the experiments.

3.5.1 Data Sets and Data Preprocessing

Hungarian Data Set: The parallel ultrasound and speech recordings were collected from a Hungarian female subject reading sentences aloud as far as we explain in chapter 2.3, using the equipment briefly described in Section 3.2. The whole duration of the recordings was about half an hour (438 sentences), from which 310 were used for training, 41 for development and 87 for testing, respectively.

The ultrasound transducer produces an ultrasound video of the tongue movement at a rate of 82 frames per second. One frame of this video has a resolution of 64x946; that is, the device collects 946 data samples along 64 scan lines. As these ultrasound images are very noisy and they contain very few details, we decreased the image size to 64x128 by applying a bicubic interpolation. The pixel intensities were min-max scaled to the [-1, 1] range.

The speech signals were recorded in parallel with the ultrasound video at a sampling rate of 22050 Hz. The speech and ultrasound signals were synchronized using the software tool provided by Articulate Instruments.

English Data Set: As the English data set, we used the TAL corpus [82]. It contains parallel speech, tongue ultrasound and lip video recordings from 81 speakers. Here, we just used the TAL1 subset of the corpus, which contains the recordings of a single trained native English speaker. The recording conditions were very similar to that of the Hungarian data set and, after division, the train, validation and test sets contained 1015, 50 and 24 utterances, respectively. We applied exactly the same preprocessing steps as for the Hungarian data.

3.5.2 DNN Configuration and Training

As the generator we used the 3D CNN from chapter 2, with the slight modification that instead of one target vector, here we specified 5 consecutive spectral vectors as targets. We hoped this modification would improve the performance of the discriminator.

As the discriminator, we applied the so-called Patch-GAN method [42, 122]. Instead of a two-class decision for the whole spectrogram image, the discriminator of

Layer	Filters	Size	Strides	Padding	Activation
Conv2D	64	(4,4)	(2,2)	same	relu
Conv2D	128	(4,4)	(2,2)	same	relu
Conv2D	256	(4,4)	(2,2)	same	relu
ZeroPadding2D	–	–	–	–	–
Conv2D	512	(2,2)	(1,1))	valid	relu
ZeroPadding2D	–	–	–	–	–
Conv2D	1	(4,4)	(1,1)	valid	tanh

Table 3.1: *The layers of the discriminator network (following Keras' terminology). Albeit not shown, each but the last Conv2D layer is followed by batch normalization.*

a patch-GAN returns a set of output values for different regions of the spectrogram. We applied a fully convolutional CNN for this purpose, with an output vector of 10 components. The actual network configuration and its parameters are shown in Table 3.1.

When synchronized training targets are available, as in our case, adversarial training can be combined with conventional MSE-training. We got the best results when the MSE loss function was combined with the adversarial training loss using a weighting ratio of 0.75-0.25 for the two loss functions, respectively.

Training Method	Hungarian Corpus				English Corpus			
	Dev		Test		Dev		Test	
	MSE	Mean R^2	MSE	Mean R^2	MSE	Mean R^2	MSE	Mean R^2
MSE	0.327	0.68	0.326	0.677	0.316	0.683	0.317	0.68
GAN	0.287	0.719	0.29	0.713	0.287	0.715	0.293	0.71

Table 3.2: *The MSE and R^2 scores of the generator with the MSE and GAN training approaches.*

3.6 Results

Evaluating the quality of the generator is not trivial, as its output is a spectrogram, which is converted to a speech signal by WaveGlow. Our main goal is to increase the quality of the synthesized speech, and we have two ways to measure this quality. First, we can perform subjective listening tests such as MUSHRA [43]. Unfortunately, this would require a lot of human subjects, hence it would be slow and troublesome, which we wished to avoid here. Instead, we can evaluate objective, formally defined metrics. As the simplest of these, we report two metrics that are popular in machine learning for regression tasks, namely the mean squared error and the correlation-based mean R^2 score (the former has to be minimized, while the latter should be maximized). As can be seen in Table 3.2, all these metrics gave a slight improvement when the GAN-style training was applied, both for the Hungarian and English subjects, and both for the development and the test sets.

As we mentioned in Section 3.2, our output is a speech signal, and in this case the above simple metrics may not perfectly reflect human perception. Fortunately, several objective measures have been developed in speech technology and in telecommunications to compare the quality or intelligibility of speech recordings. These measures include the Short-Time Objective Intelligibility (STOI) [66] and its extended version (ESTOI), the perceptual evaluation of speech quality (PESQ) method [66], and its extended version known as perceptual metric for speech quality evaluation (PMSQE) [66]. While the first three measure quality, and hence a higher value means a better performance, PMSQE was designed to be applicable as a loss function in DNN training, so in this case a lower value indicates better quality. We also evaluated the signal-to-distortion ratio (SDR) and its extended, scale invariant version (SI-SDR) [49]. As their name suggests, for these metrics a higher value means better quality. Finally, we calculated the mel-cepstral distortion (MCD) [66], which is a distortion metric, so it should be minimized to increase speech quality. These results are summarized in Table 3.3 for the test sets of the two databases. To aid readibility, the two metrics that are to be minimized were placed in the rightmost columns, and the best scores are highlighted for both corpora. As the results indicate, extending the MSE training criterion with GAN-style adversarial training led to a consistent improvement in all the evaluated metrics and for both corpora. Although in certain cases the improvement is only slight, the results clearly justify the utility of generative adversarial networks for the articulatory-to-acoustics mapping task. In comparison, Ribeiro et al. reported an MCD score of 2.99 for the English dataset using a much more sophisticated endocer-decoder neural architecture [82].

Corpus and Training method	**STOI**	**ESTOI**	**PESQ**	**SISDR**	**SDR**	**PMSQE**	**MCD**
Hun - MSE	0.7050	0.456	1.282	-39.363	-18.021	2.797	4.627
Hun - GAN	**0.7067**	**0.4673**	**1.311**	**-37.868**	**-16.893**	**2.777**	**4.558**
Eng - MSE	0.612	0.385	1.464	-36.176	-18.31	2.827	3.38
Eng - GAN	**0.623**	**0.405**	**1.503**	**-36.031**	**-17.672**	**2.735**	**3.229**

Table 3.3: *Objective quality scores for the two training methods and the two corpora.*

3.7 Summary

The application of the GAN framework has already proved successful in speech enhancement and voice conversion tasks, and here we made the first attempts to apply it to the articulatory-to-acoustic mapping task of ultrasound-based silent speech interfaces. As the baseline, we trained the generator network conventionally, using the MSE loss, and then we extended its training with adversarial training by means of a discriminator. We applied our method on two data sets, a Hungarian corpus and an English corpus, and in both cases we found that the quality of the generated speech signals improved, according to several objective speech quality metrics. Based on the results I presented, it was found that incorporating perceptual loss into an ultrasound SSI model could enhance its performance.

The author of this PhD thesis is responsible for the following contributions presented in this chapter:

III/1. Implementation of GAN and CGAN models for image generation.

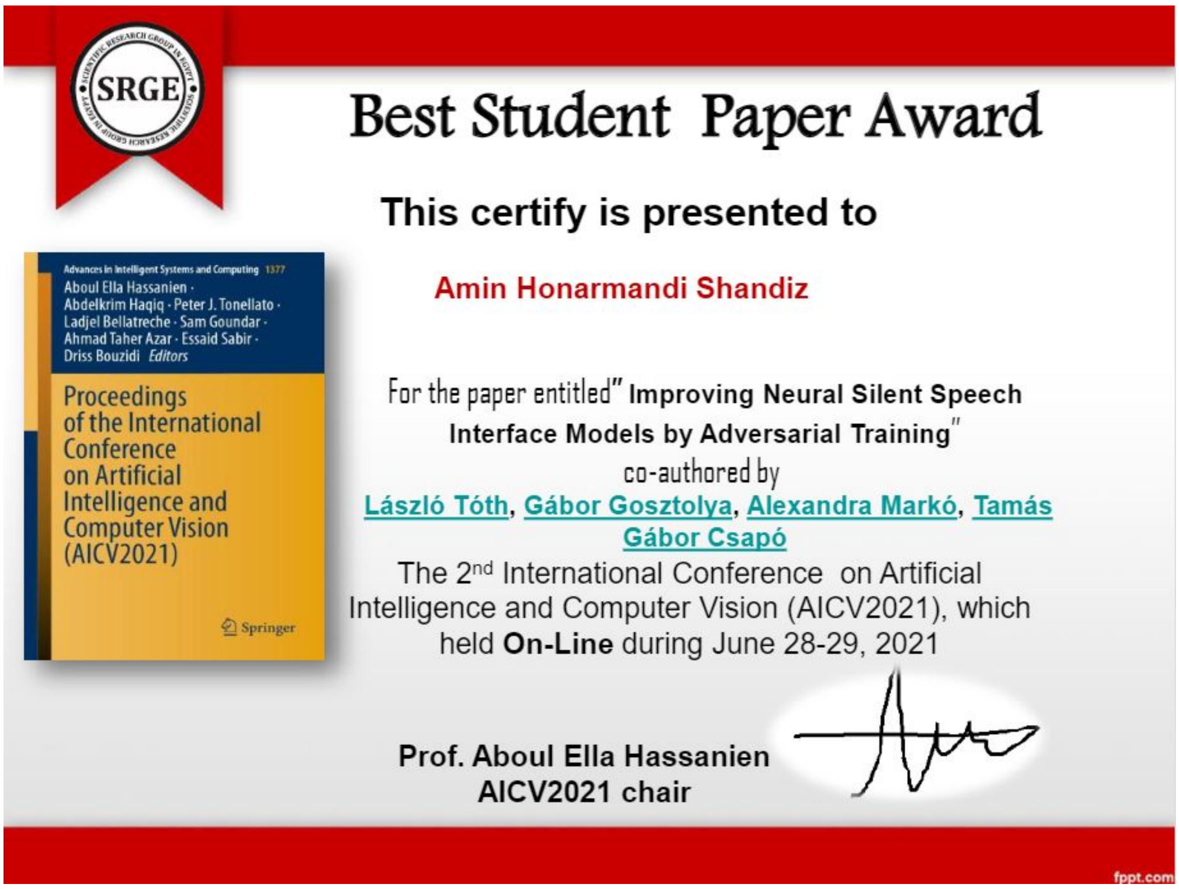
III/2. Utilization of various SSI models as generators in the GAN and CGAN frameworks.

III/3. Calculation and analysis of performance metrics to evaluate the effectiveness of the models.

III/4. Conducting research in the field of GANs and image generation.

This chapter's results were published in [36] and received a best paper award (see fig 3.5).

Figure 3.5: *Best paper award for the Resulting publication*



SRGE

SCIENTIFIC RESEARCH GROUP IN EGYPT

Best Student Paper Award

This certify is presented to

Amin Honarmandi Shandiz

For the paper entitled" Improving Neural Silent Speech Interface Models by Adversarial Training" co-authored by László Tóth, Gábor Gosztolya, Alexandra Markó, Tamás Gábor Csapó

The 2nd International Conference on Artificial Intelligence and Computer Vision (AICV2021), which held **On-Line** during June 28-29, 2021

Advances in Intelligent Systems and Computing 1377
Aboul Ella Hassanien · Abdelkrim Haqiq · Peter J. Tonellato · Ladjel Bellatreche · Sam Goundar · Ahmad Taher Azar · Essaid Sabir · Driss Bouzidi *Editors*
Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV2021)
Springer

[illegible]

Prof. Aboul Ella Hassanien
AICV2021 chair

fppt.com

Chapter 4

Neural Speaker Embeddings for Generalizing our Ultrasound SSI model

In earlier chapters, we discussed the development of new models and the improvement of their performance. However, when these models are tested on unseen data, their results may not be accurate. This could be due to a lack of comprehensive studies on generating intelligible speech from biosignals, which are often conducted on small databases of just one or a few speakers. Collecting data for these studies is time-consuming and requires the patience and motivation of the subjects. Additionally, articulatory tracking devices are highly sensitive to the anatomy of the individual speaker, and a misalignment of the recording equipment can further impact the accuracy of the results.

To address these challenges, various methods have been developed, such as signal normalization and model adaptation. These methods have been applied different modalities, including EMG and acoustic signals. One study by Ribeiro [81], used x-vector features to improve speaker-independent models by adding information from speaker recognition models. Although their research was applied to acoustic signals, we implemented their idea on articulatory signals to improve the performance of speaker-independent models. Our experiments were conducted on the English TAL dataset, and we presented new extracted features. Our results showed that this method was useful for improving the performance of speaker-independent models in the ultrasound SSI framework.

4.1 Problem Description and Literature Overview

Although there are lots of studies on generating intelligible speech from the above ultrasound video, most of these were conducted on relatively small databases of just a single or a small number of speakers [20, 26, 88]. Meanwhile, all of the articulatory tracking devices are obviously highly sensitive to the actual speaker's anatomy. A further source of variance may come from the possible misalignment of the recording equipment. For example, for ultrasound recordings, the probe fixing headset has to be remounted onto the speaker for each recording session. This inevitably causes the recorded ultrasound videos to become slightly misaligned between each recording session.

Although the properly working cross-session and cross-speaker methodologies are still missing, there have already been studies in this direction. Kim et al. investigated speaker-independent Silent Speech Recognition using EMA with 12 healthy and laryngectomized speakers [52]. For EMG-based recognition, several signal normalization and model adaptation methods were investigated by Maier-Hein et al. [65]. Janke et al. studied session-independent sEMG over 16 sessions of a speaker, and the results showed that sEMG is quite robust to minor changes in the electrode placement [46]. Wand et al. utilized domain-adversarial DNN training to increase the session-independency of their EMG-based speech recognizer [106]. For EOS-based SSR with a small vocabulary, Stone and Birkholz found the speaker-independent average word accuracy to be relatively stable, varying between 56–62% [96].

Ultrasound-based SSI systems, however, might be less robust, as slight changes in probe positioning causes shifts and rotations in the resulting image.

In this Chapter, we examined the session dependency of UTI-based direct speech synthesis, and we proposed a simple session adaptation method [28]. Ribeiro et al. experimented with the classification of UTI images of phonetic segments [81], and found that speaker dependent systems perform much better than speaker independent ones, but adding speaker information in a simple form (e.g. mean ultrasound image) helps the model generalize to unseen speakers.

The same authors reported that unsupervised model adaptation can improve the results for silent speech (but not for modal speech) [83]. They also performed multi-speaker recognition and synthesis experiments where they applied x-vectors for speaker conditioning – but they extracted the x-vectors from the acoustic data and not from the ultrasound [82].

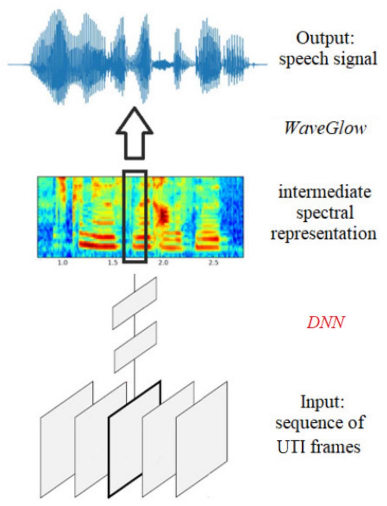
Just like the above-mentioned biosignals, speech signals also contain speaker-specific factors that hurt the cross-speaker performance of speech processing systems. Various DNN training and adaptation methods have been proposed as remedies, perhaps the simplest being is to use auxiliary input features that encapsulate information on speaker characteristics. The classic such representation was the i-vector [89], but its role has recently been taken by the x-vector [94]. The x-vector can be created by training a DNN for speaker classification, and then the activation values of an upper hidden layer are used as the speaker embedding vector in other tasks. Our goal here is to adjust this x-vector scheme to ultrasound tongue videos. To this aim, we train a DNN for speaker classification, using 3D blocks of adjacent ultrasound frames as input. First, we report the speaker recognition accuracy of this network in 4.5.1. Next, we evaluate the speaker embedding vector provided by this network on a separate set of speakers via very simple speaker recognition experiments in 4.5.2. Finally, we attempt to apply the embedding vectors as auxiliary input for a second network which is trained to perform speech synthesis, more precisely, to estimate spectral vectors from the ultrasound frames (4.5.3). Before the experiments, brief descriptions are given regarding our SSI framework, x-vectors and the experimental conditions in Sections 4.2, 4.3 and 4.4, respectively.

4.2 The SSI framework for Speaker Embedding model

We presented our approach for creating an ultrasound-based SSI in previous chapters, so we just give a brief overview here. As Fig 4.1 shows, the input to our system is a sequence of ultrasound tongue imaging frames, and the target sequence is a speech signal. This is a sequence-to-sequence mapping problem, which could be addressed by sophisticated encoder-decoder networks that would not even require aligned training data [82]. However, as we have synchronized input-output samples, most authors apply simpler networks that perform the mapping frame by frame [18, 100]. The optimal output representation is also a matter of choice. While training DNNs that generate speech signals directly is feasible [75], it would require large amounts of training data. Alternatively, one can use a dense spectral representation as the training target that can be converted to speech [15]. Recently, speech synthesis technology has introduced neural vocoders for synthesizing speech from spectrograms [75]. The main advantage of applying these in the SSI task is that we can borrow large pre-trained networks for the speech synthesis step, and we have to deal only with the ultrasound-to-spectrum mapping task. In the chapter 3, we used WaveGlow [18], and we obtained higher quality speech than with standard vocoders [15].

As WaveGlow requires a mel-spectrogram with 80 spectral components as input, our DNN was trained to estimate such spectral vectors from the UTI video in a frame-by-frame manner. The input of our network is a 3D array of consecutive images, and the output is a 80-dimensional spectral vector. The convolutional network structure that we applied here [102] was the same as the lower, 'frame-level' part of the xx-vector network, so we delay its presentation to the next section. As here the task is to estimate spectral vectors, we used a linear output layer and the network was trained to minimize the mean-squared error of the regression task.

Figure 4.1: *Schematic diagram of the UTI-to-speech conversion process applied in our SSI model.*



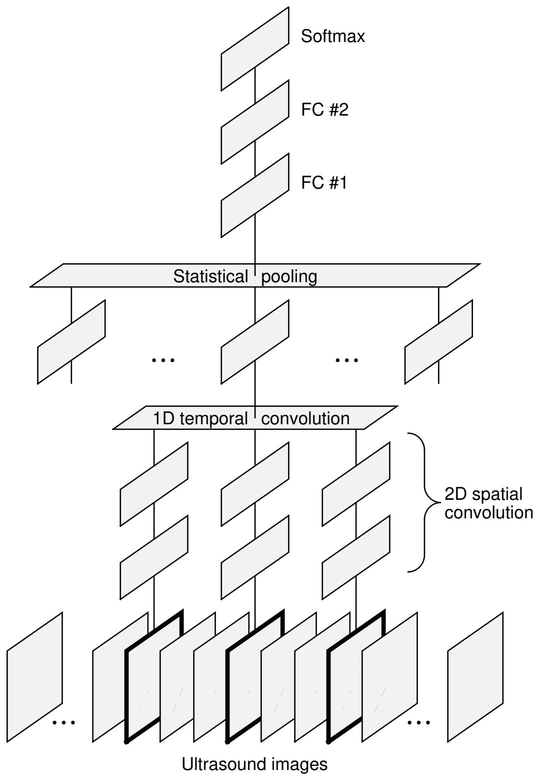
4.3 Speaker embedding vectors for ultrasound tongue imaging

The x-vector concept was introduced by Snyder et al., motivated by the goal of replacing the previous Gaussian-based i-vector approach with a purely neural solution [94]. The basis behind the x-vector is a DNN that is trained to discriminate speakers. The network structure is unusual in the sense that it consists of three main parts. The lower layers – typically a time-delay network (TDNN [74]) – operate on the level of frames. Then, the subsequent temporal pooling layer aggregates statistics over the frames of a given speech segment or utterance. This layer may collect only the mean values [61], the mean and the standard deviation [92, 94, 95] or it may even apply a sophisticated attention mechanism [72, 109]. The aggregated values are processed further by several fully connected layers. As these layer operate on the segment level, after training they can produce a fixed-size speaker embedding vector even from utterances of variable lengths. The embedding vector is typically obtained as the linear activation output of the fully connected layer right below or two layers below the softmax output [94].

We adjusted the x-vector DNN to operate with a sequence of ultrasound images instead of a sequence of speech feature vectors as follows (see also Fig 4.2).

The **frame-level part** has the same structure as the network we apply in the spectral estimation step of Fig 4.1 [102]. Its input consists of 21 consecutive images, which are processed by a 3D convolutional layer in 5-image blocks, using a relatively large stride size of 4 along the time axis. The subsequent convolutional layers apply

Figure 4.2: *Illustration of the UTI-based x-vector network.*



the 3D convolution in a decomposed "(2+1)D" form, first processing these blocks locally, and aggregating their content along time only at higher layers. This architecture was motivated by the findings of Tran et al. for video recognition [104], and also by the success of TDNNs in speech recognition [74, 101]. The top layer of the frame-level part is a dense layer that produces a local output vector at each frame position (ie., for the center frame of the input block).

The **statistical pooling** layer performs a simple average pooling. We leave more complex solutions such as standard deviation pooling or attention-based weighting for future work.

The **segment-level part** consists of two fully connected layers and a softmax output layer that has one output neuron for each speaker in the training set. The neural speaker embedding vector for a given input segment or utterance is extracted from one of these dense layers (see the experiments in Section 4.5).

4.4 Experimental Setup

In the experiments we used the TaL80 corpus [82], which contains ultrasound, speech and lip video recordings from 81 speakers. Apart from the silent speech experiments, the speech signals were also recorded in parallel with the ultrasound, and here we used these synchronized ultrasound and speech tracks. The ultrasound was recorded using Articulate Instruments' Micro system that captures a midsaggital view of the tongue at a frame rate of 82 fps. The raw ultrasound images contain 64 x 842 data values, which were resized to 64 x 128 pixels. More details about the recording process can be found in [82].

We divided the data into two sets, so we used 50 speakers to create the x-vector network, and we held out 31 speakers to train and evaluate the SSI network. Although the x-vector network is able to handle inputs with different sizes, for technical simplicity we chose to work with short uniform 2-second long chunks from each recording (similar to Shon et al. [92]). To train the x-vector network, we extracted such 2-second (164 frames) chunks from each training file of the 50-speaker subset of the corpus, resulting in 76 chunks from each speaker on the average (minimum 65, maximum 85). Altogether we obtained 3800 such blocks, from which 2298 were used for training, 357 for development, and 1145 blocks for testing.

The network used for the spectral estimation task in the SSI and the frame-level part of the x-vector network were structurally identical, consisting of 4 3D-convolutional layers (with a MaxPooling layer after every second convolution), and a fully connected layer that aggregates the convolutional outputs (the exact model parameters can be found in chapter 2). The fully connected hidden layer was followed by a linear output layer for the spectral estimation task, while in the x-vector network the fully connected layer outputs were aggregated along time by average pooling. The segment-level part consisted of two fully connected layers (FC#1 and FC#2 in Fig 4.2) with sizes of 500 and 250, and the softmax output layer contained 50 neurons, corresponding to the 50 speakers of the train set. Both networks were trained using the Adam optimizer with a learning rate of 0.0002, using a batch size of 100 for the SSI network, and batch sizes of 4-16 for the x-vector network (using smaller batch sizes for longer segments). The nonlinearity applied in all hidden layers was the swish function.

4.5 Results

4.5.1 Training the x-vector network

In the first experiment we were interested to measure the speaker classification accuracy of our x-vector network, and how it is influenced by the input duration. The motivation for the aggregation step in the original x-vector model is that normally we have not just a single frame but whole utterances (at least a word) from a given speaker, and that certain phonetic segments may be less speaker-discriminative than others [92] – the most trivial example are the silent parts. But the situation is differ-

Segment length (frames & seconds)	Aggregated frames	Error rate (Dev Set)
21 (0.25 sec)	1	3.16%
41 (0.50 sec)	21	2.87%
82 (1.0 sec)	62	2.10%
164 (2.0 sec)	144	1.96%

Table 4.1: *Speaker recognition error rates for the 50-speaker set as a function of the input segment duration.*

ent when the input is an ultrasound tongue video, as the recording device always returns an image of the tongue, even when the speaker remains silent, and thus longer segments may not be necessary. In the first experiment we gradually increased the size of the input segment from 21 frames (when the lower part of the network returns one frame-level output, so effectively no temporal pooling occurs) to 164 frames (2 seconds).

Table 4.1 shows the speaker recognition error rates on the development set of the 50-speaker corpus for different durations of temporal pooling. As we expected, we obtained a very low error rate already with just a single-frame output, which shows that the UTI videos are very speaker-specific. Gradually increasing the segment length and thus the number of aggregated frames improves the results, but the improvements are smaller than those reported, for example, in language recognition [93].

4.5.2 Validating the x-vectors on an independent speaker set

The second experiment sought to test the generalization ability of the x-vector. We were worried about overfitting the training speakers, as the popular x-vector implementations are trained with orders of magnitudes more training data and typically with more than a thousand speakers [94]. Thus, to validate our x-vector DNN, we performed a speaker recognition experiment on the dataset of the 31 speakers not used during x-vector training.

Using the speaker embeddings to recognize or verify a new set of speakers is not trivial, as the speakers will be different from those seen during training. For optimal performance, the embedding vectors are typically post-processed by factor analysis or dimension reduction methods [19, 94]. A simple and fast, though somewhat sub-optimal solution is to calculate the similarity score of two speakers based on the cosine similarity of their embedding vectors [19]. According to this, we performed a simple speaker recognition experiment with a 1-nearest neighbor classifier using the cosine distance as follows. We fused the development and the test sets of the

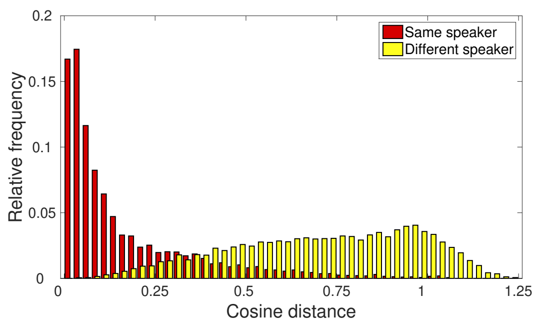
31 speakers, and calculated the x-vectors for each segment. Then we performed leave-one-out classification, that is, to each vector in this set we found the closest one (excluding itself). The classification was considered correct if the speaker IDs of the two segments were identical. We intentionally used the simplest possible 1-NN classifier, as it is very sensitive to the accuracy of the distance function applied, and hence to the accuracy of the underlying x-vectors.

Embedding layer	*FC#1*		*FC#2*	
Activation	swish	linear	swish	linear
Error rate	0.96%	0.96%	2.03%	0.70%

Table 4.2: *Speaker recognition error rates for the held-out 31 speakers using 1-NN leave-one-out testing.*

Snyder et al. defined the x-vector as the linear output of the first fully connected layer after statistical pooling [94]. However, some authors questioned whether this is the optimal strategy, and also whether dropping the nonlinearity is required [92]. So we also examined which fully connected layer gives the best result, and if the nonlinearity is necessary or not. The results are shown in Table 4.2. As can be seen, all embedding extraction methods resulted in very low speaker recognition error rates around 1-2%. The best result was obtained when the speaker embedding was produced by the lower fully connected layer (FC#2) without the nonlinearity. This coincides with the findings of Snyder et al. [94] and Shon et al. [92].

Figure 4.3: *Normalized histogram of the cosine distances for randomly chosen same-speaker and different-speaker x-vector pairs from the 31-speaker set.*



To further demonstrate the speaker discriminative abilities of the x-vectors on a new set of speakers, a histogram of the cosine distances is shown in Fig 4.3 for 10000 randomly selected same-speaker and different-speaker vector pairs from the 31-speaker dev+test subset. Although the distributions overlap slightly, this figure also suggests that the embedding vectors behave as expected, that is, vectors from the same speaker are closer to each other than vectors from different speakers.

4.5.3 Applying the speaker embedding in speech synthesis

In the last experiment we attempted to use the x-vector as an auxiliary input for the spectral estimator network of our SSI system (cf. Fig 4.1). As the baseline, we trained the net with the single-speaker TaL1 corpus. The obtained mean squared error rates shown in table 4.3 are around 0.26, which results in a speech signal with a mel-cepstral distortion of 3.12 after speech synthesis. This corresponds to a low-quality, but intelligible speech [18]. For multi-speaker training and testing we used samples from the 31-speaker subset. We did not use all the available data from the 31 speakers to demonstrate how the accuracy of the SSI network drops when switching to a multi-speaker scenario with a similar amount of training data. Table 4.3 shows that the multi-speaker setup led to a drastically larger MSE. We note that the multi-speaker scenario is still speaker-dependent in the sense that the training, development and test sets are from the same speakers. Even larger performance drops can be expected in a speaker-independent configuration [81].

Finally, we re-trained the multi-speaker model with the x-vector as auxiliary input. However, the optimal way of combining the ultrasound input with the x-vector is not trivial. As the ultrasound images are processed by convolution, simple concatenation was not an option. We decided to inject the x-vector into the network only after the convolutional layers, which were initialized by transfer learning from the multi-speaker model. However, this solution might be suboptimal and requires further studies. We mention that Ribeiro et al. used the mean ultrasound frame to represent the speaker, so they could add this image to the CNN input as a second channel [81].

SSI Train+Test configuration	Size of training set (frames)	MSE	
		Dev	Test
single-speaker	254306	0.256	0.265
multi-speaker	305040	0.603	0.669
multi-spk + Xvec		0.589	0.653

Table 4.3: *Mean squared error for the SSI spectral estimation task in the single-speaker and multi-speaker scenarios.*

The bottom row of Table 4.3 shows that although the introduction of the x-vector resulted in a consistent improvement for both the development and the test sets, this improvement is marginal. Ribeiro et al. reported similarly small gains from using the speaker mean as the speaker-characteristic input [81].

4.6 Summary

Here, we adjusted the x-vector framework of speech processing to ultrasound tongue videos to create a speaker-characteristic embedding vector. We modified the network architecture to process videos as input, and trained the network for speaker recognition. We obtained very low speaker recognition error rates, and our embedding vectors also seem to generalize well to new speakers. However, our first attempts to apply the embedding vector in a multi-speaker SSI scenario resulted in just a minimal improvement, showing that further studies are required on the proper application of the x-vector in this field.

The author of this PhD thesis is responsible for the following contributions presented in this chapter:

IV/1. Preparing data for the experiments.

IV/2. Implementing the model and conducting the experiments.

IV/3. Comparing the results obtained from the experiments.

IV/4. Calculating the relevant metrics to evaluate the model's performance.

And the results presented in this chapter were published in [37].

Chapter 5

Convolutional Neural Networks for Detecting Voice Activity in Silent Speech Interfaces based on Ultrasound

In order to prepare data for an SSI system and generate spectrograms from speech ultrasound videos, researchers often employ methods to remove parts of the signal that contain silence. One such method is voice activity detection [105], which has multiple benefits. However, previous implementations of VAD have resulted in misalignment between spectrograms and ultrasound frames due to the removal of silent parts prior to spectrogram generation.

To address this, we propose a new approach for implementing VAD, where we detect and separate silence from speech and only remove the former. We test this method on an English corpus dataset and use both objective evaluation metrics and conventional metrics to demonstrate the improved performance of our model.

5.1 Problem Description and Literature Overview

Voice Activity Detection is an important component in many speech processing applications, for example automatic speech recognition (ASR) [5, 64] and speech enhancement [105]. Its main role is to detect the presence or absence of speech [105], but sometimes it also involves a voiced/unvoiced decision [71]. Its application can not only significantly reduce the computational costs, but it may also influence the speech recognition accuracy [7]. In speech enhancement, VAD is used to identify frames which contain only noise to remove them from the signal [105]. In machine learning-based speech synthesis, during training removing noisy segments and pauses may help generate more accurate models.

In this chapter, the focus is on improving the prepared data for silent speech interfaces which convert articulatory signals to acoustic signals using deep neural networks. As we discussed in previous chapters, ultrasound images that record the movement of the tongue during speaking serve as the articulatory input. The recent studies in this field have all utilized DNNs, including structures that combine convolutional neural network (CNN) layers and recurrent layers like the long short-term memory layer.

Similar to speech applications, voice activity detection may also be useful in SSI systems, for example for sparing with the energy consumption in wearable ultrasound-based SSI devices. However, in this case creating VAD algorithms is much more difficult, as the lack of speech does not correspond to a lack of high-amplitude input signal. The tongue position is continuously being recorded and presented by the ultrasound imaging tool, even when the subject is not speaking.

In this part, we first demonstrate that the application of VAD may impact the accuracy of our SSI neural model, the speech synthesis network we apply, and even the evaluation metric we use. Then we implement a CNN to separate silence and speech frames based on the ultrasound tongue images, so we basically create a VAD algorithm that works with ultrasound images. Finally, we evaluate the performance of this algorithm experimentally, and we also compare the performance of our SSI framework with and without using the VAD algorithm.

The chapter5 is organized as follows. In section 5.2, we briefly present our SSI approach, the we talk about the problem of voice activity detection in 5.3. Then the experimental setup is described in section 5.4 and the experiments are presented and discussed in Section 5.5. We close the chapter with conclusions in Section 5.6.

5.2 The Ultrasound-Based SSI Framework

Our SSI system follows the structure explaind in chapetr 3. The input of the system is a sequence of ultrasound tongue images that were recorded at a rate of 82 frames per second. The goal of the SSI system is to estimate the speech signal that belongs to the articulatory movement recorded in the ultrasound images, so the SSI system has to create a model for the articulatory-to-acoustic mapping. We estimate this mapping using deep neural networks. For the training procedure, we assume that the speech signal was also recorded in parallel with the ultrasound video, as this speech signal serves as the training target. Also, to reduce the amount of training data required, we estimate a dense spectral representation instead of the speech signal itself. In practice it means that our SSI network converts the ultrasound video into a mel-spectrogram, and the output speech signal is generated from the mel-spectrogram using the WaveGlow neural vocoder [75]. we shown that this approach is feasible, and it can generate intelligible speech from a sequence of ultrasound images (chapter 3).

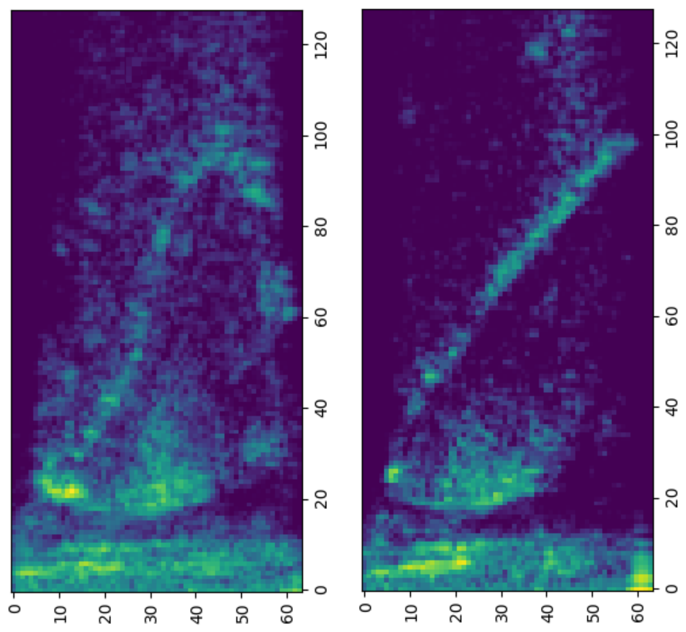
Here, we evaluate the accuracy of spectral regression by two simple metrics. One of them is simply the Mean Squared Error loss for the training of the neural network. The other one is the Mel-Cepstral Distortion between the original speech signal and the speech signal reconstructed from the ultrasound input [57], which is a popular metric of speech quality in speech synthesis [54].

5.3 Voice Activity Detection from Speech and from Ultrasound

The main role of Voice Activity Detection is to estimate the presence or absence of speech [105]. In the simplest case, that is, in a quiet environment the lack of speech activity corresponds to silent parts in the input signal. Hence, the simplest VAD algorithms compare conventional acoustic features such as the signal's energy to a threshold [78]. Exceeding the threshold signs the presence of speech (VAD=1), otherwise the signal is identified as silence (VAD=0). However, the task becomes much more difficult under noisy conditions. The speech phones can be voiced and unvoiced, and the most difficult is to separate unvoiced parts from background noise. Thus many VAD algorithms extend the two-class classification to 3 classes, corresponding voiced, unvoiced and silent (V/UV/S) parts [71, 105]. Moreover, under noisy conditions simple acoustic features such as the signal's energy may be insufficient, so several more sophisticated features have been proposed. For example, the classic paper by Atal et al. performs the prediction based on five different measurements including zero-crossing rate, speech energy, correlation features, 12-pole linear predictive coding (LPC), and the energy of the prediction error [5]. Other authors used features such as the zero-crossing rate, spectral or cepstral features, empirical mode decomposition (EMD), and so on [5, 32, 71]. More recent studies apply machine learning methods to perform the voiced/unvoiced decision [21, 69, 76, 77]. Mondal et al. applied clustering over temporal and spectral parameters to implement their VAD [70].

While there are a lot of studies on voice activity detection from speech, the input of our SSI system consists of ultrasound tongue images. Fig 5.1 shows two examples of the tongue position recorded by the device, when the subject is speaking (producing a vowel) and when he is not – the diagonal light stripes in the images correspond to the tongue of the speaker. After examining several samples, we got the impression that speaking versus remaining silent typically results in more drastic changes in the speech signal than in the corresponding ultrasound tongue images, so voice activity detection based on the latter is presumably much harder. In the following we train a CNN to perform the voiced/invoiced classification using such ultrasound images.

Figure 5.1: *Two UTI examples from the database, one for a speech (vowel) frame (left) and one for a silent frame (right).*



As the structure of this VAD-CNN and the network that we apply for the SSI task are very similar, we describe them together in the next Section.

5.4 Experimental Setup

5.4.1 The Ultrasound Dataset

For the experiments we used the English TAL corpus [82]. It contains parallel ultrasound, speech and lip video recordings from 81 native English speakers, and we used just the TaL1 subset which contains recordings from one male native speaker. We partitioned his files into training, testing and validation sets using 1015, 24 and 50 files, respectively. To preprocess the ultrasound images we applied minmax normalization to the [-1,1] range, and resized the images to 64*128 pixels using bicubic interpolation. As regards the normalization of the speech mel-spectrogram features, we tried different normalization techniques, but we got the best results with the standard mean-deviance normalization (standardization). These 80 mel-spectrogram coefficients served as the training target values for the SSI network.

5.4.2 CNNs for the SSI and for the VAD Task

Convolutional Neural Networks are currently the most popular tool in image recognition, as they proved very powerful in extracting complex features from images by creating very deep network architectures [56]. Standard CNNs convolve 2D filters with the images, but when the input is a video or a time series, CNNs can be extended to 3D by considering time as the third dimension [48, 119]. Recurrent neural networks such as the LSTM can also be very effective in extracting and combining temporal information from a sequential input [34]. However, these networks are known to be slow, so variants such as the quasi-recurrent neural network have been proposed [11]. This is why several authors apply 3D-CNNs to substitute recurrent layers when applying CNNs to a sequence of images [119]. Here, we experiment with two neural network configurations in our SSI framework, that is, to estimate a speech mel-spectrogram from a sequence of ultrasound images. The first network is a 3D-CNN, folloeing the proposal of chapter 2. The second configuration combines the 3D-CNN layers with and additional BiLSTM layer, as it may be more effective in aggregating the information along the time axis. The structure of the two networks is compared in Table 5.1. The input for both networks is the same, a short sequence of adjacent UTI frames. The output corresponds to the 80 mel-spectral coefficents that has to be estimated for the WaveGlow speech synthsis step, and the network is trained to minimize the MSE between the target and the output spectral vectors.

Conv3D	**Conv3D+BiLSTM**
Conv3D(30,(5,13,13), strides=(5,2,2))	Conv3D(30,(5,13,13), strides=(5,2,2))
Dropout(0.2)	Dropout(0.2)
Conv3D(30,(5,13,13), strides=(5,2,2))	Conv3D(30,(5,13,13), strides=(5,2,2))
Dropout(0.2)	Dropout(0.2)
Conv3D(60,(1,13,13), strides=(1,2,2))	Conv3D(60,(1,13,13), strides=(1,2,2))
Dropout(0.2)	Dropout(0.2)
MaxPooling3D	MaxPooling3D
Conv3D(60,(1,13,13), strides=(1,2,2))	Conv3D(60,(1,13,13), strides=(1,2,2))
Dropout(0.2)	Dropout(0.2)
MaxPooling3D	MaxPooling3D

Table 5.1: *The structure of the 3D-CNN and the 3D-CNN + BiLSTM networks in Keras for the SSI task. The differences are shown in bold.*

In addition to using the VAD implementation available from WebRTC[1], we trained a CNN to perform VAD directly from ultrasound images. The training in-

Conv2D
Conv2D(32, (3, 3), padding='same', Activation='relu')
MaxPooling2D
Conv2D(32, (3, 3), padding='same', Activation='relu')
MaxPooling2D((2,2))
Conv2D(64, (3, 3), padding='same', activation='relu')
MaxPooling2D
Conv2D(64, (3, 3), padding='same', activation='relu')
MaxPooling2D((2,2))
Conv2D(128, (3, 3), padding='same', Activation='relu')
MaxPooling2D()
Flatten()
Conv2D(128, (3, 3), padding='same', Activation='relu')
MaxPooling2D((2,2))
Flatten()
Dense(128, activation='relu'))
Dense(1, activation='sigmoid')

Table 5.2: *The structure of the 2D-CNN used for classification of speech/silent ultrasound images.*

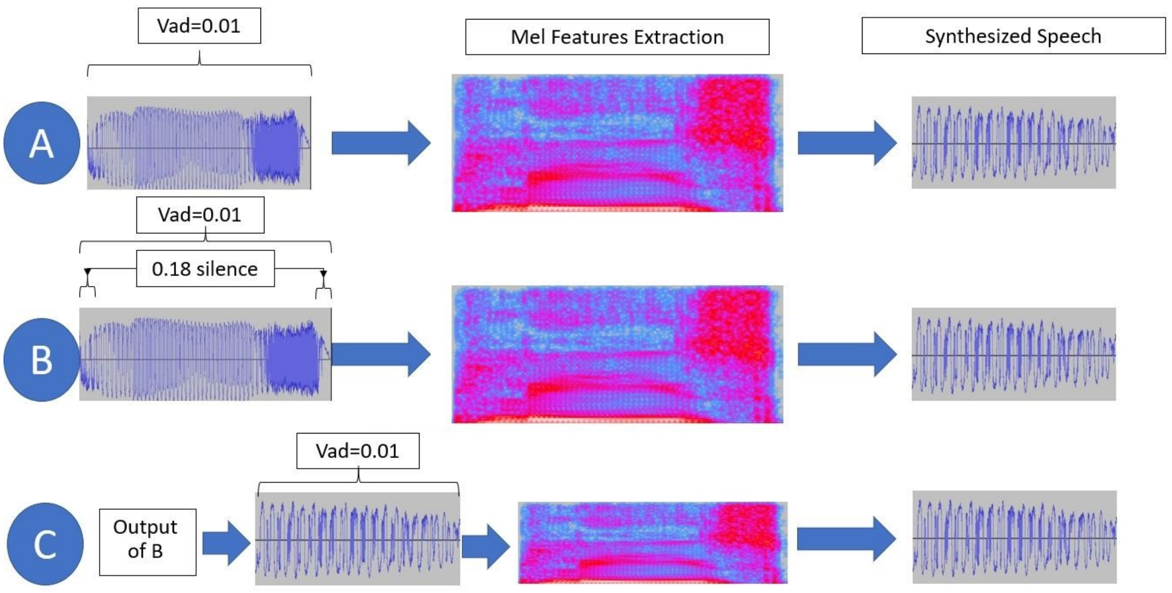
volved using a simple frame-by-frame approach, where a single image was used as input and a 2D-CNN was applied to classify each frame as either silence or speech(SI/SP). The network had a single output that estimated the probability of the frame containing silence. To train the network, we used the binary cross-entropy loss function. The architecture of the network can be found in Table 5.2.

5.5 Results

5.5.1 The Impact of VAD on the MCD Metric and on Speech Synthesis

In the first experiment, our objective was to investigate the impact of VAD application on our outcomes. It is noteworthy that this experiment did not involve SSI. Rather, we transformed the speech signals into mel-spectrograms and reconstructed them using WaveGlow. The results of this experiment highlighted the significance of VAD in improving the quality of reconstructed speech signals, especially in the removal of unwanted noise and silence. The findings demonstrate the potential of VAD as an effective pre-processing tool in speech-related tasks (see Fig 5.2).

Figure 5.2: *Illustration of the experimental configurations applied in Table 5.3,(values are in second).*



To measure the deviation between the original and reconstructed speech signals, we used the Mel-Cepstral Distortion (MCD) metric. In this experiment, we employed the VAD implementation from WebRTC [1] to detect voice activity and remove silence from the speech signals. By applying VAD, we aimed to minimize the effect of silent segments on the MCD metric and improve the quality of the reconstructed speech signals.

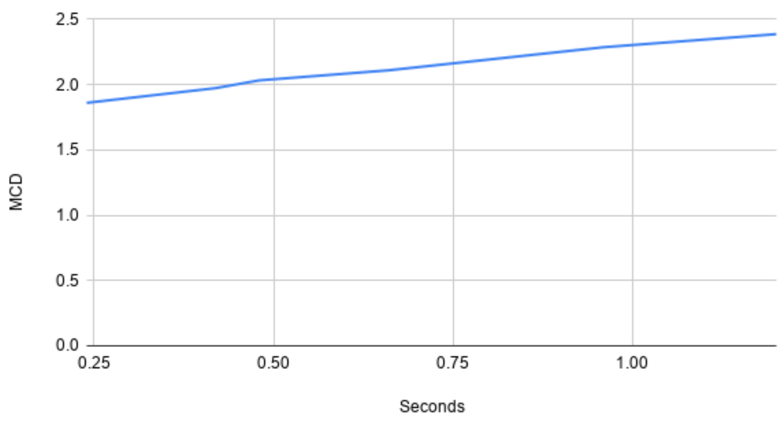
Configuration	**MCD**
A: VAD (window length = 10 ms)	1.55
B: VAD (window length = 10 ms), plus keeping 180ms silence at both ends	2.03
C: applying A after B	1.34

Table 5.3: *MCD values of the speech analysis-synthesis process when applying silence removal with three different VAD configurations.*

As shown in the first two rows of Table 5.3, retaining longer silent parts before and after the speech signal does influence the MCD.

In the third row of Table 5.3 we present one more experiment where we performed two analysis-synthesis steps. Theoretically, the analysis and synthesis steps should be the perfect inverse of each other, so experiment (c) should give the same result as experiment (a). However, we obtained a slightly different MCD value. The probable explanation is that the WaveGlow speech synthesis network does not give a prefect reconstruction, and it is sensitive to certain parameters such as the duration of the silent parts or the positioning of the input windows.

Figure 5.3: *Retaining more silence increases the MCD.*



In our experiments, we observed the impact of preserving longer silent parts on the performance of the Mel-Cepstral Distortion metric. As shown in Fig 5.3, the MCD values consistently increased with longer silent parts, suggesting that the amount of silence in the input significantly affects the MCD values. However, it is worth noting that some authors exclude non-speech frames from the MCD calculation [54], while others do not specify this step in their methods.

5.5.2 The impact of VAD on the SSI

In the second experiment we evaluated how the application of VAD on the training corpus influences the performance of our SSI system. In these UTI-to-speech conversion experiments we used the ultrasound data set presented in Section 5.4.1, and the two network configurations we described in Table 5.1. Both models were trained using the Adam optimizer with a initial learning rate of 0.0002. We repeated the experiment with using the original training data and with removing most of the silent parts from the speech signals using the WebRTC VAD implementation, which in order of this removal the respective parts in ultrasound video were removed as well.

Table 5.4: *Evaluation metrics of the SSI system after training the models with removing or retaining silence in the speech data.*

	No silence(VAD)			VAD + 180ms silence		
	MSE (dev)	MSE (test)	MCD	MSE (dev)	MSE (test)	MCD
3DCNN	0.46	0.45	3.20	0.30	0.33	3.29
3DCNN+BiLSTM	0.39	0.42	3.08	0.259	0.29	3.13

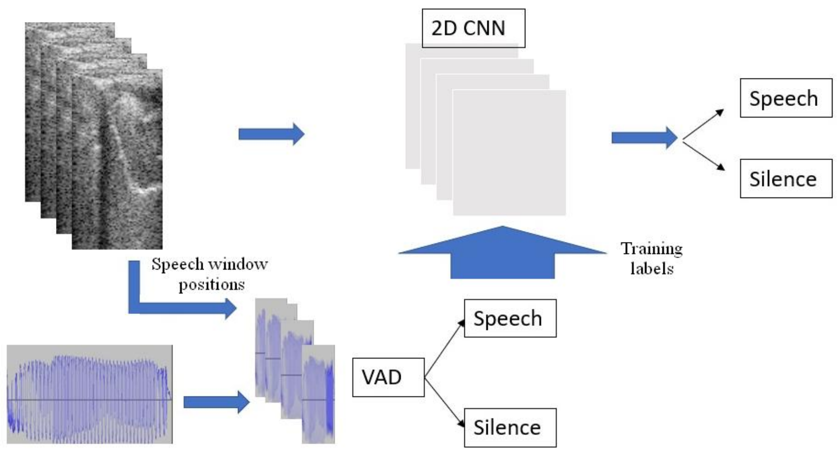
In Table 5.4 we report the MSE of the training process and the MCD values obtained from comparing the originals speech signals with those synthesized from the UTI input. The first thing we may notice is that the 3D-CNN+BiLSTM network produced much lower MSE rates and also slightly lower MCD errors. This shows the clear advantage of using a BiLSTM layer instead of a simple Dense layer. Second, the MCD scores are much higher in this case than in the previous experiment. This is because there we worked with the original spectrograms, so the reported MCD values of 1.3-2.0 were caused by the inaccuracy of the WaveGlow neural vocoder. Here, the spectrograms were estimated from the UTI images, so the errors of our spectral estimation network and the WaveGlow network add up. Our best MCD score of 3.08 corresponds to a low-quality but intelligible speech [18]. In comparison, Ribeiro et al. obtained an MCD score of 2.99 on the same corpus using more sophisticated encoder-decoder networks [82].

One interesting observation from our experiments is that retaining more silence in the corpus leads to lower mean squared error rates during training. However, this reduction in MSE can be misleading as it results in an increase in Mel-Cepstral Distortion values on the test set, as we observed. This suggests that adding more training samples from a single class, especially from a trivial class like silence, may shift the focus of training and have a negative effect on the performance of a DNN. Therefore, it is essential to carefully balance the amount of speech and silence in the training data for SSI systems.

5.5.3 Classification of Speech and Silence from Ultrasound

In the previous experiment, VAD was applied to the speech signal. However, in practical SSI scenarios, the speech signal may not be accessible, and thus VAD should be performed using ultrasound input. To address this, we conducted experiments to classify speech and silence frames of the ultrasound video using the 2D-CNN presented earlier in Table 5.2. This involved frame-by-frame training, where the input was a single image, and a 2D-CNN was applied to classify the frame as either silence or speech. The results are discussed in the following.

Figure 5.4: *Illustration of obtaining the VAD training labels and training the 2D-CNN for silence/speech classification.*



The training labels for this 2-class classification process were obtained as follows (see also Fig 5.4). As we have the synchronized speech signals for the ultrasound videos, we first identified the speech frames that belong to each ultrasound image based on the ultrasound frame rate. We split the speech signal into frames and fed it to the speech VAD function to decide about the speech/silence label of each image. We used these target labels with the ultrasound images as input to train the 2D-CNN for ultrasound-based voice activity detection. We used the ReLU activation function for all layers except the last layer which applies a sigmoid activation function to produce an output value between 0 and 1. For training we used SGD optimization with an initial learning rate of 0.001. We extracted the speech/silence training labels from the same train, development and test files as earlier, and the amount of speech labels was approximately 2-3 times more than the number of frames labelled as silence.

The evaluation metrics for this 2-class task are shown in Table 5.5. Besides the usual classification accuracy, we also report the precision and recall values which show that the two classes were slightly imbalanced.

	Dev set	**Test set**
Accuracy	0.87	0.852
recall	0.94	0.95
precision	0.877	0.864
F1	0.91	0.9
ROC AUC	0.894	0.859
Cohen's Kappa	0.672	0.57

Table 5.5: *Evaluation metrics for the silence/speech classification task.*

This is also reflected by the confusion matrices which can be seen in Table 5.6. Thus, we also evaluated the AUC score based on the ROC of the classifier, which gave 0.89 for the development and 0.86 for the test set, respectively. Finally, we also mention that the F1 measure of 0.9 is also very good, and it could even be slightly improved by fine-tuning the decision threshold (which we did not adjust here). We also display the Cohen's Kappa values, which is a preferred metric in the case of imbalanced classes[41].

Dev Data				Test Data			
	Predicted				Predicted		
Actual		Negative	Positive	Actual		Negative	Positive
	Negative	2850	1302		Negative	1671	1268
	Positive	502	9295		Positive	418	8096

Table 5.6: *Confusion Matrices for the silence/speech classification task for the development and test sets.*

5.5.4 Replacing Speech-VAD by UTI-VAD

Finally, we repeated the experiment of Table 5.4, but this time using the UTI-based VAD algorithm instead of the speech VAD. As the results in Table 5.7 show, in this case we obtained even slightly better MSE rates than earlier with the standard VAD function. The MCD values are basically equivalent with those obtained earlier, and the slight advantage of training the system with the removal of the long silent parts remained. In summary, we can say that our ultrasound-based VAD algorithm performed similarly to the standard, speech-based VAD algorithm in this experiment.

Table 5.7: *Training the SSI system with removing or retaining silence from the data using the ultrasound-based VAD algorithm.*

	No silence(VAD)			VAD + 180ms silence		
	MSE(dev)	MSE(test)	MCD	MSE(dev)	MSE(test)	MCD
3DCNN	0.436	0.428	3.15	0.38	0.27	3.28
3DCNN+BiLSTM	0.393	0.41	3.05	0.35	0.26	3.12

5.6 Summary

Here we showed that – similar to voice activity detection for speech – ultrasound images can also be used to discriminate between Si/Sp segments. We estimated our training labels based on the parallel speech recording using a public VAD implementation. Our classifier attained a promising accuracy of 86% in discriminating frames of silence and speech. We also showed that preserving too much silence in the training set can influence both the training of the model and the quality of the generated speech. For our later data preparation we use our VAD technique as a method of silence removal as an initial step before feature extraction. That it, we window the speech signal in synchrony with the ultrasound frames and feed them to the VAD, and perform the subsequent feature extraction steps for synthesizing speech or other related tasks by using only the frames retained by VAD.

The author of this PhD thesis is responsible for the following contributions presented in this chapter:

V/1. Implementing the model for voice activity detection using a CNN architecture and training it with binary cross-entropy loss function.

V/2. Developing the idea of applying VAD to remove silence from the speech signal in SSI systems.

V/3. Analyzing the results of experiments with different amounts of silence in the corpus, comparing the MCD and MSE metrics, and evaluating the impact of VAD on the SSI.

And the results presented in this chapter were published in [35].

Chapter 6

Enhanced analysis of ultrasound tongue videos via the fusion of ConvLSTM and 3D Convolutional Networks

In previous chapters, we mentioned the traditional two-step process of estimating speech signals from articulatory data in SSI systems, which involved converting the input to text using speech recognition and then synthesizing speech based on the text. However, the more popular approach nowadays is to directly convert the articulatory signals to speech using Deep Neural Networks. For our work, we followed this direct approach and utilized DNNs, specifically convolutional neural networks, for our image-based SSI task.

As our input consists of a sequence of images, we combined an LSTM with a 2D CNN to extract information from individual video frames. Alternatively, we could have extended the 2D convolution to 3D by adding the time axis as an extra dimension as we implement it in Chapter 2. While ConvLSTM models have shown potential in improving model performance, the training time cost is still a concern. Deep neural networks generally require more data for better performance, which in turn requires more time to train the model. In order to address this issue, we explored the possibility of combining convolutional layers and LSTM layers into a single layer to improve both model performance and training efficiency. Although the conv(2+1)D model has shown improvements in model performance, the training time cost remains a concern. However, in this chapter we chose to experiment with the ConvLSTM layer type, which combines the advantages of both convolutional and recurrent processing in one layer, resulting in a more efficient training process with fewer parameters. This model first was used for precipitation nowcasting [91], the application of ConvLSTM models on ultrasound data has not been extensively studied, with only a few works focusing on this approach. One such task involved using ConvLSTM to detect tongue motion in ultrasound data[118]. Here we compare the performance of ConvLSTM models with previous 2D-CNN+LSTM and 3D-CNN approaches, as well as hybrid models that combined all three types of layers. Our results show that the hybrid approach yields the best performance for our task with less training time and fewer parameters.

The chapter is organized as follows. In Section 6.1, we briefly introduce the concept of the convolutional LSTM that we are going to use. In Section 6.2, we explain the data acquisition and processing steps for our input and output data. In Section 6.3, we present our experimental setup, while in and following Section 6.4, the experimental results are discussed and explained. Finally, in Section 6.5 our main conclusions are given.

6.1 Convolutional LSTM for SSI

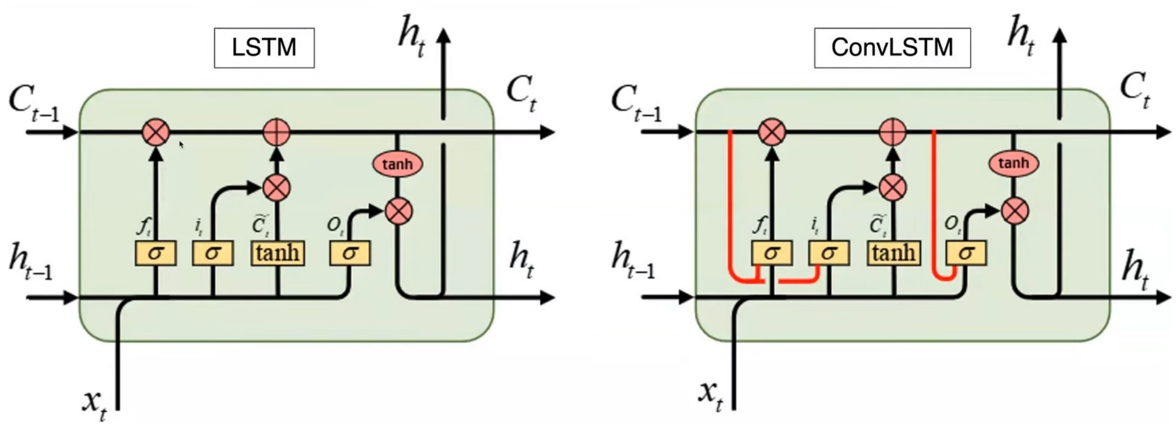
SSI systems synthesize speech from articulatory videos by learning the mapping between the input ultrasound image sequence and the output audio signal. SSI is a sequential task, as both the input and the output are sequences, with a strong correlation between consecutive elements of the sequence. As in our case the input data consist of ultrasound images, convolutional networks seems to be a proper tool for processing the input, as they are known to perform well when working with images [56], and also in particular with SSI ultrasound tongue images [50, 53]. As our input is a sequence, the information content along the time axis of the data can be extracted by applying Recurrent Neural Networks.

In particular, a variant of recurrent networks called the Long-Short Term Memory is known to be more effective in extracting long-term dependencies in the input sequence [34]. These networks have special gates in their internal implementations which improve their abilities to handle large-distance relations between time-related features.

The Long Short-Term Memory model is a type of Recurrent Neural Network that can effectively handle sequential data. In the standard implementation of LSTM, the input sequence is composed of vectors, and the data flow is shown in Fig 6.1. There are also variants of LSTM, such as the ”peephole” variant, which contains extra connections shown on the right side of Fig. 6.1, In both cases, the input of the LSTM consist of a sequence of vectors.

When processing a sequence of images, combining a Convolutional Neural Network with an LSTM is a common approach. In this approach, the images are first processed by a 2D-CNN to extract features along the spatial axes, and then the sequence of CNN outputs is integrated over time using an LSTM. This method is known as the CNN+LSTM approach, and it has been shown to work well in Single Slice Imaging

Figure 6.1: *Internal structure of a standard LSTM cell and its extended version (with extra peephole connections) used in Convolutional LSTMs [3, 4].*



implementations [50].

However, this method requires the combination of two types of layers, which can be computationally expensive. To address this issue, Shi et al. proposed a more efficient solution called the Convolutional LSTM. This method performs the two processing steps in one by applying a convolution operation in the inner steps of LSTM, instead of matrix multiplication. This allows the ConvLSTM to extract spatio-temporal features from the input data more efficiently [91].

Figure 6.2: *The equations behind the operation of Long Short Term Memory versus Convolutional LSTM neurons [91].*

LSTM

$$\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci} \circ c_{t-1} + b_i)\\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf} \circ c_{t-1} + b_f)\\
c_t &= f_t \circ c_{t-1} + i_t \circ \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)\\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co} \circ c_t + b_o)\\
h_t &= o_t \circ \tanh(c_t)
\end{aligned}$$

ConvLSTM

$$\begin{aligned}
i_t &= \sigma(W_{xi} * \mathcal{X}_t + W_{hi} * \mathcal{H}_{t-1} + W_{ci} \circ \mathcal{C}_{t-1} + b_i)\\
f_t &= \sigma(W_{xf} * \mathcal{X}_t + W_{hf} * \mathcal{H}_{t-1} + W_{cf} \circ \mathcal{C}_{t-1} + b_f)\\
\mathcal{C}_t &= f_t \circ \mathcal{C}_{t-1} + i_t \circ \tanh(W_{xc} * \mathcal{X}_t + W_{hc} * \mathcal{H}_{t-1} + b_c)\\
o_t &= \sigma(W_{xo} * \mathcal{X}_t + W_{ho} * \mathcal{H}_{t-1} + W_{co} \circ \mathcal{C}_t + b_o)\\
\mathcal{H}_t &= o_t \circ \tanh(\mathcal{C}_t)
\end{aligned}$$

This is reflected in the equations of Fig **??**, where $*$ represents the convolution operation, and $\circ$ stands for gating. Note that, apart from the convolution, the equations are exactly the same as those of the (peephole) LSTM. The convolution allows the more efficient processing of image sequences, resulting in better performance with much fewer parameters. For example, Kwon et al. successfully applied a hierarchical ConvLSTM for speech recognition [59]. Recently, Zhao et al. used ConvLSTMs for predicting subsequent ultrasound images in an SSI task [118].

Processing a sequence of images is also viable by extending the convolution operation to the time axis, resulting in a three-dimensional convolution (3D-CNN) model. The main advantage of this approach is that it is faster, as it applies only convolution operations. Convolution also allows the skipping of input images, which is not possible in an LSTM framework. In chapter 2, the 3D-CNN model gave results that were comparable or slightly better than those with the more conventional CNN+LSTM approach. Here, we extend this earlier comparison to the ConvLSTM model, and we are also going to experiment with hybrid models that combine 3D-CNN and ConvLSTM layers.

6.2 Data acquisition and preprocessing

The ultrasound data was collected from a Hungarian female subject (42 years old) while she was reading sentences aloud. Her tongue movement was recorded in a midsaggital orientation – placing the ultrasonic imaging probe under the jaw – using a ”Micro” ultrasound system by Articulate Instruments Ltd. The transducer was fixed using a stabilizer headset. The 2-4 Mhz / 64 element 20 mm radius convex ultrasound transducer produced 82 images per second. The speech signals were recorded in parallel with an Audio-Technica ATR 3350 omnidirectional condenser microphone placed at a distance of 20 cm from the lips. The ultrasound and the audio signals were synchronized using the software tool provided with the equipment. Altogether 438 sentences (approximately half an hour) were recorded from the subject, which was divided into train, development and test sets in a 310-41-87 ratio. We should add that the same data set was used in several earlier studies [16, 30, 102], and the data set is publicly available[1]. The ultrasound probe records 946 samples along each of its 64 scan lines. The recorded data can be converted to conventional ultrasound images using the software tools provided. However, due to its irregular shape, this image is harder to process by computers, while it contains no extra information compared to the original scan data. Hence, we worked with the original 964x64 data items, which were downsampled to 128x64 pixels. The intensity range of the data was min-max normalized to the [-1, 1] interval before feeding it to the network.

The speech signal was recorded with a sampling rate of 11025 Hz, and then converted to a 80-bin mel-spectrogram using the SPTK toolkit (http://sp-tk.sourceforge .net). The goal of the machine learning step was to learn the mapping between the sequence of ultrasound images and the sequence of mel-spectrogram vectors. As the two sequences are perfectly synchronized, it was not necessary to apply a sequence-to-sequence learning strategy. We simply defined the goal of learning as an image-to-vector mapping task, using the mean squared error as the loss function

[1]The dataset is available upon request from csapot@tmit.bme.hu

in the network training step. The 80 mel-frequency coefficients served as training targets, from which the speech signal was reconstructed using WaveGlow [75]. To facilitate training, each of the 80 targets were standardized to zero mean and unit variance. The input of training consisted of a block of 25 consecutive images. This allowed all DNN variants to involve the time axis in the information extraction process. The whole SSI framework followed the earlier chapter 2.

6.3 Experimental Setup

We implemented our networks using Keras with a tensorflow back-end [22]. We applied three different network architectures that can process 3-dimensional blocks of data. In the tables, ”3D-CNN” refers to the fully convolutional model proposed in Chapter 2. This model does not have any LSTM component. ”3D-CNN + BiLSTM” refers to a combination which applies a BiLSTM layer as the topmost hidden layer to integrate the temporal features extracted by the previous 3D-CNN layers. The final model referred to as ”3D-CNN + ConvLSTM” replaces the top first 3D-CNN and BiLSTM layers by a ConvLSTM layer. Notice that the ConvLSTM technique fuses the convolution and the LSTM operations into one layer, so here we can also spare one hidden layer by this substitution, therefore it could help the model to improve the model trainig time. In the following, we give a more detailed description of the three configurations.

3D Convolutional Neural Network(3D-CNN): This model was described in detail in [102] and Chapter 2, and its network layers are shown in Table 6.1. The networks processes the input sequence of 25 video frames in 5-frame blocks using 3D convolution. The overlap between these blocks is minimized by setting the stride parameter s of the time axis to 5. These blocks are processed further by 3 additional Conv3D layers, with pooling layers after every second convolution layer. Finally, the output is flattened and integrated over the time axis by a dense layer as the topmost hidden layer. The output hidden layer is a linear layer with 80 neurons, corresponding to the 80 spectral parameters given as training targets. This special network structure was motivated by Tran et al., who found that for the best result the processing should focus on the two spacial axes first, performing the integration over the temporal axis only afterwards [104]. Toth at al. also obtained the best result with performing the 3D convolution in this decomposed, ”(2D+1)” form [102]. Compared to that study, we achieved slightly better results with the same architecture by switching to the Adam optimizer instead of SGD, and by adjusting some meta-parameters, for example the dropout rate.

3D CNN + BiLSTM: As the output of the four layers of 3D convolution, the 3D-CNN network produces a sequence of 5 matrices, which are combined by a simple dense layer (cf. Table 6.1). Our first modification was to replace this fully connected

3D-CNN	3D-CNN + BiLSTM

Conv3D(30, (5,13,13), strides=(s, 2,2))
Dropout(0.3)
	Conv3D(30,(5,13,13), strides=(s,2,2))
	Dropout(0.3)

Conv3D(60, (1,13,13), strides=(1,2,2))
Dropout(0.3)
MaxPooling3D(pool_size=(1,2,2))
	Conv3D(60,(1,13,13),strides=(1,2,2))
	Dropout(0.3)
	MaxPooling3D(pool_size=(1,2,2))

Conv3D(90, (1,13,13), strides=(1,2,1))
Dropout(0.3)
	Conv3D(90,(1,13,13),strides=(1,2,1))
	Dropout(0.3)

Conv3D(85, (1,13,13), strides=(1,2,2))
Dropout(0.3)
MaxPooling3D(pool_size=(1,2,2))
	Conv3D(85, (1,13,13), strides=(1,2,2))
	Dropout(0.3)
	MaxPooling3D(pool_size=(1,2,2))

Flatten()
Dense(500)
Dropout(0.3)
	Reshape((5, 340))
	Bidirectional(LSTM(320,
	ret_seq=False))

Dense(80, activation='linear')	Dense(80, activation='linear')

Table 6.1: *The table compares the implementation of Conv3D and Conv3D+BiLSTM models, where Conv3D model uses 3D convolutional layers to extract spatial and temporal features, while the Conv3D+BiLSTM model combines the 3D convolutional layers with a bidirectional LSTM layer to capture the spatiotemporal dynamics of the data.*

layer with a (bidirectional) LSTM, which required us to reshape the matrices into vectors. The LSTM is a more sophisticated solution to extract the information from a temporal sequence, so we hoped to get slightly better results from this approach. As Table 6.1 shows, we set the $return\_sequences$ parameter of the LSTM to False, so the output is a simple vector, which serves as the input of the subsequent dense linear output layer.

3D CNN + ConvLSTM: Our main goal in this chapter was to examine the efficiency of the ConvLSTM layer for this task and improve the model training time by using a optimised network which is using both convolution and sequence processing in one layer. In the first experiment we applied it only at the topmost hidden layer of our 3D-CNN model (see Table 6.2). As the ConvLSTM layer implements the operation of a convolutional and an LSTM layer in one, we replaced the uppermost Conv3D and LSTM layers by a ConvLSTM layer, reducing the number of neural hidden layers from 5 to 4. Also, as the ConvLSTM layer works with matrices and also outputs matrices, the reshaping was required after the layer and not before it.

3D-CNN + ConvLSTM

Conv3D(30,(5,13,13),strides=(s,2,2))
Dropout(0.35)

Conv3D(60,(1,13,13),strides=(1,2,2))
Dropout(0.35)
MaxPooling3D(poolsize =(1,2,1))

Conv3D(90,(1,13,13),strides=(1,2,2))
Dropout(0.35)

ConvLSTM2D(64, (3,3), Strides=(2,2), ret_seq=False)
Flatten()

Dense(80,activation='linear')

Table 6.2: *The ConvLSTM model is implemented by combining a convolutional neural network (CNN) with a long short-term memory (LSTM) network, where the CNN extracts spatiotemporal features from the input data and the LSTM processes them along the temporal axis.*

6.4 Results

In the first experiment we compared the performance of the baseline 3D-CNN model we the two hybrid solutions proposed in the previous chapter and in Table 6.2 and Table 6.1. In Table 6.3 we report two simple objective metrics of the quality of training, the mean squared error and the R^2 score, which is popular in regression tasks implemented with neural networks (for R^2 a higher value means better performance). As can be seen, replacing the dense layer by the LSTM layer already brings a slight but consistent improvement in the results, both on the development and on the test set. Fusing the uppermost Conv3D and the LSTM layer into a ConvLSTM layer resulted in further error reduction of about the same rate, even though the network depth is decreased. This clearly proves the efficiency of the ConvLSTM layer. However, we also observed a drawback, namely that the ConvSLTM layer has much more trainable parameters than the Conv3D layer. Hence, we had to reduce the filter size in the ConvLSTM layer, in order to keep the number of parameters in the original range. Theoretically, similar to the LSTM layer, the ConvLSTM layer can also be made bidirectional. However, we ran into the same problem that it tremendously increased the number of parameters while yielding only a marginal improvement. Thus, we stuck with using the unidirectional variant. Finally, to fuse the Conv3D and the LSTM layers, we had to remove the second MaxPooling layer. We also tried to insert it back after the ConvLSTM layer, but the results did not change considerably.

Obviously, many other possible configurations exist that combine Conv3D and ConvLSTM layers. In the second experiment we tried out further combinations of these two layers. We experimented with 4 hidden layer constructs, and we fixed the

Network Type	Dev		Test	
	MSE	Mean R^2	MSE	Mean R^2
3D-CNN	0.292	0.714	0.293	0.710
3D-CNN + BiLSTM	0.285	0.721	0.282	0.721
3D-CNN + ConvLSTM	**0.276**	**0.727**	**0.276**	**0.73**

Table 6.3: *The MSE and mean R^2 scores obtained with the various network configurations for the development and test sets, respectively. The best results are highlighted in bold.*

Layer 1	Layer 2	Layer 3	Layer 4	Dev	Test
ConvLSTM	ConvLSTM	ConvLSTM	ConvLSTM	0.31	0.31
ConvLSTM	ConvLSTM	ConvLSTM	—	0.29	0.3
Conv3D	ConvLSTM	ConvLSTM	ConvLSTM	0.31	0.31
Conv3D	Conv3D	ConvLSTM	ConvLSTM	0.36	0.35
Conv3D	**Conv3D**	**Conv3D**	**ConvLSTM**	**0.27**	**0.27**
Conv3D	ConvLSTM	Conv3D	ConvLSTM	0.3	0.3
ConvLSTM	Conv3D	Conv3D	ConvLSTM	0.34	0.34

Table 6.4: *The MSE for different combinations of Conv3D and ConvLSTM layers in the four hidden layers of the network. The best results are highlighted in bold.*

uppermost layer to be a ConvLSTM, as it convincingly proved to be the better setup in the previous experiment. Table 6.4 summarizes the architectures we experimented with. As regards ConvLSTM layers, the returns sequences parameter was set to True for intermediate layers, and set to False only when the ConvLSTM layer was the topmost hidden layer. The meta-parameters were always chosen so that the global count of the free parameters stayed similar to that of the baseline model.

Seeing the good performance of the ConvLSTM layer in the previous experiment, we first tried to build a fully ConvLSTM model. However, as the first row of Table 6.4 shows, we obtained no improvement. As the ConvLSTM layer proved to be more efficient than the Conv3D layer earlier, next we tried to create a network of just 3 ConvLSTM layers instead of 4. As shown in the second row of the table, the results became slightly better, but still worse than the baseline. This result reinforces our previous observation that ConvLSTM networks do not require the same depth as a convolutional network.

Conv3D layers have the advantage that they can easily downsample the time axis using a stride parameter larger than 1. On the contrary, ConvLSTM units cannot easily skip elements of their input sequence, due to their recurrent nature, which results in a large parameter count and a slow training. Hence, it seemed to be more efficient to put a Conv3D layer into the first hidden layer. We tried to place Conv3D layers in the lower layers, and ConvLSTM layers in the remaining layers. The middle block of Table 6.4 shows that the optimal solution is to have just one ConvLSTM layer, as in our original experiment. Lastly, we tried two further configurations with alternating Conv3D and ConvLSTM layers, motivated by papers like [59, 118], but we did not receive any better results.

6.5 Summary

Here, we were seeking the optimal neural network architecture for the articulatory-to-acoustic mapping task of SSI systems. The task involves the processing of 3D data blocks – sequences of images – for which one can apply 3D-CNN models, such as in chapter 2. Alternatively, one may apply a ConvLSTM model proposed by [91]. Besides comparing the purely convolutional and ConvLSTM models, we also experimented with hybrid architectures where the two layers types are mixed. The 3D-CNN + ConvLSTM hybrid model obtained the best results, better than the baseline 3D-CNN model, and it also outperformed other models with a different order of layers, as applied in [59] for emotion recognition, and in [118] for the prediction of the subsequent ultrasound image. Applying the ConvLSTM layer in the uppermost hidden layer even made the model smaller (with one hidden layer) and slightly faster to train. The optimal model arrangement consists of three Conv3D layers and a (ConvLSTM) on top of them, which illustrates that it is worth combining the ConvLSTM layer with other layer types such as the Conv3D to extract spatio-temporal features from videos – in our case, to better capture the tongue movement. The winning architecture also shows that the Conv3D blocks are more efficient in extracting local spectro-temporal information, while ConvLSTM is more efficient in fusing these pieces of information along the time axis. Interestingly, this coincides with the observation of Tran et al. about the optimal order of feature extraction for 3D video blocks [104].

The author of this PhD thesis is responsible for the following contributions presented in this chapter:

VI/1. Preparing data for the specific task.

VI/2. Implementing code for the models (Conv3D, Conv3D+BiLSTM, ConvLSTM).

VI/3. Analyzing and interpreting the results.

VI/4. Calculating the evaluation metric for the results.

And the results presented in this chapter were published in [38].

Chapter 7

Enhancing Tongue Ultrasound-Based Silent Speech Interfaces with Spatial Transformer Networkss

In the previous chapters, we explored the generation of speech from articulatory signals associated with speech production. As discussed in Chapter 1, these signals can be recorded using various devices such as ultrasound, EEG, fMRI, etc. All of the articulatory tracking devices are highly sensitive to 1) the alignment of the recording equipment, 2) the actual speaker's anatomy. For example, in the case of ultrasound recordings, the probe fixing headset has to be remounted onto the speaker for each recording session. This inevitably causes the recorded ultrasound videos to become misaligned between each recording session [17]. Moreover, there are large individual differences across speakers, so even a system trained on the data of several speakers may still perform poorly for a new speaker.

There have already been several cross-session and cross-speaker studies, of which we mention only those related to imaging. To handle the session dependency of UTI-based synthesis, Gosztolya et al. used data from different sessions [28]. Ribeiro et al. reported that, for a speaker-independent system, unsupervised model adaptation can improve the results for silent speech [83]. In a multi-speaker framework, in Chapter 4 we experimented with the use of x-vectors features extracted from the speakers, leading to a marginal improvement in the spectral estimation step [37]. Zhang et al. evaluated UTI and lip video based unconstrained multi-speaker voice recovery with a transfer learning strategy and encoder-decoder architecture [114]. There have been more studies on multi-speaker lip-to-speech synthesis [67, 73, 87, 107]. One of the first papers that could produce intelligible speech for unseen speakers was based on WGAN with new additional critics and losses [67]. Another study proposed speaker disentanglement by inputting speaker identities or embeddings to the DNN [73]. An end-to-end ResNet-based model was claimed to outscore previous approaches on unseen speakers [87].

Most of the above approaches hope to solve speaker sensitivity simply by acquiring articulatory training data from a large quantity of speakers. In this chapter, we experiment with a direct adaptation of an UTI-based SSI network to the actual speaker or session. To avoid the need for a full retraining, we extend our network with a spatial transformer network (STN) module and retrain only this module during the adaptation step. The STN learns an affine transformation on the input images, and our assumption is that this transformation should mostly be able to compensate for the misalignment of the recording device, and to a certain extent also for the inter-speaker differences.

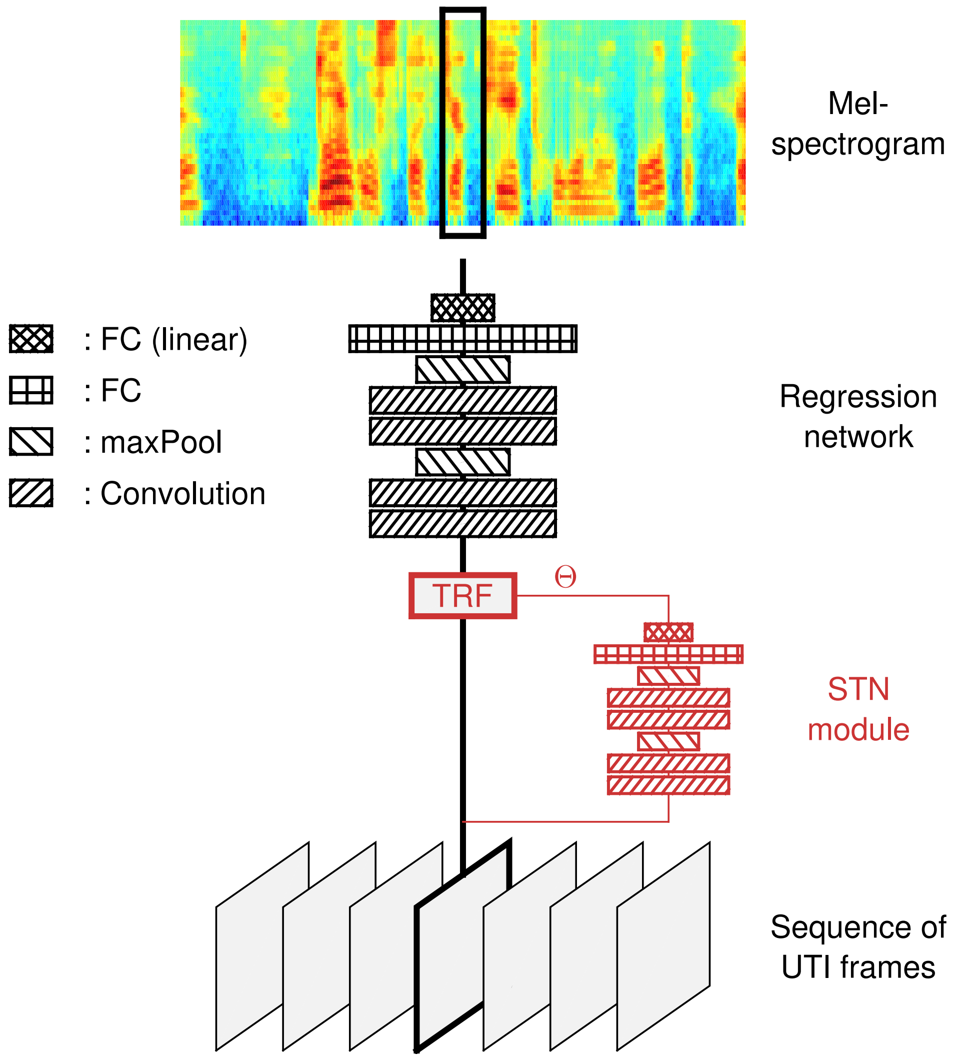
7.1 The UTI-to-Speech framework

Approaches for articulatory-to-acoustic mapping were described in the previous chapters and so we just give a brief overview here (see also Fig. 7.3). The input to our system is a sequence of ultrasound tongue images, which record the movement of the articulators at a relatively high frame rate. Our goal is to estimate the speech signal produced during articulation, but instead of producing a speech signal, we estimate only a mel-spectrogram at the network output, and we apply a neural vocoder to convert the spectrogram to speech. The main advantage of this approach is that the mel-spectrum is a very dense representation, which is easier to estimate than the speech signal itself, and we can apply large pre-trained networks for the synthesis step, such as WaveGlow [75]. Our training data consists of precisely time-aligned pairs of ultrasound videos and speech signals, so the ultrasound-to-spectrogram mapping can be performed on a simple frame-by-frame basis, converting each ultrasound image to a spectral vector [18, 100]. While this simple arrangement already performs reasonably well, significant improvement can be achieved by involving the input context, that is, by using a block of video frames as input instead of just one image. Several network architectures have been proposed to process 3D blocks of input data, for video processing in general [24, 48, 104], and for ultrasound input in particular [53, 86, 102, 113]. In the experimental section we will experiment both with 2D and 3D Convolutional Neural Networks (CNNs) for the mapping task. The problem could also be addressed even in the lack of aligned training data using encoder-decoder networks [83, 114] or video transformers [8, 90].

7.2 The Spatial Transformer Network

The concept of the Spatial Transformer Network (STN) was motivated by the fact that in image recognition the actual input is frequently shifted, rotated or scaled

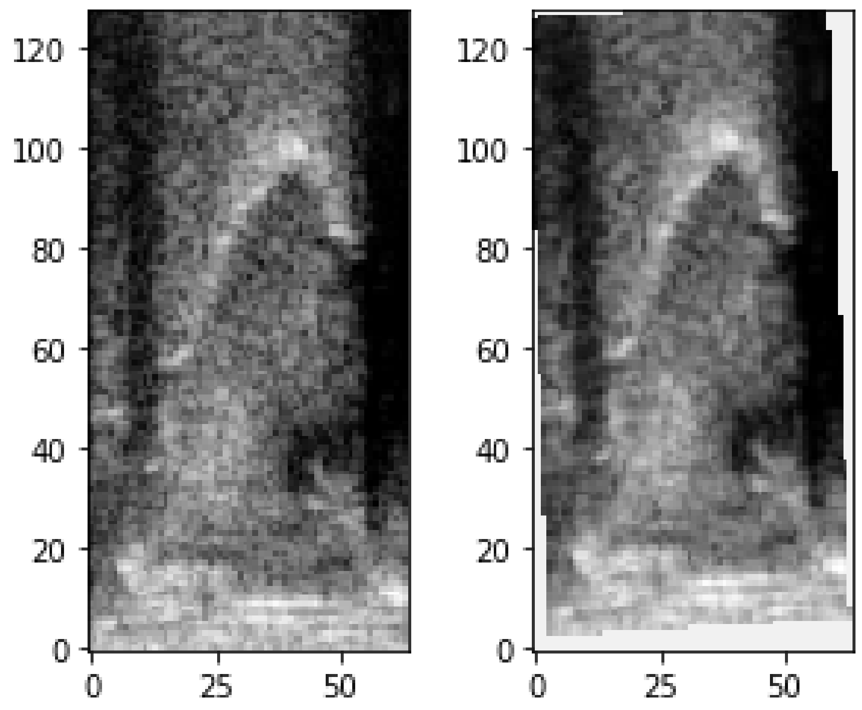
Figure 7.1: *Illustration of the network architecture used.*



compared to the training data, due to changes in camera angle, distance, and other factors [44]. Although CNNs are somewhat invariant to translations, their overall flexibility would greatly improve by introducing a dynamic mechanism that can spatially transform the input image by an appropriate transformation before classification. The STN is a network module that can be inserted into a CNN architecture at any point, but typically it is applied between the input and the first network layer. In this arrangement, the STN performs an affine transformation on the input image, and returns a transformed image of the same size. As it manipulates only the input, it can be combined with practically any type of classification or regression network. The affine transform defined by the STN is quite powerful and includes translation, scaling, rotation, shearing and cropping as special cases.

The STN consists of three main parts, namely the localization network, the grid

Figure 7.2: *An example UTI image, before and after STN.*



generator and the sampler [44]. The grid generator and the sampler together are responsible for executing the affine transformation defined by the parameter θ, which consists of 6 components in the case of a 2D input, (for example see Fig. 7.3).

The grid generator and the sampler are differentiable. This is vital for propagating the error back to the third component, the localization network, which learns to estimate the θ parameters, conditioned on the actual input image. The localization network can take any form, but its uppermost layer must be a regression layer to produce the θ values.

While we explained the STN concept assuming 2D images, the whole idea can be naturally extended to 3D data blocks [6, 44]. The 3D variant has already been used in visual speech recognition (e.g., lip reading) by Yu and Wang [112]. We discuss the options for a 3D extension in Section 7.5.3.

Although the original concept uses the STN to decrease the variance of the training data by transforming the input images to a canonical, expected pose, it may also be useful in handling a domain mismatch between the training and testing conditions. It was applied for domain adaptation in various image processing situations [14, 84], and here we evaluate its efficiency for speaker and session adaptation in UTI-to-speech conversion.

7.3 Experimental Set-Up

7.3.1 Data Acquisition and Preprocessing

The data utilized in this chapter consisted of the Hungarian dataset, which was previously introduced in Chapter 6.

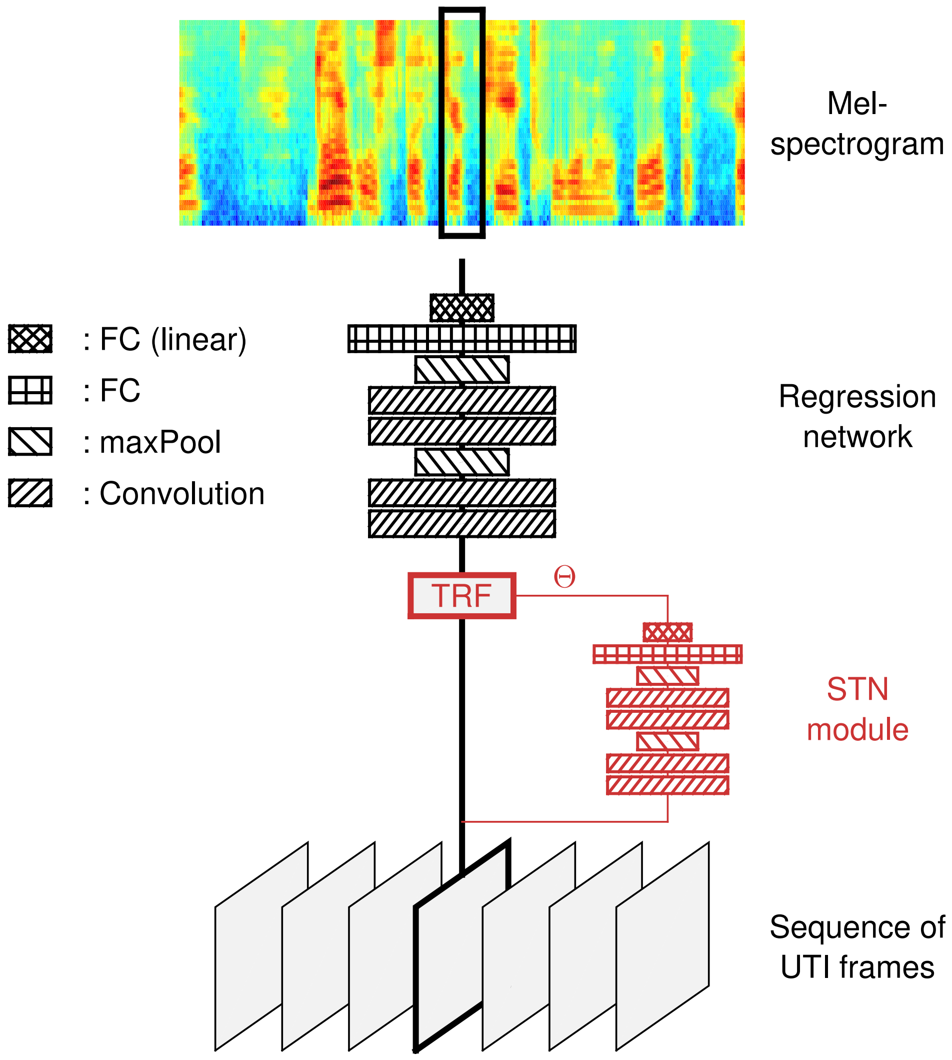
7.3.2 Network Configuration

In the first set of experiments we applied a simple 2D convolutional (2D-CNN) network that transforms one ultrasound image to one spectral vector, and the localization network of the STN also was a 2D-CNN. Although this approach is suboptimal, it may already show whether the STN is suitable to perform domain adaptation, while visualization and interpretation is easier in 2D. Fig. 7.3). The input to our system is a sequence of ultrasound tongue images, which record the movement of the articulators at a relatively high frame rate. Our goal is to estimate the speech signal produced during articulation, but instead of producing a speech signal, we estimate only a mel-spectrogram at the network output, and we apply a neural vocoder to convert the spectrogram to speech. The main advantage of this approach is that the mel-spectrum is a very dense representation, which is easier to estimate than the speech signal itself, and we can apply large pre-trained networks for the synthesis step, such as WaveGlow [75]. Our training data consists of precisely time-aligned pairs of ultrasound videos and speech signals, so the ultrasound-to-spectrogram mapping can be performed on a simple frame-by-frame basis, converting each ultrasound image to a spectral vector [18, 100]. While this simple arrangement already performs reasonably well, significant improvement can be achieved by involving the input context, that is, by using a block of video frames as input instead of just one image. Several network architectures have been proposed to process 3D blocks of input data, for video processing in general [24, 48, 104], and for ultrasound input in particular [53, 86, 102, 113]. In the experimental section we will experiment both with 2D and 3D Convolutional Neural Networks (CNNs) for the mapping task. The problem could also be addressed even in the lack of aligned training data using encoder-decoder networks [83, 114] or video transformers [8, 90].

7.4 The Spatial Transformer Network

The concept of the Spatial Transformer Network (STN) was motivated by the fact that in image recognition the actual input is frequently shifted, rotated or scaled compared to the training data, due to changes in camera angle, distance, and other factors [44]. Although CNNs are somewhat invariant to translations, their over-

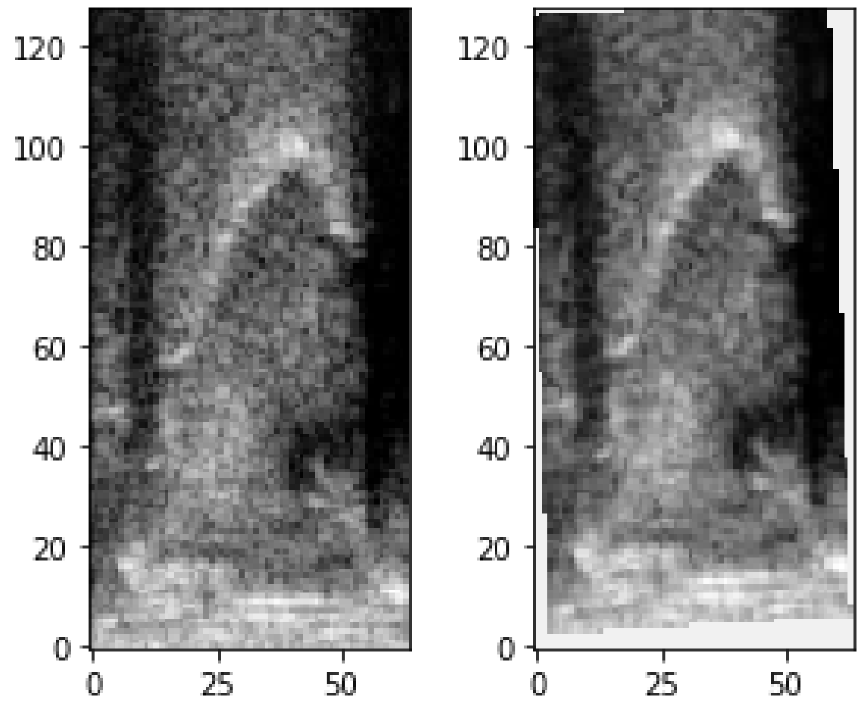
Figure 7.3: *Illustration of the network architecture used.*



all flexibility would greatly improve by introducing a dynamic mechanism that can spatially transform the input image by an appropriate transformation before classification. The STN is a network module that can be inserted into a CNN architecture at any point, but typically it is applied between the input and the first network layer. In this arrangement, the STN performs an affine transformation on the input image, and returns a transformed image of the same size. As it manipulates only the input, it can be combined with practically any type of classification or regression network. The affine transform defined by the STN is quite powerful and includes translation, scaling, rotation, shearing and cropping as special cases.

The STN consists of three main parts, namely the localization network, the grid generator and the sampler [44]. The grid generator and the sampler together are responsible for executing the affine transformation defined by the parameter θ, which

Figure 7.4: *An example UTI image, before and after STN.*



consists of 6 components in the case of a 2D input (see Fig. 7.3).

The grid generator and the sampler are differentiable. This is vital for propagating the error back to the third component, the localization network, which learns to estimate the θ parameters, conditioned on the actual input image. The localization network can take any form, but its uppermost layer must be a regression layer to produce the θ values.

While we explained the STN concept assuming 2D images, the whole idea can be naturally extended to 3D data blocks [6, 44]. The 3D variant has already been used in visual speech recognition (e.g., lip reading) by Yu and Wang [112]. We discuss the options for a 3D extension in Section 7.5.3.

Although the original concept uses the STN to decrease the variance of the training data by transforming the input images to a canonical, expected pose, it may also be useful in handling a domain mismatch between the training and testing conditions. It was applied for domain adaptation in various image processing situations [14, 84], and here we evaluate its efficiency for speaker and session adaptation in UTI-to-speech conversion.

7.5 Experimental Set-Up

7.5.1 Data Acquisition and Preprocessing

The data utilized in this chapter consisted of a Hungarian dataset, which was previously explained in Chapter 6.

7.5.2 Network Configuration

For the data In the first set of experiments we applied a simple 2D convolutional (2D-CNN) network that transforms one ultrasound image to one spectral vector, and the localization network of the STN also was a 2D-CNN. Although this approach is suboptimal, it may already show whether the STN is suitable to perform domain adaptation, while visualization and interpretation is easier in 2D. Fig. 7.3 illustrates this network arrangement, while Fig. 7.4 demonstrates the effect of STN on an actual ultrasound image. The 2D-CNN network had a quite simple and traditional structure. It consisted of 4 convolutional layers with 30-60-90-120 filters, with a MaxPooling layer after every second layer. The convolutional processing was followed by a fully connected layer of 300 neurons, and the linear output layer of 80 neurons. All hidden layers applied the Swish nonlinearity, and overfitting was minimized by placing dropout layers (p=0.2) after each processing layer. We applied the mean squared error (MSE) loss function, which was minimized using the Adam optimizer with a batch size of 100 and a learning rate of $2 \cdot 10^{-3}$. Training was halted using early stopping on the validation set.

The input of the STN module is the same image as that of the spectral regression module, and it also has to perform regression. Hence we used exactly the same 2D-CNN architecture for the STN as for the main network. However, as the STN has to estimate only 6 parameters (the θ vector) instead of 80, we configured it to be much smaller than the regression network. The number of free parameters in the STN module was only about 10% of that of the whole network.

7.5.3 Extension to 3D

The simple framework presented above can be significantly improved by extending the input from just one image to a sequence of video frames. Such 3D blocks of input data can be processed by various network types, such as by combining 2D-CNN and LSTM layers, by 3D-CNN models, or by using Convolutional LSTM layers (see previous chapters) [53, 86, 102, 117]. Here, we applied the 3D-CNN architecture that we used in Chapter 2. This network had the same global architecture as the 2D-CNN presented above. However, the convolutions were extended to the time axis as well, in a somewhat special arrangement. The first Conv3D layer applied a large stride along the time axis, dividing the sequence of 25 input images into 5 five blocks along time. The next two Conv3D layers had a filter size of 1 along time, practically processing the 5 blocks separately. Finally, the extracted information was fused along time by the last Conv3D layer and by the topmost two fully connected layers.

As regards the STN, there are several options to extend it to 3D. First, we could reformulate the affine transformation to operate on 3D data. However, for our ultrasound data, translation, shearing and rotation along the time axis seemed to be unnecessary. Hence, we chose to keep the transformation in 2D. The second question was whether the localization network should operate on 2D or 3D data.

We chose the technically simpler solution, and used the same 2D localization network as for the 2D-CNN. In this configuration, the STN applies the same 2D transformation to all the 25 images of the actual input block.

7.6 Results

Table 7.1: *MSE rates of the 2D-CNN on the dev set of the 4 speakers (top), and for the 4 extra sessions of speaker 048 (down).*

	s048	s049	s102	s103
no STN	0.361	0.387	0.390	0.343
with STN	0.358	0.396	0.389	0.340

	s048-2	s048-3	s048-4	s048-5
no STN	0.505	0.462	0.464	0.504
with STN	0.495	0.458	0.467	0.483

First of all, we examined whether the inclusion of the STN module has any positive effect on the results when using only training data from one speaker and one session. Table 7.1 shows the MSE scores of the 2D network, with and without the STN module, for all the 4 speakers and the 4 extra sessions from speaker 048. In this configuration, we expected no significant benefit from the STN, and this is exactly what we see. The scores we obtained are quite similar accross all speakers and the extra sessions of speaker 048. One notable difference is that the results are consistently worse for the additional sessions of speaker 048 than for her main session. The explanation is that these extra sessions were much shorter than the multi-speaker recordings (3.5 vs. 15 minutes).

In the adaptation experiments we considered the network trained on the data of speaker 048 as our base model. First, we evaluated it on the samples from the other speakers and the additional sessions of the same speaker without any adaptation. These results will be considered as the base results.

	Adaptation method				
Speaker	no	stn	stn+out	full	mean θ
spk049	1.049	0.588 (-71%)	0.517 (-82%)	0.400	0.887 (-25%)
spk102	1.401	0.609 (-78%)	0.449 (-94%)	0.389	1.015 (-38%)
spk103	1.322	0.552 (-79%)	0.469 (-88%)	0.350	0.909 (-42%)
avg.diff.	**-0%**	**-76%**	**-88%**	**-100%**	**-35%**

	Adaptation method				
Session	no	stn	stn+out	full	mean θ
ses-2	1.131	0.646 (-77%)	0.547 (-93%)	0.503	0.913 (-34%)
ses-3	0.998	0.619 (-69%)	0.485 (-94%)	0.451	0.934 (-11%)
ses-4	1.054	0.641 (-70%)	0.522 (-90%)	0.468	0.908 (-24%)
ses-5	1.174	0.604 (-85%)	0.566 (-91%)	0.506	0.955 (-32%)
avg.diff.	**-0%**	**-75%**	**-92%**	**-100%**	**-26%**

Table 7.2: *MSE rates of the 2D-CNN model of spk048 for the other 3 speakers (top), and for the 4 extra sessions of the same speaker (down), using various adaptation strategies.*

Table 7.2 summarizes the results achieved with the different adaptation strategies for the additional speaker (upper panel) and for the additional sessions of the same speaker (lower panel). As the relative improvements are more informative than the actual MSE values, in parenthesis we display the relative error reductions (considering the no-adaptation case as 0% and full retraining as 100%), and in the bottom line we summarize the average improvements. The first column ('no' adaptation) clearly shows that the results are unacceptable without any adaptation – the error rates are actually worse than those for a randomly initialize net without any training. Moreover, the scores in the 'full' adaptation column are pretty similar to those in the baseline table, so pre-training on speaker 048 seems to be neither beneficial nor detrimental (interestingly even for the extra sessions from the same speaker). By allowing only the STN module to adapt ('stn' column) we can get rid of 75-76% of the performance gap between the non-adapted and fully adapted models. This is pretty good, considering that the regression network itself is not modified at all, we just simply allow the STN to learn a more optimal affine transformation for the images of the given speaker or session. The improvement is even larger if we also allow the linear output layer of the regression network to adjust its weights to the new speaker or session ('stn+out'). In this case, the average error reduction is 88% for cross-speaker and 92% for cross-session adaptation. The better cross-session score is reasonable, as one would expect that the differences caused by the misalignment of the device might be easier to compensate by an affine transformation than the inherent anatomical differences between speakers.

As in the baseline evaluation (Table 7.1) the STN had no effect on the results, we speculated that maybe it is unnecessary to learn a separate transformation for *each image* – perhaps learning one global θ for *each speaker or session* would be enough. We examined the within-speaker variance of the 6 components of theta for the baseline models, and we indeed found that it was very low, on the order of 0.01, while the between-speaker variance of the θ vectors was 3-5 times larger. This fact seemed to underpin our conjecture, so we preformed the following simple experiment. Instead of using a unique theta for each image, we simply replaced the STN of the adapted models with the mean θ vector over the samples of the given speaker/session. The results are shown in the rightmost columns of Table 7.2 ('mean θ'), but the findings are negative: while there is a 25-35% reduction in the error rates, it is very far from the best achievable. It seems that even though the variance of θ is small, the minor nuances in the learned transformations per image play an important role in the regression accuracy.

Finally, we extended our experiments to 3D input blocks consisting of 25 subsequent images, using a 3D-CNN for the regression task and a 2D-CNN for the STN module (Section 7.5.3). In this case, the baseline model yielded an MSE of 0.275 and 0.278 with and without the STN on the data set of speaker 048. As speaker and session adaptation gave very similar results previously, we repeated only the cross-speaker adaptation experiments with this network, and the results are shown in Table 7.3. While all the error rates are typically lower than they were for the 2D-CNN, the relative improvements with respect to the various adaptation strategies are very similar.

7.7 Summary

Current tongue ultrasound-based SSI systems are sensitive to changing speakers, or to a slight displacement of the recording device. Here, we examined whether an STN module is able to counterbalance these factors by applying an affine transform on the

	Adaptation method			
Speaker	no	stn	stn+out	full
spk049	1.105	0.553 (-73%)	0.497 (-80%)	0.348
spk102	1.451	0.502 (-84%)	0.416 (-91%)	0.315
spk103	1.541	0.501 (-83%)	0.418 (-90%)	0.294
avg.diff.	**-0%**	**-80%**	**-87%**	**-100%**

Table 7.3: *MSE rates for cross-speaker adaptation, using the 3D-CNN network.*

input image. When applying a simple 2D-CNN for spectral estimation, we found that allowing only the adaptation of the STN module can decrease the error rate by about 75%, while allowing also the linear output layer to adapt can compensate for 88-92% of the error. We extended the experiments to 3D input blocks, and we observed similar tendencies, although the improvement was somewhat smaller. In the future we plan to run experiments with a 3D localization network and with smaller amounts of adaptation material.

The author of this PhD thesis is responsible for the following contributions presented in this chapter:

VII/1. Preparing data .

VII/2. Implementing code for the models.

VII/3. Analyzing and interpreting the results.

And the results presented in this chapter were published in [103].

Bibliography

[1] WebRtc voice activity detection. `https://webrtc.org`, 1999.

[2] ITU-R recommendation P.862 : Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs, 2001.

[3] Convolutional LSTM. `https://medium.com/neuronio/an-introduction-to-convlstm-55c9025563a7`, 2019.

[4] Recurrent neural networks and LSTMs with keras. `https://blog.eduonix.com/artificial-intelligence/recurrent-neural-networks-lstms-keras`, 2020.

[5] B. Atal and L. Rabiner. A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(3):201–212, 1976.

[6] A. Bas, P. Huber, W. Smith, M. Awais, and J. Kittler. 3d morphable models as spatial transformer networks. In *Proc. Int. Conf. on Computer Vision (ICCV) Workshops*, pages 895–903, 2017.

[7] A. Benyassine, E. Shlomot, HY. Su, D. Massaloux, C. Lamblin, and JP. Petit. A silence compression scheme for use with g. 729 optimized for v. 70 digital simultaneous voice and data applications (recommendation g. 729 annex b). *IEEE Communications Magazine*, 35(9):64–73, 1997.

[8] G. Bertasius, H. Wang, and L. Torresani. Is space-time attention all you need for video understanding. *Proc. ICML*, 139:813–824, 2021.

[9] M. C. Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[10] J. Bradbury. Linear predictive coding. *Mc G. Hill*, 2000.

[11] J. Bradbury, S. Merity, C. Xiong, and R. Socher. Quasi-recurrent neural networks. In *Proc. ICLR*, 2017.

[12] B. Cao, A. Wisler, and J. Wang. Speaker adaptation on articulation and acoustics for articulation-to-speech synthesis. *Sensors*, 22(16):6056, 2022.

[13] K. Cho, B. van Merrienboer, C. Gülcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proc. EMNLP*, pages 1724–1734, 2014.

[14] W. Chu, W. C. Hung, Y. H. Tsai, D. Cai, and M. H. Yang. Weakly-supervised caricature face parsing through domain adaptation. In *Proc. Int. Conf. on Image Processing (ICIP)*, pages 3282–3286, 2019.

[15] T. G. Csapó, T. Grósz, G. Gosztolya, L. Tóth, and A. Markó. DNN-based ultrasound-to-speech conversion for a silent speech interface. In *Proc. Interspeech*, pages 3672–3676, 2017.

[16] T. G. Csapó, T. Grósz, G. Gosztolya, L. Tóth, and A. Markó. DNN-based ultrasound-to-speech conversion for a silent speech interface. In *Proc. Interspeech*, pages 3672–3676, 2017.

[17] T. G. Csapó and K. Xu. Quantification of Transducer Misalignment in Ultrasound Tongue Imaging. In *Proc. Interspeech*, pages 3735–3739, Shanghai, China, 2020.

[18] T. G. Csapó, C. Zainkó, L. Tóth, G. Gosztolya, and A. Markó. Ultrasound-Based Articulatory-to-Acoustic Mapping with WaveGlow Speech Synthesis. In *Proc. Interspeech 2020*, pages 2727–2731, 2020.

[19] N. Dehak, R. Dehak, P. Kenny, N. Brummer, P. Ouellet, and P. Dumouchel. Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification. In *Proc. Interspeech*, pages 1559–1562, 2009.

[20] B. Denby, T. Schultz, K. Honda, T. Hueber, J. M. Gilbert, and J. S. Brumberg. Silent speech interfaces. *Speech Communication*, 52(4):270–287, 2010.

[21] H. Deng and D. O'Shaughnessy. Voiced-unvoiced-silence speech sound classification based on unsupervised learning. In *2007 IEEE International Conference on Multimedia and Expo*, pages 176–179. IEEE, 2007.

[22] çois F. Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[23] M. J. Fagan, S. R. Ell, J. M. Gilbert, E. Sarrazin, and P. M. Chapman. Development of a (silent) speech recognition system for patients following laryngectomy. *Medical Engineering and Physics*, 30(4):419–425, 2008.

[24] C. Feichtenhofer, H. Fan, J. Malik, and K. He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6202–6211, 2019.

[25] Jose A. Gonzalez, Lam A. Cheah, Angel M. Gomez, Phil D. Green, James M. Gilbert, Stephen R. Ell, Roger K. Moore, and Ed Holdsworth. Direct speech reconstruction from articulatory sensor data by machine learning. *IEEE/ACM Trans. ASLP*, 25(12):2362–2374, 2017.

[26] Jose A. Gonzalez-Lopez, Alejandro Gomez-Alanis, Juan M. Martin Donas, Jose L. Perez-Cordoba, and Angel M. Gomez. Silent Speech Interfaces for Speech Restoration: A Review. *IEEE Access*, 8:177995–178021, 2020.

[27] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[28] G. Gosztolya, T. Grósz, L. Tóth, A. Markó, and T. G. Csapó. Applying DNN Adaptation to Reduce the Session Dependency of Ultrasound Tongue Imaging-Based Silent Speech Interfaces. *Acta Polytechnica Hungarica*, 17(7):109–124, 2020.

[29] P. Govalkar, J. Fisher, F. Zalkov, and C. Dittmar. A comparison of recent neural vocoders for speech signal reconstruction. In *Proc. ISCA Speech Synthesis Workshop*, 2019.

[30] T. Grósz, G. Gosztolya, L. Tóth, T. G. Csapó, and A. Markó. F0 estimation for DNN-based ultrasound silent speech interfaces. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 291–295. IEEE, 2018.

[31] T. Grósz, L. Tóth, G. Gosztolya, T. G. Csapó, and A. Markó. Kísérletek az alapfrekvencia becslésére mély neuronhálós, ultrahang-alapú némabeszéd-interfészekben. In *MSZNY 2018*, pages 196–205, Szeged, Hungary, 2018.

[32] JA. Haigh and JS. Mason. Robust voice activity detection using cepstral features. In *Proceedings of TENCon'93. IEEE Region 10 International Conference on Computers, Communications and Automation*, volume 3, pages 321–324. IEEE, 1993.

[33] G. Hinton, L. Deng, D. Yu, G. Dahl, AR. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[34] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[35] A. Honarmandi Shandiz and L. Tóth. Voice activity detection for ultrasound-based silent speech interfaces using convolutional neural networks. pages 499–510, 2021.

[36] A. Honarmandi Shandiz, L. Tóth, G. Gosztolya, A. Markó, and T. Gábor Csapó. Improving neural silent speech interface models by adversarial training. *arXiv e-prints*, pages arXiv–2104, 2021.

[37] A. Honarmandi Shandiz, L. Tóth, G. Gosztolya, A. Markó, and T. G. Csapó. Neural Speaker Embeddings for Ultrasound-Based Silent Speech Interfaces. In *Proc. Interspeech 2021*, pages 1932–1936, 2021.

[38] Amin Honarmandi Shandiz and L. Tóth. Improved processing of ultrasound tongue videos by combining ConvLSTM and 3D convolutional networks. In *Advances and Trends in Artificial Intelligence. Theory and Practices in Artificial Intelligence: 35th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2022, Kitakyushu, Japan, July 19–22, 2022, Proceedings*, pages 265–274. Springer, 2022.

[39] T. Hueber, G. Bailly, and B. Denby. Continuous articulatory-to-acoustic mapping using phone-based trajectory HMM for a silent speech interface. In *Proc. Interspeech*, pages 723–726, Portland, OR, USA, 2012.

[40] T. Hueber, E-L. Benaroya, G. Chollet, G. Dreyfus, and M. Stone. Development of a silent speech interface driven by ultrasound and optical images of the tongue and lips. *Speech Communication*, 52(4):288–300, 2010.

[41] K. Iskarous. Detecting the edge of the tongue: A tutorial. *Clinical Linguistics & Phonetics*, 19(6-7):555–565, jan 2005.

[42] P. Isola, J-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proc. Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017.

[43] ITU-R. ITU-R recommendation BS.1534: Method for the subjective assessment of intermediate audio quality, 2001.

[44] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *NIPS 28*, pages 2017–2025. 2015.

[45] M. Janke and L. Diener. EMG-to-speech: Direct generation of speech from facial electromyographic signals. *IEEE/ACM Trans. ASLP*, 25(12):2375–2385, 2017.

[46] M. Janke, M. Wand, K. Nakamura, and T. Schultz. Further investigations on EMG-to-speech conversion. In *Proc. ICASSP*, pages 365–368, 2012.

[47] A. Jaumard-Hakoun, K. Xu, C. Leboullenger, P. Roussel-Ragot, and B. Denby. An articulatory-based singing voice synthesis using tongue and lips imaging. In *Proc. Interspeech*, pages 1467–1471, 2016.

[48] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):221–231, 2013.

[49] L. R. Jonathan, W. Scott, H Erdogan, and J. R. Hershey. SDR - half-baked or well done. In *Proc. ICASSP*, 2019.

[50] E. M. Juanpere and T. G. Csapó. Ultrasound-based silent speech interface using convolutional and recurrent neural networks. *Acta Acustica united with Acustica*, 105(4):587–590, 2019.

[51] T. Kaneko, H. Kameoka, K. Tanaka, and N. Hojo. StarGAN-VC2: Rethinking conditional methods for StarGAN-based voice conversion. In *Proc. Interspeech*, pages 679–683, 2019.

[52] M. Kim, B. Cao, T. Mau, and J. Wang. Speaker-Independent Silent Speech Recognition From Flesh-Point Articulatory Movements Using an LSTM Neural Network. *IEEE/ACM Trans. ASLP*, 25(12):2323–2336, 2017.

[53] N. Kimura, M. Kono, and J. Rekimoto. Sottovoce: An ultrasound imaging-based silent speech interaction using deep neural networks. In *Proc. of CHI Conf. on Human Factors in Computing Systems*, 2019.

[54] J. Kominek, T. Schultz, and A.W. Black. Synthesizer voice quality of new languages calibrated with mean cepstral distortion. In *Proc. SLT*, pages 63–68, 2008.

[55] Y. Korkmaz and A. BOYACI. Examining vowels' formant frequency shifts caused by preceding consonants for turkish language. *Journal of Engineering and Technology*, 2(2):38–47, 2018.

[56] A. Krizhevsky, I. Sutskever, and GE. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105, 2012.

[57] R.F. Kubichek. Mel-cepstral distance measure for objective speech quality assessment. In *Proc. ICASSP*, pages 125–128, 1993.

[58] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brébisson, Y. Bengio, and A. C. Courville. MelGAN: Generative adversarial networks for conditional waveform synthesis. In *Advances in Neural Information Processing Systems*, volume 32, pages 14910–14921, 2019.

[59] S. Kwon et al. CLSTM: Deep feature-based speech emotion recognition using the hierarchical ConvLSTM network. *Mathematics*, 8(12):2133, 2020.

[60] T. László, Amin. Honarmandi Shandiz, G. Gábor, Z. Csaba, M. Alexandra, and C. T. Gábor. 3d konvolúciós neuronhálón és neurális vokóderen alapuló némabeszéd-interfész. *Proc. MSZNY*, pages 123–137, 2021.

[61] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu. Deep Speaker: an End-to-End Neural Speaker Embedding System. *arXiv e-prints*, page arXiv:1705.02304, 2017.

[62] Z-H. Ling, S-Y. Kang, H. Zen, A. Senior, M. Schuster, X-J. Qian, H. Meng, and L. Deng. Deep learning for acoustic modeling in parametric speech generation. *IEEE Signal Processing Magazine*, 32(3):35–52, 2015.

[63] Z. C. Liu, Z. H. Ling, and L. R. Dai. Articulatory-to-acoustic conversion using BLSTM-RNNs with augmented input representation. *Speech Communication*, 99(2017):161–172, 2018.

[64] N. N. Lokhande, N. S. Nehe, and P. S. Vikhe. Voice activity detection algorithm for speech recognition applications. In *IJCA Proceedings on International Conference in Computational Intelligence (ICCIA2012), vol. ICCIA*, number 6, pages 1–4, 2012.

[65] L. Maier-Hein, F. Metze, T. Schultz, and A. Waibel. Session independent non-audible speech recognition using surface electromyography. In *Proc. ASRU*, pages 331–336, 2005.

[66] J. Martín-Doñas, A. Gomez, J. Gonzalez Lopez, and A. Peinado. A deep learning loss function based on the perceptual evaluation of the speech quality. *IEEE Signal Processing Letters*, 25(11):1680 – 1684, 2018.

[67] R. Mira, K. Vougioukas, P. Ma, S. Petridis, B. W. Schuller, and M. Pantic. End-to-End Video-To-Speech Synthesis using Generative Adversarial Networks. *arXiv preprint arXiv:2104.13332*, 2021.

[68] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[69] M. H. Moattar, M. M. Homayounpour, and N. K. Kalantari. A new approach for robust realtime voice activity detection using spectral pattern. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4478–4481. IEEE, 2010.

[70] S Mondal and A. D. Barman. Clustering based voiced-unvoiced-silence detection in speech using temporal and spectral parameters. In *2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, pages 390–394. IEEE, 2015.

[71] B. Nirmalkar and S. Kumar. Voiced/unvoiced classification by hybrid method based on cepstrum and EMD. 2016.

[72] K. Okabe, T. Koshinaka, and K. Shinoda. Attentive statistics pooling for deep speaker embedding. In *Proc. Interspeech*, pages 2252–2256, 2018.

[73] D. Oneaţă, A. Stan, and H. Cucu. Speaker disentanglement in video-to-speech conversion. In *Proc. EUSIPCO*, pages 46–50, 2021.

[74] V. Peddinti, D. Povey, and S. Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Proc. Interspeech*, pages 3214–3218, 2015.

[75] R. Prenger, R. Valle, and B. Catanzaro. Waveglow: A flowbased generative network for speech synthesis. In *Proc. ICASSP*, pages 3617–3621, 2019.

[76] F. Qi, C. Bao, and Y. Liu. A novel two-step svm classifier for voiced/unvoiced/silence classification of speech. In *2004 International Symposium on Chinese Spoken Language Processing*, pages 77–80. IEEE, 2004.

[77] Y. Qi and B. R. Hunt. Voiced-unvoiced-silence classifications of speech using hybrid features and a network classifier. *IEEE Transactions on Speech and Audio Processing*, 1(2):250–255, 1993.

[78] L. R. Rabiner, R. W. Schafer, et al. *Digital processing of speech signals*. Prentice-hall, 1978.

[79] P. Ramachandran, B. Zoph, and Q. V. Le. Swish: a Self-Gated Activation Function. *ArXiv e-prints 1710.05941*, 2017.

[80] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In *International Conference on Machine Learning*, pages 1060–1069, 2016.

[81] M. Ribeiro, A. Eshky, K. Richmond, and S. Renals. Speaker-independent Classification of Phonetic Segments from Raw Ultrasound in Child Speech. In *Proc. ICASSP*, pages 1328–1332, 2019.

[82] M. Ribeiro, J. Sanger, J. Zhang, A. Eshky, A. Wrench, K. Richmond, and S. Renals. Tal: a synchronised multi-speaker corpus of ultrasound tongue imaging, audio, and lip videos. *arXiv preprint arXiv:2011.09804*, 2020.

[83] M. Sam Ribeiro, A. Eshky, K. Richmond, and S. Renals. Silent versus modal multi-speaker speech recognition from ultrasound and video. In *Proc. Interspeech*, 2021.

[84] R. Robinson, Qi. Dou, D. Coelho de Castro, K. Kamnitsas, M. de Groot, R. M. Summers, D. Rueckert, and B. Glocker. Image-level harmonization of multi-site data using image-and-spatial transformer networks. In *Proc. MICCAI*, pages 710–719, 2020.

[85] P. Saha and S. Fels. Learning Joint Articulatory-Acoustic Representations with Normalizing Flows. In *Proc. Interspeech*, pages 3196–3200, online, 2020.

[86] P. Saha, Y. Liu, B. Gick, and S. Fels. Ultra2speech – a deep learning framework for formant frequency estimation and tracking from ultrasound tongue images. In A.L. Martel, P. Abolmaesumi, D. Stoyanov, D. Mateus, M.A. Zuluaga, Kevin S.Z., D. Racoceanu, and L. Joskowicz, editors, *MICCAI 2020*, volume 12263 of *LNCS*, pages 473–482. Springer, 2020.

[87] N. Saleem, J. Gao, M. Irfan, E. Verdu, and J. P. Fuente. E2E-V2SResNet: Deep residual convolutional neural networks for end-to-end video driven speech synthesis. *Image and Vision Computing*, 119:104389, 2022.

[88] T. Schultz, M. Wand, T. Hueber, D. J. Krusienski, C. Herff, and J. S. Brumberg. Biosignal-based spoken communication: A survey. *IEEE/ACM Trans. ASLP*, 25(12):2257–2271, 2017.

[89] A Senior and I. Lopez-Moreno. Improving DNN speaker-independence with I-vector inputs. In *Proc. ICASSP*, pages 225–229, 2014.

[90] D. Serdyuk, O. Braga, and O. Siohan. Transformer-based video front-ends for audio-visual speech recognition. *arXiv preprint arXiv:2201.10439*, 2022.

[91] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *arXiv preprint arXiv:1506.04214*, 2015.

[92] S. Shon, H. Tang, and J. R. Glass. Frame-level speaker embeddings for text-independent speaker recognition and analysis of end-to-end model. In *Proc. SLT*, pages 1007–1013, 2018.

[93] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur. Spoken language recognition using x-vectors. In *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, pages 105–111, 2018.

[94] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur. X-vectors: Robust DNN embeddings for speaker recognition. In *Proc. ICASSP*, pages 5329–5333, 2018.

[95] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur. Deep neural network-based speaker embeddings for end-to-end speaker verification. In *Proc. (SLT)*, pages 165–170, 2016.

[96] S. Stone and P. Birkholz. Cross-speaker silent-speech command word recognition using electro-optical stomatography. In *Proc. ICASSP*, pages 7849–7853, online, 2020.

[97] C.H. Taal, R.C. Hendriks, R. Heusdens, and J. Jensen. An algorithm for intelligibility prediction of time–frequency weighted noisy speech. *IEEE Trans. ASLP*, 19(7):2125–2136, 2011.

[98] F. Taguchi and T. Kaburagi. Articulatory-to-speech conversion using bi-directional long short-term memory. In *Proc. Interspeech*, pages 2499–2503, 2018.

[99] M. Tatham. The place of electromyography in speech research. *Behav. Technol*, 6, 1971.

[100] E. Tatulli and T. Hueber. Feature extraction using multimodal convolutional neural networks for visual speech recognition. In *Proceedings of ICASSP*, pages 2971–2975, 2017.

[101] L. Tóth. Combining time- and frequency-domain convolution in convolutional neural network-based phone recognition. In *Proc. ICASSP*, pages 190–194, 2014.

[102] L. Tóth and Amin. Honarmandi Shandiz. 3D convolutional neural networks for ultrasound-based silent speech interfaces. In L. Rutkowski, R. Scherer, M. Korytkowski, W. Pedrycz, R. Tadeusiewicz, and J.M. Zurada, editors, *ICAISC 2020*, volume 12415 of *LNCS*, pages 159–169. Springer, 2020.

[103] L. Tóth, Amin. Honarmandi Shandiz, G. Gosztolya, and C. T. Gábor. Adaptation of tongue ultrasound-based silent speech interfaces using spatial transformer networks. *arXiv preprint arXiv:2305.19130*, 2023.

[104] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proc. CVPR*, 2018.

[105] E. Verteletskaya and K. Sakhnov. Voice activity detection for speech enhancement applications. *Acta Polytechnica*, 50(4), 2010.

[106] M. Wand, T. Schultz, and J. Schmidhuber. Domain-Adversarial Training for Session Independent EMG-based Speech Recognition. In *Proc. Interspeech*, pages 3167–3171, 2018.

[107] D. Wang, S. Yang, D. Su, X. Liu, D. Yu, and H. Meng. VCVTS: Multi-speaker Video-to-Speech synthesis via cross-modal knowledge transfer from voice conversion. In *Proc. ICASSP*, 2022.

[108] C. Wu, S. Chen, G. Sheng, P. Roussel, and B. Denby. Predicting tongue motion in unlabeled ultrasound video using 3D convolutional neural networks. In *Proc. ICASSP*, pages 5764–5768, 2018.

[109] Zhu. Y., T. Ko, D. Snyder, B. Mak, and D. Povey. Self-attentive speaker embeddings for text-independent speaker verification. In *Proc. Interspeech*, pages 3573–3577, 2018.

[110] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, et al. The htk book. *Cambridge university engineering department*, 3(175):12, 2002.

[111] T. Young, D. Hazarika, S. Poria, and E. Cambria. Recent trends in deep learning based natural language processing. *IEEE Comput. Intell. Mag.*, 13(3):55–75, 2018.

[112] J. Yu and S. Wang. Visual speech recognition in natural scenes based on spatial transformer networks. In *Proc. Int. Conf. on Anti-counterfeiting, Security, and Identification (ASID)*, 2020.

[113] C. Zainkó, L. Tóth, A. Honarmandi Shandiz, G. Gosztolya, A. Markó, G. Németh, and T. G. Csapó. Adaptation of Tacotron2-based Text-To-Speech for Articulatory-to-Acoustic Mapping using Ultrasound Tongue Imaging. In *Proc. ISCA SSW11*, pages 54–59, 2021.

[114] J. Zhang, P. Roussel, and B. Denby. Creating Song from Lip and Tongue Videos with a Convolutional Vocoder. *IEEE Access*, 9:13076–13082, 2021.

[115] S. Zhang, M. Lei, Z. Yan, and L. Dai. Deep-FSMN for large vocabulary continuous speech recognition. In *Proc. ICASSP*, 2018.

[116] Z. Zhang, C Deng, Y. Shen, D. S. Williamson, Y. Sha, Y. Zhang, H Song, and H. Li. On loss functions and recurrency training for GAN-based speech enhancement systems. In *Proc. Interspeech*, pages 3266–3270, 2020.

[117] C. Zhao, J. Zhang, C. Wu, H. Wang, and K. Xu. Predicting tongue motion in unlabeled ultrasound video using convolutional LSTM neural networks. In *Proc. ICASSP*, pages 5926–5930, 2019.

[118] C. Zhao, P. Zhang, J. Zhu, C. Wu, H. Wang, and K. Xu. Predicting tongue motion in unlabeled ultrasound videos using convolutional LSTM neural networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5926–5930. IEEE, 2019.

[119] S. Zhao, Y. Liu, Y. Han, R. Hong, Q. HU, and Q. Tian. Pooling the convolutional layers in deep convnets for video action recognition. *IEEE Trans. Circuits and Systems for Video Technology*, 28(8):1839–1849, 2018.

[120] Y. Zhao, Y. Xiong, and D. Lin. Trajectory convolution for action recognition. In *Advances in Neural Information Processing Systems 31*, pages 2204–2215, 2018.

[121] Y. Zhao, B. Xu, R. Giri, and T. Zhang. Perceptually guided speech enhancement using deep neural networks. In *Proc. ICASSP*, pages 5074–5078, 2018.

[122] J. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proc. Int. Conf. on Computer Vision*, pages 2223–2232, 2017.

Summary

This thesis presented new strategies for improving various aspects of the ultrasound SSI project, including model implementation and enhancement, data preparation and processing, generalization, and model training speedup. The proposed methods were tested on two large datasets, and the results were evaluated. Chapter 1 provided a brief introduction to the basic components of the SSI system, such as the feature extractor, different modalities, and model training. The chapter also described how deep neural networks, particularly CNNs, work and their usefulness in image processing. In subsequent chapters, various aspects of the SSI system were examined, and algorithms from related areas were adapted to improve the performance of the SSI system.

3D Convolutional Neural Networks for Ultrasound-Based Silent Speech Interfaces

In Chapter 2, we used deep neural networks to convert ultrasound video of tongue movements into speech. We used convolutional neural networks (CNNs) to process the sequence of images, which is popular for image recognition. The input was a video sequence containing time trajectory information of tongue movements. Then We explored different network structures for processing time sequences, including stacking a 2D CNN and a recurrent neural network, and extending the 2D CNN to 3D by adding time as an extra dimension. We found that the 3D CNN model achieved a lower error rate, was smaller, and had faster training than the CNN+LSTM model, suggesting that 3D CNNs are a feasible alternative to recurrent neural models for ultrasound video-based speech synthesis interface (SSI) systems. This chapter represents a significant contribution to the field of speech synthesis, demonstrating the potential for using deep neural networks to convert ultrasound video into speech. The use of CNNs for processing time sequences offers a new approach that can achieve lower error rates with faster training times, making it a promising alternative to previous methods. The findings suggest that 3D CNNs may offer a feasible alternative to recurrent neural models for SSI systems, which could have implications for improving speech synthesis technology in the future.

Utilizing adversarial training to improve Deep Neural Network models

In Chapter 3, GANs were employed to enhance the performance of ultrasound SSI models by incorporating perceptual loss in addition to conventional loss for two different datasets. The goal was to generate high-quality ultrasound images with improved accuracy and fidelity, which is crucial for SSI project. The proposed method involves adding a perceptual loss term to the conventional loss function used in the SSI model. The perceptual loss term is calculated based on the difference between the features extracted from the real and synthetic images by a pre-trained neural network. This approach enables the GAN to generate images that not only match the target distribution but also capture the relevant features and structures of the real images. To evaluate the performance of the proposed method, the SSI model was trained on two different datasets: one containing images of Hungarian corpus, and the other containing images of English corpus. The results showed that incorporating perceptual loss led to a significant improvement in the quality and accuracy of the generated images. The proposed method has the potential to improve the performance of ultrasound SSI models, making them more reliable and accurate for SSI applications. Moreover, the use of GANs and perceptual loss can be extended to other modalities.

Neural Speaker Embeddings for Generalizing Ultrasound SSI model

Chapter 4, delved into the exploration of Embedding Neural Networks to enhance the SSI model. The SSI model had previously not been performing optimally due to poorly tuned parameters. To address this issue, a new approach called x-vector was implemented and evaluated on unseen speakers to determine its effectiveness in improving the model's performance by incorporating speaker information into the input ultrasound data. The proposed strategy was tested on an English corpus dataset prepared at both the frame and speaker levels, including their respective spectrograms. The results indicated that this approach significantly improved the model's generalizability.

Convolutional Neural Networks for Detecting Voice Activity in Silent Speech Interfaces based on Ultrasound

In Chapter 5, we demonstrated that ultrasound images can be used to differentiate between silent and speech segments, similar to voice activity detection in speech. The training labels were estimated based on a public VAD implementation applied to the parallel speech recording. The classifier achieved an accuracy of 86% in discriminating silence and speech frames. We also highlighted the impact of retaining too much silence in the training set, which can affect the quality of the generated speech and the training of the model.

To address the challenge of preprocessing in Silent Speech Interfaces (SSI), we proposed a new method of voice activity detection. We implemented VAD on each speech frame instead of the entire speech, which resulted in better alignment between spectrogram and ultrasound frames and more reliable features for the model. we used VAD technique as a method of silence removal as an initial step before feature extraction. Then synchronized the windowed speech signal with the ultrasound frames and fed them to the VAD. Finally We performed the subsequent feature extraction steps for synthesizing speech or other related tasks using only the frames retained by VAD.

Overall, we showed that implementing VAD on each speech frame rather than the entire speech improved the accuracy of the SSI model. The proposed VAD method was effective in removing silence from the input data and generating more reliable features for the model. The results suggest that ultrasound images can be a useful tool for discriminating between silent and speech segments, which could have important implications for the development of SSI technology.

Enhanced analysis of ultrasound tongue videos via the fusion of ConvLSTM and 3D Convolutional Networks

In this Chapter, we present a structured approach to address the challenge of high computational cost in deep learning models, which require a large amount of data to achieve good performance. Our proposed solution involves a combination of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), which we call ConvLSTM, along with a 3D Convolutional Network (Conv3D). This combination allows us to extract sequential and volumetric information from the data and decrease the number of layers and parameters needed to train the model, while still achieving high performance.

To evaluate our proposed method, we conducted experiments on a Hungarian dataset and compared our results against previous high-performance models. Our chapter explains in detail the implementation of the ConvLSTM model and demonstrates its ability to extract spatial and temporal features from ultrasound tongue videos. The results of our experiments show that our proposed method outperforms the previous state-of-the-art models, achieving better accuracy and efficiency.

In conclusion, our method presents a promising approach to improving the performance of deep learning models while reducing the computational cost. We believe that our approach could be valuable for future research in the field of image processing and deep learning.

Enhancing Tongue Ultrasound-Based Silent Speech Interfaces with Spatial Transformer Networks

In the Chapter 7 we explored the use of spatial transformer networks (STNs) to enhance the speaker and session adaptation of ultrasound tongue imaging-based silent speech interfaces (SSIs). SSIs leverage deep learning algorithms to synthesize intelligible speech from articulatory movement data. However, existing models are often speaker-specific and perform poorly when switching between users or across different sessions.

To address this limitation, we propose extending deep networks with a spatial transformer network module. The STN module enables affine transformations on the input images, facilitating quick adaptation for different speakers and sessions.

By integrating the spatial transformer networks, the adapted SSI models can improve performance when faced with variations in tongue articulation across speakers or changes in the recording setup.

Contributions of the thesis

In the **first thesis group**, the contributions are related to the publication '3D Convolutional Neural Networks for Developing Silent Speech Interfaces Utilizing Ultrasound'. Detailed discussion can be found in Chapter 2.

I/1. Implementing the neural networks used in the experiments to restore speech signals from articulatory recordings. Specifically, we implemented a 3D convolutional neural network with different window length and compared it with different variations of CNN and combination with CNN+LSTM and BiLSTM networks.

I/2. Calculating the performance of the models using objective metrics such as STOI, PESQ, and MCD. The results obtained from these metrics were analyzed to compare the performance of the different network architectures.

In the **second thesis group**, the contributions are related to the publication 'Utilizing adversarial training to improve Deep Neural Network models'. Detailed discussion can be found in Chapter 3.

II/1. Implementation of GAN and CGAN models for image generation.

II/2. Utilization of various SSI models as generators in the GAN and CGAN frameworks.

II/3. Calculation and analysis of performance metrics to evaluate the effectiveness of the models.

II/4. Conducting research in the field of GANs and image generation.

In the **third thesis group**, the contributions are related to the publication 'Neural Speaker Embeddings for Generalizing Ultrasound SSI model'. Detailed discussion can be found in Chapter 4.

III/1. Preparing data for the experiments.

III/2. Implementing the model and conducting the experiments.

III/3. Comparing the results obtained from the experiments.

III/4. Calculating the relevant metrics to evaluate the model's performance.

In the **forth thesis group**, the contributions are related to the publication 'Convolutional Neural Networks for Detecting Voice Activity in Silent Speech Interfaces based on Ultrasound'. Detailed discussion can be found in Chapter 5.

IV/1. Implementing the model for voice activity detection using a CNN architecture and training it with binary cross-entropy loss function.

IV/2. Developing the idea of applying VAD to remove silence from the speech signal in SSI systems.

IV/3. Analyzing the results of experiments with different amounts of silence in the corpus, comparing the MCD and MSE metrics, and evaluating the impact of VAD on the SSI.

In the **fifth thesis group**, the contributions are related to the publication 'Enhanced analysis of ultrasound tongue videos via the fusion of ConvLSTM and 3D Convolutional Networks'. Detailed discussion can be found in Chapter 6.

V/1. Preparing data for the specific task.

V/2. Implementing code for the models (Conv3D, Conv3D+BiLSTM, ConvLSTM).

V/3. Analyzing and interpreting the results.

V/4. Calculating the evaluation metric for the results.

In the **six thesis group**, the contributions are related to the publication 'Enhancing Tongue Ultrasound-Based Silent Speech Interfaces with Spatial Transformer Networks'. Detailed discussion can be found in Chapter 7.

VI/1. Preparing data .

VI/2. Implementing code for the models.

VI/3. Analyzing and interpreting the results.

Összefoglalás

Ez a dolgozat új stratégiákat mutatott be az ultrahang SSI projekt különböző aspektusainak javítására, beleértve a modell megvalósítását és továbbfejlesztését, az adatok előkészítését és feldolgozását, az általánosítást és a modell betanítási gyorsítását. A javasolt módszereket két nagy adathalmazon tesztelték, és az eredményeket értékelték. A 1 fejezet röviden bemutatta az SSI rendszer alapvető összetevőit, mint például a funkciókivonó, a különböző módozatok és a modellképzés. A fejezet bemutatta a mély neurális hálózatok, különösen a CNN-ek működését és hasznosságát a képfeldolgozásban. A következő fejezetekben az SSI rendszer különböző aspektusait vizsgálták meg, és a kapcsolódó területekről származó algoritmusokat adaptálták az SSI rendszer teljesítményének javítására.

3D konvolúciós neurális hálózatok ultrahang alapú csendes beszéd felületekhez

A 2 fejezetben mély neurális hálózatokat használtunk a nyelvmozgások ultrahangos videójának beszéddé alakítására. Konvolúciós neurális hálózatokat (CNN) használtunk a képsorok feldolgozásához, ami népszerű a képfelismerésben. A bemenet egy videosorozat volt, amely a nyelvmozgások időpálya-információit tartalmazza. Ezután különböző hálózati struktúrákat vizsgáltunk az időszekvenciák feldolgozásához, beleértve a 2D CNN és egy ismétlődő neurális hálózat egymásra helyezését, valamint a 2D CNN kiterjesztését 3D-re az idő extra dimenzióként való hozzáadásával. Azt találtuk, hogy a 3D CNN-modell alacsonyabb hibaarányt ért el, kisebb volt, és gyorsabb volt a betanítása, mint a CNN+LSTM modell, ami arra utal, hogy a 3D CNN-ek megvalósítható alternatívát jelentenek az ultrahangos videó alapú beszédszintézis interfész (SSI) ismétlődő neurális modelljeivel szemben. rendszerek. Ez a fejezet jelentős hozzájárulást jelent a beszédszintézis területéhez, bemutatva a mély neurális hálózatok alkalmazásának lehetőségét az ultrahang videó beszéddé alakítására. A CNN-ek használata az időszekvenciák feldolgozására új megközelítést kínál, amely alacsonyabb hibaarányt érhet el gyorsabb betanítási idővel, így ígéretes alternatívát jelent a korábbi módszerekhez képest. Az eredmények azt sugallják, hogy a 3D CNN-ek megvalósítható alternatívát kínálhatnak az SSI-rendszerek visszatérő neurális modelljeivel szemben, amelyek hatással lehetnek a beszédszintézis technológia fejlesztésére a jövőben.

A kontradiktórius képzés felhasználása a Deep Neural Network modellek fejlesztésére

A 3. fejezetben a GAN-okat az ultrahangos SSI-modellek teljesítményének fokozására használták azáltal, hogy a hagyományos veszteség mellett az észlelési veszteséget is beépítették két különböző adatkészlet esetében. A cél az volt, hogy kiváló minőségű ultrahangképeket hozzanak létre, javított pontossággal és pontossággal, ami kulcsfontosságú az SSI projekt számára. A javasolt módszer magában foglalja az észlelési veszteség kifejezés hozzáadását az SSI modellben használt hagyományos veszteségfüggvényhez. Az észlelési veszteséget a valós és a szintetikus képekből egy előre betanított neurális hálózat által kinyert jellemzők különbsége alapján számítjuk ki. Ez a megközelítés lehetővé teszi a GAN számára, hogy olyan képeket állítson elő, amelyek nemcsak a céleloszlásnak felelnek meg, hanem a valós képek releváns jellemzőit és struktúráit is rögzítik. A javasolt módszer teljesítményének értékeléséhez az SSI-modellt két különböző adathalmazra betanították: az egyik a magyar korpusz képeit, a másik pedig az angol korpusz képeit tartalmazza. Az eredmények azt mutatták, hogy az észlelési veszteség beépítése jelentősen javította a generált képek minőségét és pontosságát. A javasolt módszer javíthatja az ultrahangos SSI-modellek teljesítményét, megbízhatóbbá és pontosabbá téve azokat az SSI alkalmazásokhoz. Sőt, a GAN-ok és az észlelési veszteség alkalmazása más módokra is kiterjeszthető.

Neurális hangszóró beágyazások az ultrahang SSI modell általánosításához

Fejezet 4, az SSI-modell fejlesztése érdekében végzett neurális hálózatok beágyazásának feltárásával foglalkozik. Az SSI modell korábban nem működött optimálisan a rosszul hangolt paraméterek miatt. A probléma megoldása érdekében egy új megközelítést, az x-vektort vezették be és értékelték ki a nem látott hangszórókon, hogy meghatározzák annak hatékonyságát a modell teljesítményének javításában azáltal, hogy a hangszóró információit beépítették a bemeneti ultrahangadatokba. A javasolt stratégiát egy angol korpusz adatkészleten tesztelték, amelyet mind a keret, mind a hangszóró szintjén készítettek, beleértve a megfelelő spektrogramokat. Az eredmények azt mutatták, hogy ez a megközelítés jelentősen javította a modell általánosíthatóságát.

Konvolúciós neurális hálózatok a hangtevékenység észlelésére a csendes beszédfelületeken ultrahangon alapuló

A 5 fejezetben bemutattuk, hogy az ultrahangképek segítségével megkülönböztethető a néma és a beszédszegmens, hasonlóan a beszéd hangaktivitás-érzékeléséhez. A képzési címkéket a párhuzamos beszédrögzítésre alkalmazott nyilvános VAD implementáció alapján becsültük meg. Az osztályozó 86%-os pontosságot ért el a csend és a beszédkeretek megkülönböztetésében. Kiemeltük továbbá a túl sok csend megtartásának hatását a tanítókészletben, ami befolyásolhatja a generált beszéd minőségét és a modell képzését.

A Silent Speech Interfaces (SSI) előfeldolgozással kapcsolatos kihívások megoldására új módszert javasoltunk a hangtevékenység észlelésére. A VAD-ot minden beszédkeretre implementáltuk a teljes beszéd helyett, ami jobb összehangolást eredményezett a spektrogram és az ultrahang keretek között, és megbízhatóbb funkciókat eredményezett a modell számára. a VAD technikát alkalmaztuk a csend eltávolításának módszereként, a jellemzők kivonása előtti kezdeti lépésként. Ezután szinkronizálta az ablakos beszédjelet az ultrahang keretekkel, és betáplálta őket a VAD-ba. Végül elvégeztük az ezt követő jellemző kivonási lépéseket a beszéd szintetizálásához vagy más kapcsolódó feladatokhoz, csak a VAD által megtartott keretek felhasználásával.

Összességében megmutattuk, hogy a VAD implementálása az egyes beszédkeretekre, nem pedig a teljes beszédre, javította az SSI modell pontosságát. A javasolt VAD módszer hatékonyan eltávolította a csendet a bemeneti adatokból, és megbízhatóbb jellemzőket generált a modell számára. Az eredmények arra utalnak, hogy az ultrahangos képek hasznos eszközei lehetnek a néma és a beszédszegmensek megkülönböztetésének, aminek fontos következményei lehetnek az SSI technológia fejlődésére.

Az ultrahangos nyelvvideók továbbfejlesztett elemzése a ConvLSTM és a 3D Convolutional Networks fúziójával

Ebben a fejezetben egy strukturált megközelítést mutatunk be a mély tanulási modellek magas számítási költségéből adódó kihívások kezelésére, amelyek nagy mennyiségű adatot igényelnek a jó teljesítmény eléréséhez. Javasolt megoldásunk a konvolúciós neurális hálózatok (CNN) és a visszatérő neurális hálózatok (RNN) kombinációját foglalja magában, amelyeket ConvLSTM-nek nevezünk, valamint egy 3D konvolúciós hálózatot (Conv3D). Ez a kombináció lehetővé teszi számunkra, hogy szekvenciális és volumetrikus információkat nyerjünk ki az adatokból, és csökkentsük a modell betanításához szükséges rétegek és paraméterek számát, miközben továbbra is nagy teljesítményt érünk el.

Javasolt módszerünk értékeléséhez magyar adathalmazon végeztünk kísérleteket, és összehasonlítottuk eredményeinket korábbi nagy teljesítményű modellekkel. Fejezetünk részletesen elmagyarázza a ConvLSTM modell megvalósítását, és bemutatja, hogy képes az ultrahangos nyelvvideókból térbeli és időbeli jellemzőket kinyerni. Kísérleteink eredményei azt mutatják, hogy a javasolt módszerünk felülmúlja a korábbi csúcsmodelleket, jobb pontosságot és hatékonyságot ér el.

Összefoglalva, módszerünk ígéretes megközelítést kínál a mély tanulási modellek teljesítményének javítására, miközben csökkenti a számítási költségeket. Meggyőződésünk, hogy megközelítésünk értékes lehet a jövőbeli kutatások számára a képfeldolgozás és a mély tanulás területén.

A nyelv ultrahang alapú csendes beszéd interfészek javítása tértranszformátor hálózatokkal

A 7 fejezetben megvizsgáltuk a térbeli transzformátorhálózatok (STN-ek) használatát az ultrahang-nyelvkép-alapú néma beszéd interfészek (SSI-k) hangszóró- és munkamenet-adaptációjának javítására. Az SSI-k mély tanulási algoritmusokat alkalmaznak, hogy az artikulációs mozgásadatokból érthető beszédet szintetizáljanak. A meglévő modellek azonban gyakran hangszóró-specifikusak, és gyengén teljesítenek a felhasználók közötti váltáskor vagy a különböző munkamenetek között.

Ennek a korlátozásnak a megoldására javasoljuk a mély hálózatok térbeli transzformátoros hálózati modullal történő kiterjesztését. Az STN modul affin transzformációkat tesz lehetővé a bemeneti képeken, megkönnyítve a különböző hangszórókhoz és munkamenetekhez való gyors alkalmazkodást.

A térbeli transzformátor-hálózatok integrálásával az adaptált SSI-modellek javíthatják a teljesítményt, ha a hangszórók nyelvi artikulációjának eltéréseivel vagy a felvételi beállítások változásaival szembesülnek.

A dolgozat hozzájárulásai

A **első téziscsoport** a hozzászólások a 3D konvolúciós neurális hálózatok ultrahangot használó csendes beszédinterfészek fejlesztéséhez” című kiadványhoz kapcsolódnak. A részletes vita a 2 fejezetben található.

I/1. A kísérletek során alkalmazott neurális hálózatok implementálása beszédjelek helyreállítására artikulációs felvételekből. Konkrétan egy 3D konvolúciós neur ális hálózatot implementáltak különböző ablakhosszokkal, majd összehasonlították különböző változataival, valamint a CNN+LSTM és a BiLSTM hálózatok kombinációjával.

I/2. A modellek teljesítményének kiszámítása objektív metrikák, például az STOI, a PESQ és az MCD használatával történt. Az ezekből a metrikákból nyert eredményeket elemezték annak érdekében, hogy összehasonlítsák a különböző hálózatarchitektúrák teljesítményét.

A **második téziscsoport** a hozzászólások az Adverzális tréning felhasználása a mély ideghálózati modellek javítására” című kiadványhoz kapcsolódnak. A részletes vita a(z) 3 fejezetben található.

II/1. GAN és CGAN modellek megvalósítása képgeneráláshoz.

II/2. Különböző SSI modellek felhasználása generátorként a GAN és CGAN keretrendszerben.

II/3. Teljesítménymutatók kiszámítása és elemzése a modellek hatékonyságának értékelése érdekében.

II/4. Kutatás végzése a GAN-ok és képgenerálás területén.

A **third thesis group** a hozzászólások a Neurális hangszóró beágyazások az általánosító ultrahang SSI modellhez” című kiadványhoz kapcsolódnak. A részletes vita a 4 fejezetben található.

III/1. Adatok előkészítése a kísérletekhez.

III/2. A modell megvalósítása és a kísérletek végrehajtása.

III/3. Az kísérletekből nyert eredmények összehasonlítása.

III/4. A releváns metrikák kiszámítása a modell teljesítményének értékeléséhez.

A **forth thesis group** hozzászólásai a Konvolúciós neurális hálózatok a hangtevékenység kimutatására ultrahangon alapuló néma beszédfelületeken” című kiadványhoz kapcsolódnak. A részletes vita a 5 fejezetben található.

IV/1. A hangaktivitás-detekció modelljének megvalósítása CNN architektúrával és bináris keresztentrópia veszteségfüggvénnyel való betanítása.

IV/2. Az ötlet kidolgozása a VAD alkalmazásáról a csend eltávolítása érdekében besz édjelekből a SSI rendszerekben.

IV/3. Az eredmények elemzése különböző csendmennyiséggel rendelkező korpuszokon végzett kísérletek után, az MCD és MSE metrikák összehasonlítása, vala mint a VAD hatásának értékelése a SSI-re. .

A **ötödik téziscsoport** hozzászólásai az Ultrahangos nyelvvideók továbbfejlesztett elemzése a ConvLSTM és a 3D Convolutional Networks fúziójával" című kiadványhoz kapcsolódnak. A részletes vita a 6 fejezetben található.

V/1. Adatok előkészítése a konkrét feladathoz.

V/2. A modellek (Conv3D, Conv3D+BiLSTM, ConvLSTM) kódjának megvalósítása.

V/3. z eredmények elemzése és értelmezése.

V/4. Az értékelési metrika kiszámítása az eredményekhez.

A **hat téziscsoport** hozzászólásai az A nyelv ultrahangon alapuló csendes beszédinterfészek fejlesztése térbeli transzformátor hálózatokkal" című kiadványhoz kapcsolódnak. Részletes beszélgetés a 7. fejezetben található.

VI/1. Az adatok előkészítése.

VI/2. A modellek kódjának implementálása.

VI/3. Az eredmények elemzése és értelmezése.

Publications

Journal publications

[1] T. G. Csapó, G. Gosztolya, L. Tóth, **A. Honarmandi Shandiz** , A. Markó. Optimizing the Ultrasound Tongue Image Representation for Residual Network-based Articulatory-to-Acoustic Mapping. *Sensors*, 22(22), 8601, 2022.

Full papers in conference proceedings

[2] L. Toth, **A. Honarmandi Shandiz**. 3D convolutional neural networks for ultrasound-based silent speech interfaces. In *International Conference on Artificial Intelligence and Soft Computing*, 159-169, Springer, 2020.

[3] **A. Honarmandi Shandiz**, T. G. Csapó, G. Gosztolya, L. Tóth, A. Markó. Improving neural silent speech interface models by adversarial training. In *Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV)*, 430-440, Springer, 2021.

[4] **A. Honarmandi Shandiz**, L. Tóth, G. Gosztolya, A. Markó, T. G. Csapó. Neural Speaker Embeddings for Ultrasound-Based Silent Speech Interfaces. In *Proceedings of the International Conference on Interspeech*, 1932-1936, Springer, 2021.

[5] **A. Honarmandi Shandiz**, L. Tóth. Voice activity detection for ultrasound-based silent speech interfaces using convolutional neural networks. In *Text, Speech, and Dialogue: 24th International Conference*, 499-510, Springer, 2021.

[6] **A. Honarmandi Shandiz**, L. Tóth. Improved Processing of Ultrasound Tongue Videos by Combining ConvLSTM and 3D Convolutional Networks. In *Advances and Trends in Artificial Intelligence. Theory and Practices in Artificial Intelligence: 35th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, 265–274, Springer, 2022.

[7] Y. Yide, **A. Honarmandi Shandiz**, L. Tóth. Reconstructing speech from real-time articulatory MRI using neural vocoders. In *2021 29th European Signal Processing Conference (EUSIPCO)*, 245–249, IEEE, 2021.

[8] C. Zainkó, L. Tóth, **A. Honarmandi Shandiz**, G. Gosztolya, A. Markó, G. Németh, T. G. Csapó. Adaptation of Tacotron2-based Text-To-Speech for Articulatory-to-Acoustic Mapping using Ultrasound Tongue Imaging. In *11th ISCA Speech Synthesis Workshop (SSW 11)*, 54-59, Springer, 2021.

[9] L. Tóth, **A. Honarmandi Shandiz**, G. Gosztolya, T. G. Csapó. Adaptation of Tongue Ultrasound-Based Silent Speech Interfaces
Using Spatial Transformer Networks. In *Proceedings of the International Conference on Interspeech*, Springer, 2023.

Further related publications

[10] T. G. Csapó, L. Tóth, **A. Honarmandi Shandiz**,G. Gosztolya, A. Markó. 3D konvolúciós neuronhálón és neurális vokóderen alapuló némabeszéd-interfész. In *MSZNY*, 2021.

Acknowledgments

I would like to take this opportunity to express my heartfelt gratitude to everyone who has contributed to my success in completing my PhD dissertation. It has been a long and challenging journey, but I could not have done it without the support and guidance of so many wonderful people.

Firstly, I would like to extend my sincere appreciation to my professor, Dr. Tóth László, for their exceptional guidance and unwavering support throughout my PhD journey. Their insightful feedback, constructive criticism, availability, knowledge and invaluable mentorship have been instrumental in shaping my research and helping me to achieve my goals.

I would also like to express my gratitude to Dr. Tamas Gabor CSAPO, for their continuous encouragement and support. Their expertise in the field, willingness to share their knowledge, and genuine interest in my work have been incredibly motivating and inspiring. Their contributions have been critical to my progress and have helped me to stay motivated and focused.

I would also like to thank Dr. Gosztolya Gábor for their support, valuable advice, and encouragement.

I am also grateful to the other professors who have been involved in my thesis, from BME, ELTE, and SZTE universities To finish this Amazing journey.

Last but not least, I would like to thank my family and my friends for their constant support, encouragement, and understanding. Their unwavering belief in me and my abilities has been a source of strength and inspiration throughout this journey.

Once again, thank you to everyone who has been a part of my PhD journey. Your support and guidance have been invaluable, and I could not have achieved this without you.

With enthusiastic steps, I moved forward towards progress
Sometimes I was happy, other times I had sorrow in my heart
But I was always following my dream
And with the hope that one day that dream would become a reality
Now, I have reached the end of a stage of my life
But I look at it with pride and joy

Because I have achieved many accomplishments along the way
And I am proud of my unwavering efforts to get here
With enthusiasm, I continue to move forward towards my progress
And with hope for even greater achievements in the future, along with happiness and prosperity.

As one stage comes to an end, another stage begins.

Amin Honarmandi Shandiz, August 2023