university of groningen

campus fryslân

Using voice conversion and time-stretching to enhance the quality of dysarthric speech for automatic speech recognition

Marjolein Spijkerman, S3219305, m.spijkerman.1@student.rug.nl,
External supervisor: B. M. Halpern,
Internal supervisor: V. Verkhodanova

July 17, 2022

Abstract

Dysarthria is a motor-speech disorder that is caused by damage to the central or peripheral nervous system. It affects the muscular control over the speech mechanism. This can cause communication issues. To help people to engage in conversation and to use assistive devices a voice conversion system is used to enhance the quality of the dysarthric speech. Non-parallel voice conversion methods are applied in the form of a MaskCycleGAN-based model, this is combined with time-stretching methods to account for the slowness of spastic dysarthric speech. The results indicate that applying the voice conversion and the time-stretching does improve the performance, however the actual performance is still quite low.
Keywords: Automatic Speech Recognition, Speech dysarthria, Speech Enhancement, Generative Adversarial Networks

Contents

1 Introduction

1.1 Dysarthria

Dysarthria is a motor-speech disorder that affects a person's muscular control over the speech mechanisms, which influences that person's ability to speak[43]. Unlike other speech disorders, dysarthria does not affect the ability to understand language. See section 2.1 for more details on dysarthria. Dysarthria can have a huge influence on people's lives, as being able to communicate is a necessity for a lot of tasks. As dysarthria does affect muscles in general and not only the muscles related to producing speech, other methods for communicating such as writing, typing on a keyboard, or using sign language are not always a viable alternative.

1.2 Assistive Devices

A possible option for helping people with dysarthria with communication is by using assistive devices, this has already been thoroughly researched in the past. For example, according to a pilot study by DeRosier and Farber in 2005 on user satisfaction and psychosocial impact of speech recognition software as an assistive device for participants with a non-speech related physical disability, participants were satisfied with the assistive technology offered[11]. Young and Mihailidis in 2010 on the other hand looked at dysarthric speakers specifically. They concluded that ASR systems did not work well enough yet and that for improvement custom-designed ASR systems needed to be created for each user[34]. Yilmaz et al. in 2019 looked into the articulatory and bottleneck features for speaker-independent ASR of dysarthric speech. They mitigated some of the problems of dysarthric speech, which improved the performance on the ASR systems but did not yet reach the level of healthy speech[48]. De Russis and Corno researched in 2019 the performance of three cloud based ASR systems, namely IBM Watson, Google Cloud and Microsoft Azure on dysarthric speech[10]. They concluded that even people with mild speech disorders will not be able to use these services at a similar level as healthy speakers. See section 2.1.2 for more details.

Thus, publicly available systems that use ASR, still don't perform as well on dysarthric speech as on healthy speech. Models that seem to specifically focus on dysarthric speech can get reasonable results, but on the other hand systems that are available to the public are not yet up to the task of understanding dysarthric speech.

1.3 Limitations of ASR systems

So, ASR systems can be used as assistive devices for people with dysarthria. Many different techniques can be used to create an ASR system. For example there are traditional ASR systems that use an acoustic model, a lexical model and a language model together to recognize speech. On the other hand there are also more modern approaches that use deep learning techniques, such as ASR systems based on recurrent neural networks, LSTMs or attention based models. An explanation on traditional ASR systems is given in section 2.2.1. An overview of examples of these different types of deep learning-based ASR systems in given in section 2.2.2.

ASR systems also still have limitations that they have to deal with. Some of the problems that they have to deal with stem from the environment, such as background noise or low quality microphones. Other problems stem from the persons who speaks themselves, such as variations in language usage, dialects or the pitch of a voice. See section 2.2.1.1 for more details. These speaker dependent problems can already be a problem in the case of healthy speakers, however in the case of dysarthric speakers this is even harder.

In general ASR systems can be considered pretty robust for healthy speakers who speak with a standard accent. However, next to the problems that ASR systems have to deal with, biases in ASR systems can also cause problems. Biases occur in ASR systems due to the composition of the training data and human biases that are in some of the transcriptions of corpora. The training data is generally mostly based on native speakers who speak with a standard accent, which causes mismatches with most speakers who don't conform to this speaking style. Biases in speech corpora can be, for example, caused when the people who transcribe the speech correct mistakes in speech. For example, when someone mispronounces a word or uses the wrong word in a context, the transcriber may correct this mistake in the transcription. This can negatively affect the performance of the ASR[14]. Feng et al. did research on these biases in Dutch and found that things like gender, age, and speaking style indeed influence the performance of the ASR system[14]. See section 2.2.3 for more details on this topic.

So, to conclude biases based on gender, age, dialect and speaking style do occur in ASR systems. So, overall there are still quite a lot of biases that may influence the performance of an ASR system. It is still not a one size fits all system. While ASR system can still have trouble with even regular healthy speech, it generally performs considerably better on healthy speech than on dysarthric speech.

1.4 Generative Adversarial Networks and Voice Conversion

Instead of only focusing on the more traditional approaches to speech recognition, there are also other types of networks that can help improve ASR systems. A generative network that has become very popular in the recent years is the generative adversarial network (GAN). A GAN is a model that focuses on generating new data. It does this by training two different models at the same time, a generator and a discriminator. These two models work as each other's adversary. The generator tries to create better content to try and fool the discriminator, while the discriminator tries to distinguish between the real and the fake data[23][15]. An in depth description of generative adversarial networks is given in sections 2.3.1 and 2.3.2.

Improvements to this model have been made in the form of a cycle-consistent version of a GAN, which uses two different generators instead of one. This allows the model to be able to translate back and forth between the two domains, which makes the conversion process more stable. These CycleGANs can work for unpaired and non-parallel data[49]. CycleGANs are described in more depth in section 2.3.2.2.

Most of the early usages of GANs focused on image generation[23][49], however it is also possible to use GANs for speech recognition. For example Dumpala et al. used CycleGANs to convert speech that contains perturbations to speech without perturbations to increase the performance on ASR systems with reasonable success[13]. Chen et al. used GANs to synthesize training data that has more acoustic diversity than synthesizers normally have and was able to successfully train an ASR system on this generated data[6]. These example applications of generative adversarial networks for speech recognition purposes are discussed in section 2.3.2.3.

Thus, GANs have been used successfully in the past for speech recognition purposes by doing things like converting speech data or generating more speech data. Due to the nature of most dysarthric speech datasets, there is very little parallel data, see section 2.4.2. This means that a non-parallel approach is necessary for using voice conversion on dysarthric speech.

Many different attempts at voice conversion have been done on healthy speech data. A quick overview of some of the attempts at using voice conversion before the discovery of GANs is given in section 2.4.3.1, these were all done on parallel datasets and would therefore not work for dysarthric speech. There have been reasonably successful attempts at non-parallel voice conversion using variational autoencoders, these are discussed in section 2.4.3.2. GAN-based approaches are discussed in section 2.4.3.3.

1.4.1 Voice Conversion and GANs used for Dysarthric Speech

Most of the attempts at using GANs for non-parallel voice conversion had been done on healthy speech. As ASR systems perform better on healthy speech, being able to convert dysarthric speech into healthy sounding speech could improve the performance on ASR systems. Work on non-parallel voice conversion of dysarthric speech into more healthy sounding speech has also been done before.

The topic of voice conversion is discussed in more depth in section 2.4. The ethical drawbacks of voice conversion are discussed in section 2.4.1. Some example approaches in voice conversion are discussed in section 2.4.3.

Besides voice conversion, other approaches can also be explored to improve the performance of ASR systems on dysarthric speech. For example Vachhani et al. worked on tempo and speed based augmentations of healthy speech data in order to mimic dysarthric speech. This can then be used as training data for ASR systems that focus on dysarthric speech. They concluded that using this augmented data as training data slightly improve the performance of the ASR system on dysarthric data[46]. Rudzicz looked at several adjustments that can be made to dysarthric speech in order to improve intelligibility, this also included experiments regarding speech tempo. However, he concluded that modifications to the tempo did not result in significant improvements in speech naturalness. The test he performed however focused only on naturalness, and not on other aspects of speech[40].

MaskCycleGAN-VC In the paper by Halpern et al.[37], they researched voice conversion. Specifically, they researched which parts of CycleGAN-based non-parallel techniques are essential to improving dysarthric speech recognition and how using time stretching affects the performance of GAN-based methods for dysarthric-to-normal voice conversion. They concluded that the choice of vocoder and the application of dynamic time warping influences the performance of dysarthric speech recognition when using CycleGAN-based techniques. Applying time stretching on the other hand improves recognition to a comparable level or better level than state-of-the-art GANs. And applying MaskCycleGAN-based voice conversion on time-stretched speech gives better performance than just using time stretching. The authors conclude that the enhancements are highly dependent on the temporal aspects of the speech, since both time-stretching and dynamic time-warping affected effect performance. However, their experiments assumed that the healthy speaker's speech rate is optimal without any justification.

1.5 Research Question and Hypothesis

Since both voice conversion and time stretching has a positive effect on dysarthric speech for ASR purposes, this paper will focus on continuing research on this topic.

The research question that is discussed in this research is the following:

> "Does applying time stretching techniques to audio files produced by people with speech dysarthria before applying voice conversion on the audio files improve the performance on an ASR system?"

We expect that applying these techniques will improve the performance on the ASR system in comparison to just using voice conversion on the audio files. The experiment is performed on the speech data of two different dysarthric speakers. The expectation is that the ASR system will perform significantly better on the speaker that had a higher intelligibility score in comparison to the speaker with a lower intelligibility score.

2 Literature Review

2.1 Dysarthria

The term dysarthria includes several speech disorders that are caused by damage to the central or peripheral nervous system, it affects the muscular control over the speech mechanisms. As opposed to other speech disorders such as aphasia or apraxia, dysarthria is unrelated to problems with understanding language or the planning of specific movements. Dysarthria does effect muscle control due to things like paralysis or weakness of muscles. Dysarthria can affect several aspects of speech, such as loudness, pitch or speed rate, which can cause speech to sound slurred, incomprehensible or uneven. Several different types of dysarthria exist, each type has their own cause and affects speech in a different manner.

2.1.1 Different Types of Dysarthria

Darley et al. defined five different types of dysarthria, namely flaccid dysarthria in the bulbar palsy, spastic dysarthria in the pseudobulbar palsy, ataxic dysarthria in cerebellar disorders, hypokinetic dysarthria in parkinsonism and hyperkinetic dysarthria in dystonia and chorea. Next to these five categories a mixed dysarthria, that combined elements of flaccid and spastic dysarthria, was found in amyotrophic lateral sclerosis[9]. Below is a short overview of the main symptoms of the five different categories of dysarthria.

2.1.1.1 Flaccid Dysarthria

Patients who have flaccid dysarthria show signs of hyperreflexia, they have overactive/overresponsive reflexes, and muscle flaccidity, where muscles are less prone to move and tend to become "floppy". Speech problems include hypernasality, imprecise consonants, breathy voice and monopitch[9].

2.1.1.2 Spastic Dysarthria

Patients who have spastic dysarthria have speech problems which include things such as pronouncing consonants imprecisely, speaking with a monotonous and possibly low pitch and monotonous loudness, speaking with reduced stress or emphasis patterns, speaking with a harsh, strained or strangled voice, speaking at a slower rate, speaking with a high level of nasality and having trouble with producing longer phrases[9].

2.1.1.3 Ataxic Dysarthria

Patients who have ataxic dysarthria struggle with timing, force, range and direction of movements. The problems occur during the production of speech include pronouncing consonants imprecisely, adding too much stress on unstressed parts of words, irregularly decreasing the accuracy of articulation, pronouncing vowel sounds distorted and speaking with a harsh voice[9].

2.1.1.4 Hypokinetic Dysarthria

This includes the patients who suffer from parkinsonism, patients with parkinsonism have symptoms that include tremors while at rest, slow and limited movements, rigidity of muscles and postural instability. These are the motor symptoms that are also found in Parkinson's disease. The slow movements may sometimes be alternated with movements that are, while limited in range, very fast. The problems in speech that these patients are facing include speaking at a monotonous pitch and loudness, speaking with reduced stress, pronouncing consonants imprecisely, having inappropriate silences while speaking combined with short rushes of speech and speaking with a harsh or breathy voice[9].

2.1.1.5 Hyperkinetic Dysarthria

This includes both people who suffer from dystonia and chorea. Dystonia can be considered as slow hyperkinesia, it builds up slowly and has a prolonged distorted effect on muscle movements and then gradually subsides again. Speech problems in the case of dystonia include imprecise consonants, distorted vowels, harsh voice, irregulary articulatory breakdown, strained-strangled voice, monopitch and monoloudness. Chorea on the other hand is a quick hyperkinesia, its effects are irregular, random and rapid. The effects on speech include imprecise consonants, prolonged intervals, variable speed rate, monopitch, harsh voice, inappropriate silences, distorted vowels and excess loudness variations[9].

2.1.2 Assistive Devices: General Examples

DeRosier and Farber did a pilot study on user satisfaction and psychosocial impact of speech recognition software as an assistive device back in 2005. They found that in general the participants were satisfied with the assistive technology and that the assistive technology had a somewhat positive psychosocial impact on their lives. The assistive software allowed them more independence as it gave them an efficient and flexible way to access a computer. However, this research focused on participants with a physical disability. So while it gives a good idea on the usability of ASR systems in general, in the case of patients with dysarthria an ASR system is needed that can understand dysarthric speech[11].

Young and Mihailidis in 2010 looked at the difficulties in ASR of dysarthric speakers and the usage of assistive devices by elderly people, they found that at the moment of their research the ASR system did not work well enough for either people with speech dysarthria or elderly people. Challenges included lack of user training, fatigue, frustration, errors and the surroundings. They mention that custom-designed ASR systems for specific speakers could solve these problems partially. However, creating a custom-designed ASR system for a very large group of people would be a costly solution, as each system would need to be created and trained separately[34].

Yilmaz et al. in 2019 looked into the articulatory and bottleneck features for speaker-independent ASR of dysarthric speech. They used speaker-independent bottleneck and articulatory features on dysarthric speech combined with a neural network-based acoustic model. They found the bottleneck features in the bottleneck layer, a fixed dimensional non-linear transformation that gives a representation of the sub-word unit subspace given the acoustics. This allows them to mitigate the effect of variations due to poor articulation of dysarthric speech. The model that worked best was their time-frequency convolutional neural network. While there was a performance gap between the normal and dysarthric speech, the results of the ASR on dysarthric speech is quite good[48].

De Russis and Corno researched in 2019 the performance of three cloud based ASR systems, namely IBM Watson, Google Cloud and Microsoft Azure on dysarthric speech. They researched this to find out whether people with dysarthria would be able to use these assistive services. They found that all three ASR systems perform similarly. The healthy control group had a very low word error rate of around 4-6%, while the dysarthric speakers in general had a word error rate between 59-67%. In the cases of people with a low level of dysarthria severity the word error rate was between 15% and 23%, and the people with a very severe case of dysarthria had a word error rate between 78% and 89%. They concluded that even people with mild speech disorders will not be able to use these services at a similar level as healthy speakers[10].

2.2 Automatic Speech Recognition Systems

2.2.1 Traditional ASR systems

The architecture of traditional automatic speech recognition (ASR) systems tend to contain the following aspects, namely an acoustic model, a pronunciation dictionary / lexical model and a language model. First the speech goes through a feature extraction pipeline. The feature extraction will return the most important features of each audio file. These features will then be analyzed by the acoustic model, the language model, and the pronunciation model at the same time. Together they will pick the most likely sequence of words. The acoustic model tries to predict the most likely occurring phonemes. The pronunciation model contains lists of possible pronunciations of different words and thus returns the possible words that are being said based on phonemes or graphemes that are found by the acoustic model. Lastly, the language model looks at the probability that a specific word is actually used, thus it looks at the probability that a specific sequence of words can actually occur[3].

2.2.1.1 Difficulties of ASR Systems

ASR systems still tend to have trouble with several elements of spoken text. These include things like speech comprehension, noise, body language, channel variability, and speaker variability. In the case of speech comprehension, ASR systems tend to lack knowledge of what exactly is being spoken. Even when an ASR system is able to do things like predicting the next word in a sequence of words to help with things such as estimating what words are possible in a certain sentence, this is no guarantee that the system will predict the correct word. The people who use the ASR system may use words in contexts that are illogical, but the system still needs to be able to understand those words. In the case of noise, words spoken in a noisy environment, tend to be hard for an ASR system to understand, as it first needs to filter out all unnecessary noises. In the case of body language, humans do not only use speech for communication. Lots of information is also shared in the form of soundless movements, such as by moving your hands or shaking your head. In the case of channel variability, the audio quality that an ASR system receives depends partially on the quality of the microphone. So, in some cases it will be better than in others. Speaker variability includes several things, such as the speaking style or the gender of a person. In the same language people may use different dialects when speaking, these all sound slightly different and things like the gender of the speaker influences the pitch of the voice. An ASR system needs to take all these small differences into account when making predictions [3].

2.2.2 Deep Learning Approaches to ASR Systems

Instead of using an acoustic, lexical and language model, deep learning approaches use neural networks. Popular approaches within deep learning for ASR include for example end-to-end models that use recurrent neural networks (RNN), long short-term memory (LSTM) models, and attention based models.

Recurrent Neural Networks In other neural networks such as convolutional neural networks the data is independent of each other. For example, in image recognition the order of the images does not matter. However, there are many applications in which the ordering of the data is important, for example in cases that deal with language or in other cases that have some form time-based or sequential ordering. Recurrent neural networks specifically focus on looking at sequences of data. The idea behind RNNs is to use states to keep track of information, since these states are not visible they are also called hidden states. The state keeps track of what happened in earlier steps, the current status and the influence of new inputs on the current state. Inserting a new input causes the state to get updated and gives some output value. The order of these inputs matter, as this is what makes it sequential. Without this ordering it would be more similar to a normal connected layer in a neural network. The different inputs in the RNN are called time steps. However, not all RNNs deal with data that is necessarily related to time, as things like written text are not affected by time. However, the RNN does get the words one by one, making it sequential. They are called recurrent as the recurrent cells are used over and over again during the training process[16].

In deep learning, these states get implemented in the form of a recurrent cell. The recurrent cell is saved as a tensor and manages the states and represents the output. These recurrent cells are set in a recurrent layer. A recurrent neural network generally has multiple recurrent layers. An example usage of RNNs in language usage would be that the recurrent cell gets an word as input for every training step, it would then predict the next word in the sequence. If it incorrectly predicts the output, it can be corrected using a special version of the backpropagation algorithm that takes this recurrent information into account. However, since the backpropagation is constantly applied to the same recurrent cell, this can lead to either the vanishing gradient or the exploding gradient problem. In these case of the vanishing gradient problem, the gradient becomes very small very quickly. This causes the network to stop training, this also negatively affects the other layers. In the case of the exploding gradient problem the opposite happens and the gradients become too large. This causes the updates to the weights to also become too large, which can make the entire network unstable[16].

Long Short-Term Memory As a solution to the vanishing and exploding gradient problem, a slightly different RNN model can be used, the Long Short-Term Memory (LSTM). The LSTM works similarly to a normal RNN, but can actively choose what it wants to keep track of and what it wants to forget. It uses three internal networks, one to forget the information it no longer needs, one to insert new information and one that keeps track of the internal state and the output of the cell. This means that the LSTM does not require repeated copies, and can thus avoid the vanishing and exploding gradient problems. Most RNN models use LSTM in their application. A popular version of LSTM is the Gated Recurrent Unit (GRU)[16].

Attention Attention can be added as an additional mechanism to a LSTM RNN model. It makes it possible for a model to focus on specific elements of a sequence by focusing the resources of the network on the parts of the input that matter the most. For example in translation tasks, it allows the model to focus on the words that affect the meaning of the sentence the most, such as nouns[17].

2.2.2.1 Examples

Below a quick overview of few of the approaches that use RNN-based models for ASR is given:

In 2015 Chorowski et al. worked on attention based models for speech recognition. Attention based models already worked quite well for other tasks, such as machine translation. So, they tried to implement it for speech recognition. They implemented two new ideas which gave them good results, namely a different normalization approach for smoother alignments, and a new principle for the usage and extraction of features of previous alignments. They reached an phoneme error rate of 18.7% on their baseline phoneme recognition task, this version only worked for utterance of the same length as the training data. Their two experimental setups that focused on adding location-awareness to the attention mechanism and avoiding concentrating too long on the same frames, allowed them to use longer utterances and decreased the phoneme error rate to 18% and 17.6% respectively[7].

In 2015 Sak et al. worked on LSTM RNN models for speech recognition, specifically they worked on some techniques to further improve the performance of these types of models. They worked on using state level minimum Bayes Risk-trained connectionist temporal classification models for ASR, they showed that by adding context dependent phone models and using the appropriate features this model outperformed previous LSTM RNN models. [41].

In 2018 Rao et al. worked on an sequence-to-sequence ASR using an RNN-Transducer for real-time streaming speech recognition purposes. The architecture learns both acoustic and language model components from trascribed acoustic data. The model has an encoder, which uses a connectionist temporal classification-based acoustic model and a decoder that uses recurrent neural network language model that is trained on text data. They compared different model settings. The best result that they achieved had a word error rate of 8.5% on voice search and 5.2% on voice dictation tasks, using a RNN-T system with a twelve-layer LSTM encoder and a two-layer LSTM decoder[38].

In 2018 He et al. also worked on ASR model for streaming purposes, specifically an end-to-end model for mobile devices. They compared an connectionist temporal classification-based model with a RNN-T based model. They build their model using a recurrent neural network transducer. Their encoder network consisted of 8 layers of LSTM cells. They tested their models on both voice search and dictation recordings. They found out that their proposed method for the RNN-T can outperform their connectionist temporal classification-based model in terms of latency and accuracy in some of their experiments[24].

2.2.3 Biases in ASR Systems

ASR systems generally are a pretty robust system for healthy speakers who speak with a standard accent. It is more complicated for people who do not adhere to the rules of the acoustic and pronunciation model. Thus, while working most of the times for specific types of speech, ASR systems do still have quite a few pitfalls regarding variations in speech.

Feng et al. have researched biases that can occur in ASR systems. ASR systems should give objective transcriptions of human speech, however in reality this does not happen. The performance of ASR systems is affected by things like gender, age, speech impairments, race and accents. They mentioned that in some languages females are understood better, while in others male speakers are understood better. In general younger teen and adult speakers are understood better than older speakers, while at the same time child language is understood worse as children have a shorter vocal tract and sometimes speak at a variable rate with inaccurate articulation. Some ASR systems are also affected by race and make more mistakes in transcribing the speech of black speakers than white speakers[14].

These biases can be caused by the composition of the training data, as ASR systems are generally trained on native speech without any unusual dialects. This negatively affects people who speak with non standard accents or non native speakers of the language. This can cause mismatches on several levels. Differences in language usage can cause mismatches with the language model. Differences in articulation and sound realisation caused by speaking style, accent, speech rate, and vocal tract length can cause mismatches with the acoustic model. Besides speech based biases, biases in transcription can also occur. As transcriptions are created by humans, human-like biases end up in the language corpora. These include problems such as that people sometimes correct and normalize the errors made by children. These corrections cause the transcriptions to become inaccurate, which increases the number of out of vocabulary words and affects the performance of the ASR on child speech[14].

They researched these biases by performing experiments on a Dutch ASR system. They compared different groups, including different genders, different ages, regional accents and non-native accents. They used the word error rate to measure the performance. They tested both read speech, which is less affected by speaker style and tends to be articulated better, and human machine interaction speech, which is more natural and contains more variability. ASR systems tend to perform better on read speech. They found that in Dutch female speakers perform better than male speakers for all groups. The group with the largest difference between male and female speakers was the group of native older adult speakers and the group with the least difference in word error rate between male and female speakers was in the group of native children. In the group of native speakers, the group of teens had the best performance, followed by the older adults and in the last place were the children. The difference between the performance of different age groups in non native speech was minimal. Also, the effect of language fluency of the non-native speakers was minimal. There was a significant difference between the performance of native and non native speakers. The different regional dialects also influenced the performance, as some regions performed worse than others[14].

2.3 Generative Adversarial Networks

2.3.1 Introduction

Generative adversarial networks (GAN) are a framework for estimating generative models using a adversarial process. These GANs can be used for supervised, semi-supervised, unsupervised and reinforcement learning. A GAN generally consists of two models that are being trained at the same time: a generator and a discriminator. One model generates new content, while the other model tries distinguish between the original data and the generated data. These two models are each other's adversary, hence the name adversarial. The generator learns to generate better content to fool the discriminator, while the discriminator improves at seeing the difference between the real and fake content[23][15].

2.3.1.1 The Difference Between Generative and Discriminative Models

To understand the difference between the generator and the discriminator, the difference between generative and discriminative models needs to be explained. Discriminative models can see the difference between multiple instances and then for example indicate the most likely label for this instance. Generative models on the other hand focus on generating new data instances.

Generative models need more knowledge than discriminative models, as they need to model more complex things. So, instead of modelling the conditional probability of a label given an instance, it models the joint probability between the data and the label. This also allows the model to calculate the conditional probability, meaning it also can perform classification tasks. A generative model can also indicate how likely an instance in a data set is, this allows it to perform tasks such as predicting the next word in a sequence. A discriminative model on the other hand would not be able to perform this task. Thus, generative models learn by using all available data, allowing them to perform multiple tasks. On the other hand, discriminative models only use the data that is needed for the task they are trained on, and does not model other data that may be necessary for other tasks[1].

As an example of using a generative and a discriminative model for a classification task, Ng and Jordan performed several experiments in 2002 on a logistic regression model as a discriminative classifier, and a naive Bayes as a generative classifier. They concluded that while discriminative classifiers are generally preferable over generative classifiers, there are cases in which a combination of the two would give a better performance. In the case of classification problems, when using Bayes rules to calculate the most likely label, the classifier would learn a model of the joint probability of the input and the label. Then afterwards it would calculate the probability of the label given the input. The discriminative classifier, logistic regression, on the other hand calculates the probability of the label given the output directly. They both perform the same task, however generative models learn an extra step[36].

2.3.2 Overview of Implementations of GAN Models

2.3.2.1 The First GAN Model

The first Generative Adversarial Network (GAN) was proposed by Goodfellow et al. in 2014[23]. While generative models already existed beforehand, this was the first attempt at estimating generative models using a adversarial process. When they did their research on GANs, generative models were less popular than discriminative models in the field of deep learning. Deep learning at that time required models that could represent probability distributions for data such as images, audio and linguistic symbols and discriminative models were able to use this data as input and produce an output label, using backpropagation and dropout algorithms, which in turn use piecewise linear units, which have a well-behaved gradient. To explain these terms, deep learning uses neural networks as models, these neural networks contain multiple layers of nodes. The data goes in the input layers, then it goes through several hidden layers, and then the output layer gives the prediction. The hidden layers have so-called weights that influence the model's output, when training the model the weights get changed to best fit the data. Next to that an activation function is necessary to add non-linearity to the model, without this a model could only approximate linear relations. For example, the earlier mentioned piecewise linear unit is a type of activation function. This function is made up of several pieces of straight lines where the lines together do not form a single straight line. A popular activation function of this type is ReLU, rectified linear unit. This function returns 0 for every input of 0 or lower, and otherwise returns the input[18]. Backpropagation is the method that is used as part of the training process of neural networks. It calculates the gradient of the loss function with respects to the weights. It allows the information of the cost function flow backwards through the network, such that this gradient can be calculated. The actual learning of the model is later done by using another algorithm such as stochastic gradient descent, which uses this gradient that was calculated using backpropagation[19][22]. Dropout is a regularization technique. Regularization helps against overfitting. Dropout is added in the form of a dropout layer. The dropout layer sets a previously chosen percentage of units in the hidden layer to 0, each training round it chooses the units randomly. The units that are set to 0 are temporarily disabled and will not be taken into account when updating the weights of the model[20].

Generative models on the other hand struggle due to the difficulty of needing to approximate many probabilistic computations, such as in maximum likelihood estimation, without being able to calculate this in an efficient manner. Next to this it is difficult to use piecewise linear units in a generative context. Goodfellow et al. present their adversarial nets framework as a model that can avoid these problems. They use two models, a generative model and a discriminative model, and set these up as adversaries. One tries to produce fake data without being detected, while the other tries to learn whether the data is from the model or data distribution. The competition between these two models allows them to train until the fake data is not detectable from real data anymore.

The GAN model by Goodfellow et al. consists of two models, so called adversarial nets. Both models are trained using only backpropagation and dropout algorithms. The sampling from the generative model is done by using forward propagation. Unlike some older generative models, these models do not require Markov chains. The comparisons between the GAN model and other generative models that Goodfellow et al. made in their paper are summarized below in section 2.3.2.1. The two adversarial models, that are represented as multi-layer perceptrons, play a min-max game together to minimize the value function over the two models. The generator learns the distribution over the data using a prior on the input noise variable and represents a mapping of the dataspace as a differentiable function that is represented by a multi-layer perceptron. The other multi-layer perceptron which represents the discriminator outputs a scalar. This perceptron calculates the probability of some data belonging to the training data or to the generated data. The discriminator is trained to maximize the probability of assigning the correct label.

The min-max game is implemented by alternating between k steps of optimizing the discriminator and 1 step of optimizing the generator. This is done to avoid overfitting and to make sure that the discriminator performs as optimally as possible, while still slowly training the generator. This number of training steps k for training the discriminator is their hyperparamter, different values for k give different results. They describe the algorithm that they are following for this implementation as a minibatch stochastic gradient descent training algorithm. This algorithm consists of the following parts: for each training iteration, they do k rounds of updating the discriminator followed by 1 round of updating the generator. In each of the k steps, first a minibatch of noise samples from the prior get sampled, then a minibatch of examples from the data generating distribution gets sampled. These samples are then used to update the discriminator by calculating the stochastic gradient ascent. The part for the discriminator works similarly, first a minibatch of noise samples from the prior is sampled. Then this is used to update the generator using stochastic gradient descent. Thus, the discriminator uses both the noise prior and the data generating distribution to update it with stochastic gradient ascent, while the generator only uses the noise prior and updates itself using the stochastic gradient descent.

To evaluate their GAN they compared the performance with three other models, namely a stacked convolutional autoencoder, a deep belief network and a deep generative stochastic network. They trained on the MNIST data set and the Toronto Face Database. Both these data sets contain images, so the generator focused on the creation of similar images. The generator uses a combination of two different activation functions, namely rectifier linear activation and sigmoid activation. The discriminator uses maxout activation. They applied dropout when training the discriminator. They evaluated the performance by fitting a Gaussian Parzen window to the generated samples and calculating the log-likelihood over this. They mentioned that this evaluation method has a high variance and does not perform well in high-dimensional spaces, but that they did not have a better evaluation method at the moment that could be used. The adversarial networks performed better than the other networks in the case of the MNIST data set. It outperformed the deep belief network and deep generative stochastic network, and performed similarly to the Stacked convolutional autoencoder for the Toronto Face Database data set.

They concluded that the advantages of this system are mostly computational, as it removes the need for Markov chains. Furthermore, the generator is trained using the gradients instead of using the data examples directly. This way no components of the input data are copied into the parameters of the generator[23].

Other Generative Models
They also described other deep generative models that already existed and how their GAN differed from these. Below an overview of the most important generative models and the changes is given:

Boltzmann Machines: Boltzmann machines are a class of stochastic models named after the Boltzmann distribution. They term Boltzmann machine was first coined by Ackley, Hinton and Sejnowski in 1985[2]. Boltzmann machines are a connectionist approach to learning probability distributions. In time newer versions of the Boltzmann machines were created such as, Salakhutdinov and Hinton's deep Boltzmann machine in 2009[42]. Generally Boltzmann machines are trained by maximizing the log-likelihood. Calculating these require many approximations of the likelihood gradient, without having an efficient way of calculating these gradients. Besides this, Boltzmann machines also require the use of Markov chains[23].

Generative Stochastic Networks Bengio et al. worked on a generative stochastic network framework[4]. This framework learns the transition operator of a Markov chain. The transition distribution of the Markov chain depends on the previous state. Each move is generally quite small, resulting in the conditional distribution having few dominant nodes. Thus, it is easier for this model to learn, as this allows the model to learn using backpropagation. It works similarly to a deep Boltzmann machine, but uses backpropagation instead of layerwise pretraining. As Boltzmann machines and other similar models have trouble with the calculations of the likelihood gradient, "generative machine"-models, such as these generative stochastic networks can be used as an alternative. These do not represent the likelihood explicitly, but instead generate the samples from the desired distribution. They are trained using backpropagation instead of approximations. But, they still require the use of Markov chains. The GAN on the other hand, remove the need for Markov chains from the process[23].

Variational autoencoders Autoencoders are a type of learning architecture that focuses on encoding data. They are mostly used for dimensionality reduction and removing noise. Autoencoders use make use of latent variables, these latent variables are values that are inherent to the data, they are what is left of the data after transforming it into a lower dimensional space. Variational autoencoders are similar to normal autoencoders, as in they perform the same task in a similar manner. However, regular autoencoders are deterministic, they always create the same output given a specific input. Variational autoencoders are nondeterministic, they add a form of unpredictability to the output[21]. Research in this topic was done around the same time as the first research in GANs, by two different groups in 2014[33][39]. They work similarly to GANs as in they also combine a generative model with a second neural network, however they use a different technique to create their system. These systems use a recognition model that performs approximate inference. They require differentiation through the hidden units and cannot have discrete latent variables. GANs on the other hand differentiate the visible units and cannot have model discrete data[23].

2.3.2.2 CycleGAN

Zhu et al. first presented the idea of a cycle-consisted generative adversarial network in 2017. They created their CycleGAN for the purpose of translating images. Examples of image translation include things like changing an image of a horse into an image of a zebra by changing the colours of the horse or changing a realistic photograph into a specific painterly style. However, paired image data generally does not exist for tasks such as these. So, they worked on finding a way to convert images from one domain into another domain, without being able to train on paired input and output data. To still be able to train a model in a supervised manner, they make the assumption that there is an underlying relationship between the different domains and then learn that relationship. So, they supervise at the level of the data sets. There is a set of images X and a set of images Y. The generator gets an image from set X as input and should convert this in an image that is indistinguishable from images in set Y. The adversarial discriminator model should thus learn to distinguish the image data from set Y and the converted images from set X.

However, in practice this does not work. The optimal generator should produce an output distribution over the converted images that matches distribution of the images of set Y. However, there are many different mappings of the generator that create the same distribution over the converted data which do not necessarily pair up the input and the output in a meaningful way. This method of learning also risks mode collapse, where several different inputs generate the same output image which causes the model to fail to learn.

To solve this issue, they proposed to make the GAN cycle consistent. Instead of just having the generator that changes images from set X into images from set Y, there should be a second generator, which they call a translator, that converts the images from set Y back into images from set X. These two generator functions should be each other's inverse. To assert that the generators are indeed each other's inverse, they added a cycle consistency loss that makes the model ensure that $F(G(x)) \approx x$ and $G(F(y)) \approx y$. Here F() is the translator that maps data from set Y to set X, and G() is the generator that maps data from set X to set Y. [49][5].

2.3.2.3 Generative Adversarial Networks used for ASR Purposes

Below a short overview is given of some example papers that use generative adversarial networks for automatic speech recognition related purposes.

In 2018 Sriram et al. worked on an End-to-End framework that uses a GAN for the purpose of robust speech recognition. They wanted a data-driven approach that does not require a lot of supervision and does not require the creation of new high quality data sets to improve the performance of ASR systems for noisy speech data. The changes that they added to the GAN model included learning to map noisy audio to the same embedding space as clean audio as a way to improve the variance of the model. They also used the Wasserstein distance to train the ASR model. They compared near and far-field audio data. Near-field speech is spoken closeby, while far-field speech is spoken further away which decreases the clarity of the audio, as it may contain more noise. They concluded that while their encoder was not able to to completely outperform the discriminator, they did make some progress. However, the was still a performance difference of the ASR on near-field and far-field audio[44].

In 2019 Dumpala et al. worked on creating a CycleGAN to model natural perturbations in speech. Perturbations in speech are changes to the speech due to the psychological or physical state of the speaker. These include things like laughter, frustration, being out of breath, having a cold, or having a creaky voice. To increase the robustness of ASR systems, they focused on using a CycleGAN based approach to convert perturbed speech into normal speech. The training was done using non-parallel data examples. They tested their model on speech that contains laughter and creaky voices. The selected this data from two data sets that contain spontaneous speech with manual annotations of where the perturbations occur. They tested the speech that was converted by the CycleGAN on four different ASR models, namely Google cloud ASR, IBM ASR, Kaldi ASR using ASpIRE models, and a DeepSpeech model. They concluded that the performance on all four ASR models improves significantly after transforming the perturbed speech data using the CycleGAN. They also concluded that the addition of aperiodic components to the spectral features improves the performance [13].

In 2020 Chen et al. worked on improving speech recognition using GAN-based speech synthesis. Their focus was to use Text-to-Speech synthesis and data sets that only contain text data for the improvement of ASR systems. Generally, speech synthesis creates limited acoustic diversity, they want to combine a GAN with multi-style training to increase this acoustic diversity in the synthesized data. They combine this with a contrastive language model-based data selection technique. They concluded that their approach allowed the ASR model to learn from synthesized data based on mostly unspoken text sources which resulted in a decrease of the WER for a voice-search task[6].

2.4 Voice Conversion

Voice conversion changes the non-linguistic characteristics of the speech, without changing the linguistic contents. So, the actual spoken text stays the same, but things like pitch and other speech characteristics get changed. So, in the case of dysarthric-to-normal voice conversion you try to change the dysarthric speech in such a manner that the speech starts to sound healthy while keeping the exact same words as were originally spoken.

2.4.1 Ethical Drawbacks of Voice Conversion

There are however some ethical drawbacks to voice conversion. As, while there are many applications that use voice conversion for good purposes, there are also risks of people using voice conversion for illegal practices. If people can accurately recreate the voices of for example politicians and combine that with DeepFakes, you could make realistic looking videos that show politicians doing and saying things that they actually did not do and say. This could destroy someone's reputation and cause overall negative effects. On the other hand, there are initiatives such as the ASVspoof initiative that organize spoofing and countermeasures challenges to work on ways to detect DeepFakes[1].

2.4.1.1 ASVspoof Initiative

The ASVspoof initiative hosts yearly challenges, where they define a set of tasks and rules and participants try to create solutions to these tasks. The tasks and summarized results for each year are available to the public. For example the paper by Yamagishi et al. gave a short summary of the ASVspoof challenge of 2021. The challenge focused on developing solutions to detecting spoofed or deepfake speech. The participants had to work on three different tasks using the ASVspoof databases. The first of the three tasks is a logical access task, where synthetic and converted speech needs to be detected in communication systems. The second task is a physical access task, where replay attacks recorded in physical rooms need to be detected. The third task was a speech deepfake task focuses on the detection of spoofed speech in non-ASV scenarios. In total 72 participants and teams worked on the challenge. The results show that the logical

[1]`https://www.asvspoof.org/`

access task showed the most improvement. The other two tasks were considered to be harder. The solutions to the physical access task only showed a small improvement. The results on the deepfake speech task indicate that the models were prone to overfitting[47].

2.4.2 Parallel vs. Non-Parallel Voice Conversion

Most voice conversion systems use parallel data. When training the model to convert the audio between two speakers, it uses the same data prompts of both speakers. However, in the case of dysarthric speakers high quality healthy speech generally does not exist, as most people generally do not record themselves a lot. Thus, the voice conversion model needs to be trained on two different speakers; namely, the dysarthric speaker and a different healthy speaker. This is why it is necessary to use a non-parallel voice conversion system.

2.4.3 Overview of Voice Conversion Approaches

2.4.3.1 Older approaches

This section gives a few examples of older approaches to voice conversion using parallel data, that were used before the use of generative models that were introduced in the last decade such as variational autoencoders and generative adversarial networks.

For example in 1995 Narendranath et al. worked on transforming formants using feedforward neural networks trained using the backpropagation algorithm. They used the formants to represent the features and used a formant vocoder to turn these features back into speech. While this approach worked for some of the speaker characteristics, several other speech characteristics were missing from the converted speech[35].

In 1998 Kain and Macon worked on spectral voice conversion. In their approach they performed the spectral conversion by using linear transformations based on Gaussian mixture models. They trained the parameters using a joint density function. The speech was synthesized using a linear predictive coding based diphone synthesizer. Their approach performed similarly well as other approaches at the time that used Gaussian mixture models, but theirs was more robust when there was smaller training data sets[27].

In 2009 Desai et al. worked on voice conversion using artificial neural networks. They used the artificial neural networks on parallel data to automatically extract the features for the training data. They compared the performance of their model against Gaussian mixture models. They concluded that their approach performed both subjectively and objectively better than the Gaussian mixture models[12].

2.4.3.2 Non-parallel Voice Conversion using Variational Autoencoders

As an example of using VAEs for voice conversion purposes, in 2016 Hsu et al worked on voice conversion using a variational autoencoder. They specifically focused on creating a framework that could deal with non-parallel data. Their encoder learned the speaker independent phonetic representations and their decoder learned to recreate the speech of the designated speaker. They concluded that their framework's ability to convert spectra had a similar performance as baseline systems that were trained on parallel data[26].

In 2018 Kameoka et al. worked on non-parallel many-to-many voice conversion. They used a variant on the normal VAE, namely an auxiliary classifier VAE. The auxiliary part of the VAE ensures that the attribute class label is not lost when converting speech, it does this using an information-theoretic type of regularization when training the model. Their system avoids producing buzzy sounds by transplanting the spectral details of the input during the voice conversion. The subjectively tested their system on a many-to-many identity conversion task. They concluded that it performed reasonably well[28].

In 2019 Lumban Tobing et al. worked on non-parallel voice conversion using a cyclic version of a VAE. They claim that when using VAE-based voice conversion, the converted spectra can not be directly optimized when using non-parallel data. To avoid this they decided to use a CycleVAE. This allows them to optimize the reconstructed spectra that are obtained after converting the already converted features again. After experimenting with their proposed CycleVAE, they concluded that the accuracy of the converted spectra is higher, the latent features that are generated have a higher correlation degree and that the quality and conversion accuracy of the converted speech was significantly improved[45].

2.4.3.3 GAN-based Approaches to Voice Conversion

As an example of using GANs for voice conversion purposes, in 2017 Kaneko et al. worked on a CycleGAN for non-parallel voice conversion called CycleGAN-VC. It uses gated convolutional neural networks, this is to keep track of the sequential and hierarchical structures of speech. The model is trained using an identity-mapping loss. The identity mapping loss ensures that the linguistic information does not get lost during training. They evaluated the system both subjectively and objectively. They concluded that objectively the converted speech sounded near natural. Subjectively, the speech quality is just as good compared to a Gaussian mixture model that uses a larger parallel data set. However, the converted speech is not yet at the same level as natural speech[30].

After this in 2019 Kaneko et al. proposed an improved version of this CycleGAN, CycleGAN-VC2. The improvements that they proposed included using a two-step adversarial loss as objective instead of a one-step adversarial loss. This is used to help with the problem of oversmoothing. Next to that they used a different generator and a different discriminator. For the generator they are using a 2-1-2D CNN instead of a 1 dimensional CNN. 1D CNNs are better at capturing dynamic changes, while 2D CNNs are better at preserving the original structures of the speech during conversion. The 2-1-2D CNN combines the two different CNN types, it uses the 2D part for the up and down-sampling and the 1D part for the conversion process. For the discriminator it uses a PatchGAN instead of a FullGAN. A FullGAN uses a fully connected layer as the last layer of the network, while the PatchGAN uses a convolution layer as the last layer of the network. They concluded that their new version of the CycleGAN both sounds more natural and more similar for each speaker pair than the original CycleGAN. However, it is limited to mel-cepstrum conversion and cannot do mel-spectrogram conversion[31].

As an further improvement to the CycleGAN to allow it to also convert mel-spectrograms, in 2021 Kaneko et al. worked on both a new version of CycleGAN-VC2, and an alternative model called MaskCycleGAN-VC. CycleGAN-VC3 uses time-frequency adaptive normalization, however this causes the number of parameters to almost double in size. The alternative improved model, MaskCycleGAN-VC, uses an auxiliary task called filling in frames. This is a self-supervised approach in which the converter learns the time-frequency structures by filling in temporal masks in the mel-spectrogram based on the surrounding frames. This avoids the issue of having too many parameters. They concluded that the MaskCycleGAN-VC performs better than both the second and third version of the CycleGAN-VC[29].

3 Methods

3.1 Dataset

In this work, we use the UASpeech dataset which was created by the University of Illinois[25]. This dataset is freely available to researchers at universities and government labs. However, explicit permission to use the data needs to be granted beforehand. After receiving this permission the data can only be used for scientific purposes and the data may not be shared outside the university. The dataset contains isolated word prompts that are read from a computer monitor by 15 different participants with Cerebral Palsy. The same isolated word prompts are also read by 13 age-matched healthy speakers. The prompts include the following words:

- digits (10 words, 3 repetitions),
- the 26 letters of the International Radio Alphabet (26 words, 3 repetitions),
- computer commands, such as "paragraph" or "enter" (19 words, 3 repetitions),
- common words, such as "the" or "and" (100 words, 3 repetitions),
- uncommon words, such as "faithfulness" (300 words, 1 repetition)

At the moment of recording, the 15 speakers were between the ages of 18 and 58 and included 4 female speakers and 11 male speakers. Twelve speakers had spastic dysarthria, two had athetoid dysarthria and one had a mix of the two different types. The spastic dysarthria corresponds to the spastic dysarthria as described in the introduction, the athetoid dysarthria is also known as dyskinetic cerebral palsy. It causes slow and uncontrolled movements, which places this in the hyperkinetic dysarthria as described in the introduction. The level of intelligibility differed per speaker. The dataset contained four different categories of intelligibility, these were defined as very low (0-25%), low (25-50%), mid (51-75%) and high (76-100%). In the dataset 4 speakers had a very low intelligibility between 2% and 15%, 3 had a low intelligibility between 28% and 43%, 3 had a medium intelligibility between 58% and 62%, and the remaining 5 speakers had a high intelligibility between 86% and 95%[25].

The intelligibility per speaker was determined by the average score of a word transcription task by 5 different listeners. This test was done to measure the overall intelligibility and did not focus on specific elements of the speech such as phonetic features. For each speaker in the dataset five native American English speakers between the ages of 18 to 40 were recruited, they each had no experiences with speech disorders, had no language disabilities and had no training in phonetic transcription. They were told that they had to give orthographic transcriptions of words spoken by a speaker with a speech disorder. They listened to the words in a quiet room using headphones and they were allowed to listen to each word as often as necessary. They had to give a transcription for each word and additionally had to indicate the certainty of their transcription with a score between 0 and 2, the former being unsure and the latter being sure. In total they listened to 200 unique speech files, including 10 digits, 25 radio alphabet letters, 19 computer commands, 73 random words for the common words category and 73 random words from the uncommon words category. Lastly, 25 words were arbitrarily selected and repeated twice in the list. The word list was presented in a random order, however repeated words were not adjacent to each other. This was used as an intra-listener reliability assessment. In the cases where the listeners were very sure, the agreement rate between the transcriptions for the same word was 91.64%, for the words with a lower rating, the transcriptions were either identical or phonetically similar. The average intelligibility per speaker was calculated by calculating the percentage of correct transcriptions per listener and then averaging over the five different listeners. [25].

The word prompts were recorded in three blocks, each block contained each digit, letter, command and common word once and 100 of the uncommon words. The prompts were recorded on a 7-channel microphone at a sampling rate of 48 kHz, resulting in 7 recordings per prompt. This leads to a total of 5355 recordings per speaker, and a total of 80325 audio files for the dysarthric speakers and 69615 audio files for the healthy control speakers. The dataset was already split into a train/test split based on these blocks. Blocks B1 and B3 were used for training and block B2 was used for testing. This means that the test set contains 155 words that already occurred in the training set and 100 new words.

3.2 Baseline Model

To see whether the changes to the audio files do positively influence the performance on the ASR system a baseline model is needed. This baseline model will thus, be used to compare the influence of the experiments on the results of the ASR system. The baseline model shows the performance of the ASR model on the healthy data and on the data of the dysarthric speakers. The performance on the healthy data shows the upper bound of the performance on the ASR model, other models are not expected to outperform this baseline. The baseline models on the dysarthric speech show the lower bound of the experiment. In order to improve the audio created by the experiments needs to perform better than this baseline on the ASR system. To create the baseline model, three different datasets were tested on a pretrained English ASR model[1]. To ensure that the audio files of the baseline model and the audio files of the later experiments are in the exact same format before inserting them in the ASR, they all go through the same data preprocessing pipeline. This pipeline is described in more depth in the experiments section. In this preprocessing pipeline the audio files are loaded with a fixed sampling rate of 22050 Hz and are mixed to mono. Then afterwards the audio files are saved as Mel-spectrograms and stored in a pickle file. In the next step, this pickle file containing all the Mel-spectrograms is used to generate the audio files that are inserted in the ASR system for the testing purposes.

The first dataset that is tested on this ASR model is contains the speech data of one of the speakers of the control group of the UASpeech dataset, specifically speaker CM01. This dataset contains healthy speech data in the same format as the dysarthric speech data. This way, the upper bound on the achievable performance of the ASR system can be tested.

The other two datasets that are used are part the dysarthric speech dataset. The dataset contains speakers with either a very low, low, medium or high level of intelligibility, for this experiment two speakers were selected from the middle two categories. Namely speaker M05 and speaker M07. Speaker M05 was 21 years old at the time of recording, uses a wheelchair, has a medium level of intelligibility of 58% and has a spastic type of dysarthria. Speaker M07 was 58

[1] `https://huggingface.co/jonatasgrosman/wav2vec2-large-xlsr-53-english`

years old at the moment of recording, uses a wheelchair and is able to sign, has a low level of intelligibility of 28% and has a spastic type of dysarthria. For the baseline experiment, the data of speaker M05 and M07 will be tested on the same ASR system as the control data. These two speakers were chosen for the baseline model and the further experiments to best represent the most common elements of the dataset, as most of the participants are male speakers with a spastic type of dysarthria. Due to time constraints the experiments could not be performed on all the speakers in the dataset. Thus, the two speakers were chosen from the middle two intelligibility categories. These categories were chosen instead of the very low and high category, to make sure in the case of the very low category that the speech was at least slightly understandable making it less hard to improve upon than when the speech is completely unintelligible. Furthermore, this speaker was chosen instead of a speaker with a high level of intelligibility since those speakers may already perform quite well on the ASR system, making it harder to see improvements.

3.3 Experiments

The main experiment that was run on the data consists of two parts, voice conversion and time-stretching. First of all for the voice conversion a generative adversarial network will be trained on separately on both dysarthric speakers' datasets. Then, using time-stretching the audio files will be sped up using different factors, namely 0.75, 1.0, 1.25, 1.5, 1.75 and 2.0. Then the previously trained GAN model will be used to convert the audio files for each time-stretching factor. Thus, the comparisons will be between the effects of the conversion and the time-stretching factors. For each speaker, there will be twelve comparisons.

For the voice conversion part of the experiment, the GitHub repository, MaskCycleGAN-VC[2][8], was used. This repository contains an unofficial PyTorch implementation of Kaneko et al.'s MaskCycleGAN-VC[32]. For the time-stretching Librosa's phase vocoder is used [3]. The MaskCycleGAN was mostly used as can be found on the GitHub repository, but some small changed had to be made to include the time-stretching elements and to allow the code to work with the dysarthric speech data. All the changes to the existing code that were made in order to do the experiments are stored on GitHub [4].

Two different models were trained, one for each of the two dysarthric speakers. The model for each speaker was trained before applying any time-stretching. The model was trained for 25 epochs on the entire dataset, which includes all seven different recordings of each word prompt. The conversion was trained between the control speaker CM01 and the respective dysarthric speaker.

After training the models, the time-stretching was implemented. The time-stretching was implemented as part of the preprocessing pipeline. In the original preprocessing pipeline the audio files are read as mono audio with a fixed sample rate of 22050 Hz. Then afterwards the audio file is run through a vocoder to get the mel-spectrograms of the audio files. Then 64 randomly cropped frames were selected. This list is used to calculate the mean and standard deviation. Next, the mel-spectrograms are normalized and stored in a pickle file. This pickle file will then later be used in the testing part of the code to generate the original and converted audio files.

The best location to add the time-stretching was during this preprocessing process, specifically between the loading the audio files and running the audio file through the vocoder to create the mel-spectrograms. The actual code for the time-stretching contains three steps. First, it uses

[2]`https://github.com/GANtastic3/MaskCycleGAN-VC`

[3]`https://librosa.org/doc/latest/generated/librosa.phase_vocoder.html`

[4]`https://github.com/MarjoleinSpijkerman/Voice-Technology-2022-Master-Thesis-Marjolein-Spijkerman`

the librosa stft function to calculate the short-time fourier transform of the audio. Secondly, it uses the librosa phase vocoder to speed up the audio by a given factor. Than lastly, it uses the librosa istft function to calculate the inverse short time fourier transform. This is to transform the spectrogram back into time series.

In the case that the length of the audio is too small to create the 64 frames, the audio will be skipped. For time-stretching factors that are higher than 2.0, some of the audio files will become too short to create this 64 frames. Due to this reason, the factor 2.0 is the highest time-stretching factor that will be used for this experiment.

At the end of the preprocessing pipeline the code has generated the pickle files containing the mel-spectrograms of all the audio. At this point we have six pickle files per speaker, each containing a different speed factor. From here on the test code can be run, this part of the code uses the trained model to generate both the original and converted files from the pickle files. These can then be inserted in the ASR model to transcribe the audio files.

After using the ASR model to create the predictions, both the Word Error Rate (WER) and the Character Error Rate (CER) of each set will be calculated.

3.4 Word and Character Error Rate

The Word Error Rate (WER) is a evaluation metric that is often used to evaluate the performance of tasks such as speech recognition and machine translation. The WER calculates the difference between two texts at word level. Specifically it calculates the Levenshtein distance between two texts. Namely, it calculates the number of insertions, deletions ans substitutions that need to be performed to change one sentence into another sentence. In the formula below, S is the number of substitutions, D is the number of deletions, I is the number of insertions and N is the number of words in the reference.

$$\text{WER} = \frac{S + D + I}{N}. \tag{3.1}$$

A completely correct predictions returns a word error rate of 0. Yet, since the number of steps needed can be larger than the number of words in the reference, it is possible to get a WER that is higher than 1. It is a useful metric for evaluating the usability of a system as you generally want predictions that are correct at word level. It can also serve as a comparison tool between two systems. While the WER gives a good overview of the performance of a model, it does not specify the types of errors that are made. In the case of the dysarthric speech dataset all the word prompts are single words. This means that according to the WER each predict will either completely wrong or completely correct. This is reasonable in the case of a large error, but when the error between the reference and the prediction is very small and it still gets the same WER, it does not give a good view of the actual performance of the model. To illustrate this with an example, the word "cat" and "bat" would be considered completely erroneous with the WER, while it is only a small error when you look at the word at a character level. This is why the Character Error Rate (CER) of each word will also be calculated as part of the evaluation. The CER will be able to show the results and improvements at a more granular, character level. The CER is calculated in almost the same manner as the WER, only this time it is calculated on a character level. Furthermore, it also counts errors made in the spaces between words as errors.

To evaluate the results, the WER is calculated by using the WER function as defined in the code of ESPNET[5], this function used the editdistance library[6]. The editdistance library calculates the Levenshtein distance between two text strings. The Levenshtein distance is defined as follows:

$$\text{lev(a, b)} = \begin{cases} |a| & \text{if } |b| = 0 \\ |b| & \text{if } |a| = 0 \\ \text{lev (tail(a), tail(b))} & \text{if } a[0] = b[0] \\ 1 + \min \begin{cases} \text{lev(tail(a), b)} \\ \text{lev(a, tail(b))} \\ \text{lev(tail(a), tail(b))} \end{cases} & \text{otherwise} \end{cases} \tag{3.2}$$

Here the tail of a string is the entire string minus the first character of the string. So, basically the function recursively compares the first character of each string. If the absolute length of either string is 0 the end of the comparison is reached. If not and both strings have the same first character, the algorithm is continued after removing the first character of each string. And in all other cases, it adds 1 to the error count and continues recursively the path that has the least errors. These paths include:

- **tail(a), b:** here you remove the first letter of string "a", while keeping string "b" as it is. This corresponds to insertion. As you need one additional character in string "b", to get the same length as string "a"
- **a, tail(b):** this is the opposite of "tail(a), b", meaning that this is deletion
- **tail(a), tail(b):** this step is done for substitution. The length is the same, but the characters are different.

The function on ESPNET uses the editdistance function to calculate the WER. This function takes two lists of strings as input, one containing the original transcription and one containing the predictions. Then at word level it compares each sentence. It stores the number of words in each original sentence and the lowest distance between two sentences. At the end of the function it returns the WER. For easier comparison the WER will be multiplied by 100 to get the percentages. As the implementation of the code does consider lowercase letters and capital letters as different, all the predictions and transcriptions will be inserted in the WER calculator in lowercase. The calculation of the CER was done using the same function, but instead of using a list of words as input for the editdistance function, a single string is used. This returns the CER instead of the WER.

[5]`https://github.com/espnet/espnet/blob/b008ac7d58e9ced1a9f8c89cc85ee69d9e9461ab/espnet/nets/e2e_asr_common.py`

[6]`https://pypi.org/project/editdistance/0.3.1/`

4 Results

The baseline results are summarized in table 4.1. This includes the WER and the CER of the control data. The results for the voice conversion and time-stretching experiments on the dysarthric speech are summarized in tables 4.2 and 4.3

Baseline Results	**WER**	**CER**
Control data (CM01)	69.69%	30.53%
Medium Intelligibility Speaker (M05)	241.34%	158.42%
Low Intelligibility Speaker (M07)	232.49%	162.03%

Table 4.1: Word Error Rate (WER) and Character Error Rate (CER) in percentage for the control data (Speaker CM01) and the dysarthric speakers M05 and M07

Speaker M05	**WER**		**CER**	
Speed rate	**Original**	**Converted**	**Original**	**Converted**
0.75	249.92%	317.09%	170.81%	223.72%
1.0	241.34%	288.52%	158.42%	196.1%
1.25	248.12%	261.68%	160.06%	175.48%
1.5	238.32%	236.69%	151.42%	156.48%
1.75	233.56%	216.08%	148.95%	143.87%
2.0	219.83%	**199.83%**	138.7%	**133.53%**

Table 4.2: Word Error Rate (WER) and Character Error Rate (CER) of the experiments performed on the data by speaker M05 (medium level intelligibility).

Speaker M07	**WER**		**CER**	
Speed rate	**Original**	**Converted**	**Original**	**Converted**
0.75	237.37%	268.4%	169.95%	195.05%
1.0	232.49%	267.11%	162.03%	189.44%
1.25	230.92%	252.04%	160.11%	174.74%
1.5	224.59%	228.74%	152.16%	159.15%
1.75	213.05%	208.96%	145.44%	147.03%
2.0	202.46%	**195.52%**	139.2%	**136.72%**

Table 4.3: Word Error Rate (WER) and Character Error Rate (CER) of the experiments performed on the data by speaker M07 (low level intelligibility).

Figure 4.1 and 4.2 contain graphs of the WER and the CER of both datasets for comparison purposes.

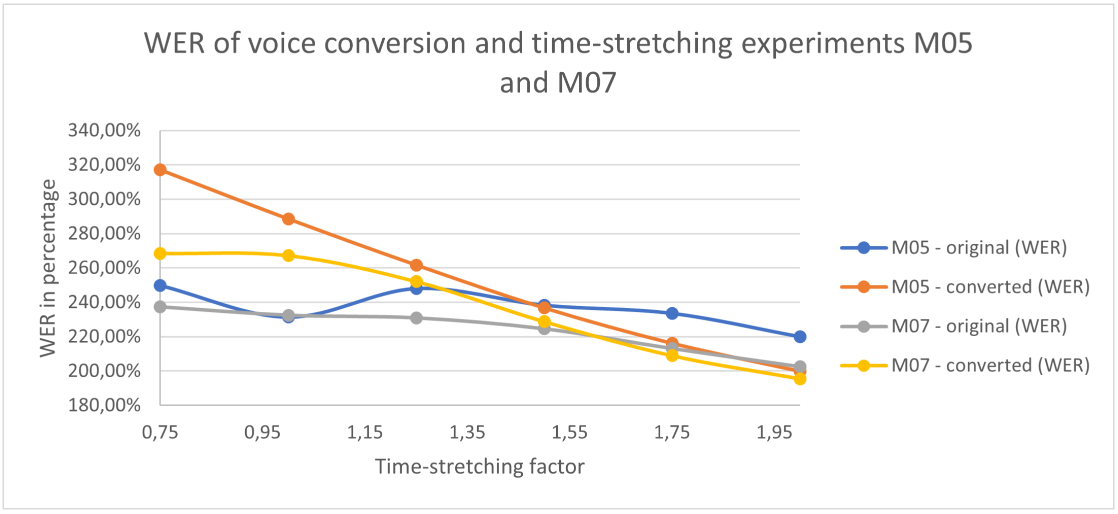

Figure 4.1: The WER of the experiments done on the data of speaker M05 and M07

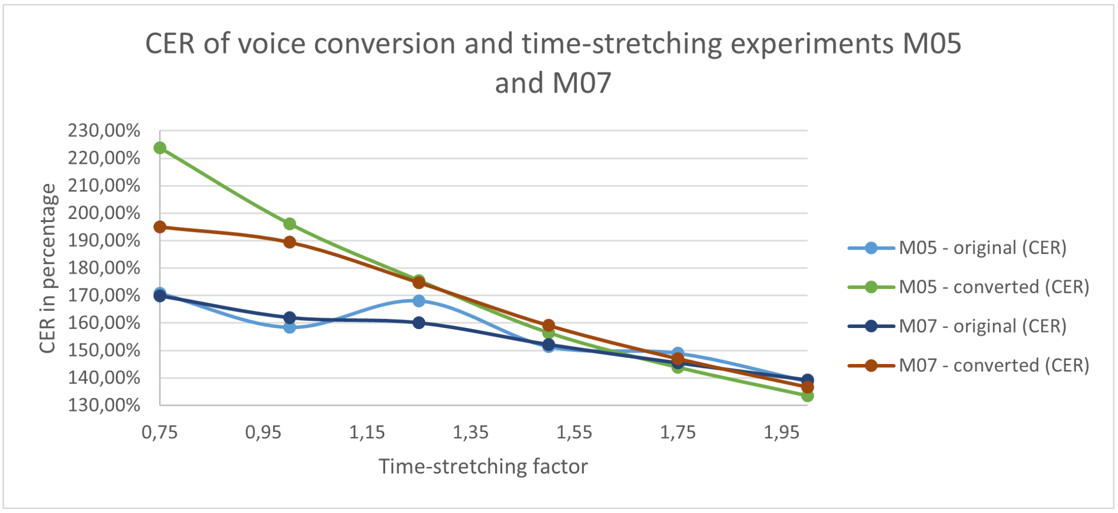

Figure 4.2: The CER of the experiments done on the data of speaker M05 and M07

5 Discussion

The results show that the control data has both the lowest WER and the lowest CER, which was to be expected. The control data was used as the upper bound of the performance, it was unlikely that any of the dysarthric speech data would perform better than the healthy control data. Furthermore, the CER was higher than the WER. This also agreed with the expectation, as the dataset contains only single word prompts, so each prompt would return a 0 or 1 in the case of WER. The CER is more than half as low as the WER, this is due to the fact that the ASR system predicts a lot of words almost correct. For example, it predicted the word "backspace" as "bekspace", which is a large error according to the WER, but a reasonably small error according to the CER.

The results show that the control data has the lowest WER, which was to be expected. All the dysarthric speech data performs quite a lot worse than the control data. For the lower speeds, the performance of the original data is better than the converted data, however at the higher speeds the converted data starts to outperform the original non-converted datasets. While the WER is going down, the actual predictions by the ASR system are not accurate yet, as it does not predict any words completely correctly. However, for the lower speeds the ASR system tends to predict either words that are too long or multiple words. In the case for the converted speech at a speed rate of 2.0, the system predicts less words or shorter words, which causes the WER to go down as well. So, while it is definitely a step in the right direction a lot of work still needs to be done in order to get better results.

The results of the dysarthric speech partially matches the expectations. As expected both the WER and the CER improve as the speed rate goes up. This matches the general expectation that spastic dysarthria generally has a low speed rate. The voice conversion only shows a very small improvement over the non-converted audio in the later cases. The voice conversion thus does not work for the regular speed rates. Thus, the conversion does most likely not fully fix the other issues that are considered part of spastic dysarthria, namely the imprecise pronunciation of consonants, the monotony in pitch and loudness and the reduced stress on words. Other issues such as patients having trouble with longer sentences are most likely not visible in this dataset, as the dataset only contains single word prompts.

The expectation that was not met is regarding the difference between the two speakers, M05 and M07. As M05 had an almost twice as high intelligibility level compared to speaker M07, the expectation was that this would be visible in the performance on the ASR model. However, the two models perform roughly the same. Speaker M05 has a better final CER, while speaker M07 has a better final WER. But other than that there are not many differences between the results of the two speakers. The faster the speed rate, the more similar the two models start performing. This is also clearly visible in both figures, but specifically in figure 4.2, which shows the CER. All four models at speed rate 2.0 go to almost the same point.

The reason that the WER and the CER are decreasing, as the speed rate is increasing is most likely due to the system predicting too many words for the slower audio files. As the speed rate goes up, the number of predictions goes down and thus decreases the WER and the CER. But, even though this rate is going down, the ASR does not actually predict any words completely correct. It is however slowly moving towards a slightly more accurate prediction than before applying the time-stretching and voice conversion methods.

To conclude, the hypothesis was partially correct. The performance does indeed improve for the combination of voice conversion and high level time-stretching. However, the data by speaker M05 does not perform better than speaker M07.

Due to the large improvements of the results as the speed rate is increased it would have been interesting to test for higher speed rates. However, due to the short length of some of the audios increasing the speed rate even further would have made some of the audio files too short to go through the preprocessing pipeline. The experiment could still have been performed, but not on the entire dataset. For better comparisons, it was decided to skip these higher speed rates so that the same set of audio files could be compared.

Lastly, to compare with Halpern et al.'s[37] work, the performance of my experiments in all categories is worse. However, the data does seem to show the same trend that applying time-stretching experiments do improve the performance. Furthermore, the baseline model on the control data also performs worse, this is most likely due to the basic ASR model that was used. So, this could also affect the final results.

6 Conclusion

To conclude, the data from two different speakers from a dysarthric speech dataset was tested on a pretrained English ASR model after applying voice conversion using the MaskCycleGAN-VC model combined with time-stretching. The expectation was that the combination of time-stretching combined with the voice conversion would improve the performance on the ASR model. This turned out to be true, the case with the highest level of time-stretching combined with the voice conversion performed the best of all cases on both the level of WER and CER. However, the expectation that the speaker with the higher intelligibility level would perform better did not end up being true. Other conclusions that can be made is that the WER of the baseline model of the control data is quite high. Thus for future research, it can be interesting to try out different ASR models. Next to that, the MaskCycleGAN-VC was only trained for 25 epochs on the regular data due to time constraints, it could be interesting to see if the performance would improve if we were to train the model for several days instead. And lastly, the training was performed on data that was not time-stretched. It could be interesting to see if time-stretching the training data before training would further influence the performance on the ASR model.

Bibliography

[1] Background: What is a generative model? https://developers.google.com/machine-learning/gan/generative. Accessed: 11-7-2022.

[2] David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for boltzmann machines. *Cogn. Sci.*, 9:147–169, 1985.

[3] Shipra J. Arora and Rishi Pal Singh. Automatic speech recognition: A review. *International Journal of Computer Applications*, 60:34–44, 2012.

[4] Yoshua Bengio, Eric Thibodeau-Laufer, and Jason Yosinski. Deep generative stochastic networks trainable by backprop. *CoRR*, abs/1306.1091, 2013.

[5] Jason Brownlee. A gentle introduction to cyclegan for image translation. https://machinelearningmastery.com/what-is-cyclegan/. Accessed: 15-7-2022.

[6] Zhehuai Chen, Andrew Rosenberg, Yu Zhang, Gary Wang, Bhuvana Ramabhadran, and Pedro J. Moreno. Improving speech recognition using gan-based speech synthesis and contrastive unspoken text selection. In *INTERSPEECH*, 2020.

[7] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition, 2015.

[8] Sofian Zalouk Claire Pajot, Hikaru Hotta. Maskcyclegan-vc. https://github.com/GANtastic3/MaskCycleGAN-VC, 2021.

[9] Aronson A. E. Brown J. R. Darley, F. L. Differential diagnostic patterns of dysarthria. *Journal of speech and hearing research*, 12(2):246–269, 1969.

[10] Luigi De Russis and Fulvio Corno. On the impact of dysarthric speech on contemporary asr cloud platforms. *Journal of Reliable Intelligent Environments*, 5(3):163–172, Sep 2019.

[11] Robert DeRosier and Ruth S Farber. Speech recognition software as an assistive device: a pilot study of user satisfaction and psychosocial impact. *Work*, 25(2):125–134, 2005.

[12] Srinivas Desai, E. Veera Raghavendra, B. Yegnanarayana, Alan W. Black, and Kishore Prahallad. Voice conversion using artificial neural networks. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3893–3896, 2009.

[13] Sri Harsha Dumpala, Imran Sheikh, Rupayan Chakraborty, and Sunil Kumar Kopparapu. A cycle-gan approach to model natural perturbations in speech for asr applications, 2019.

[14] Siyuan Feng, Olya Kudina, Bence Mark Halpern, and Odette Scharenborg. Quantifying bias in automatic speech recognition. *arXiv preprint arXiv:2103.15122*, 2021.

[15] Andrew Glassner. *Deep Learning: A Visual Approach*, chapter Generative Adversarial Networks, page 649–673. No Starch Press, 1st edition, 2021.

[16] Andrew Glassner. *Deep Learning: A Visual Approach*, chapter Recurrent Neural Networks, pages 539–564. No Starch Press, 1st edition, 2021.

[17] Andrew Glassner. *Deep Learning: A Visual Approach*, chapter Attention and Transformers, pages 565–599. No Starch Press, 1st edition, 2021.

[18] Andrew Glassner. *Deep Learning: A Visual Approach*, chapter Neural Networks, page 314–349. No Starch Press, 1st edition, 2021.

[19] Andrew Glassner. *Deep Learning: A Visual Approach*, chapter Backpropagation, page 351–386. No Starch Press, 1st edition, 2021.

[20] Andrew Glassner. *Deep Learning: A Visual Approach*, chapter Optimizers, page 387–425. No Starch Press, 1st edition, 2021.

[21] Andrew Glassner. *Deep Learning: A Visual Approach*, chapter Autoencoders, pages 495–538. No Starch Press, 1st edition, 2021.

[22] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*, chapter Back-Propagation and Other Differentiation Algorithms, pages 200–220. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[23] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[24] Yanzhang He, Tara N. Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, Qiao Liang, Deepti Bhatia, Yuan Shangguan, Bo Li, Golan Pundak, Khe Chai Sim, Tom Bagby, Shuo-yiin Chang, Kanishka Rao, and Alexander Gruenstein. Streaming end-to-end speech recognition for mobile devices, 2018.

[25] Adrienne Perlman Jon Gunderson Thomas Huang Kenneth Watkin Heejin Kim, Mark Hasegawa-Johnson and Simone Frame. Dysarthric speech database for universal access research. *In Proc. Interspeech*, pages pp. 1741–1744, 2008.

[26] Chin-Cheng Hsu, Hsin-Te Hwang, Yi-Chiao Wu, Yu Tsao, and Hsin-min Wang. Voice conversion from non-parallel corpora using variational auto-encoder. pages 1–6, 12 2016.

[27] A. Kain and M.W. Macon. Spectral voice conversion for text-to-speech synthesis. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181)*, volume 1, pages 285–288 vol.1, 1998.

[28] Hirokazu Kameoka, Takuhiro Kaneko, Kou Tanaka, and Nobukatsu Hojo. Acvae-vc: Non-parallel many-to-many voice conversion with auxiliary classifier variational autoencoder, 2018.

[29] Takuhiro Kaneko, H. Kameoka, Kou Tanaka, and Nobukatsu Hojo. Maskcyclegan-vc: Learning non-parallel voice conversion with filling in frames. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5919–5923, 2021.

[30] Takuhiro Kaneko and Hirokazu Kameoka. Parallel-data-free voice conversion using cycle-consistent adversarial networks. 11 2017.

[31] Takuhiro Kaneko, Hirokazu Kameoka, Kou Tanaka, and Nobukatsu Hojo. Cyclegan-vc2: Improved cyclegan-based non-parallel voice conversion. pages 6820–6824, 05 2019.

[32] Takuhiro Kaneko, Hirokazu Kameoka, Kou Tanaka, and Nobukatsu Hojo. Maskcyclegan-vc: Learning non-parallel voice conversion with filling in frames. *CoRR*, abs/2102.12841, 2021.

[33] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.

[34] Victoria Young MHSc and Alex Mihailidis PhD. Difficulties in automatic speech recognition of dysarthric speakers and implications for speech-based applications used by the elderly: A literature review. *Assistive Technology*, 22(2):99–112, 2010. PMID: 20698428.

[35] M. Narendranath, Hema A. Murthy, S. Rajendran, and B. Yegnanarayana. Transformation of formants for voice conversion using artificial neural networks. *Speech Communication*, 16(2):207–216, 1995. Voice Conversion: State of the Art and Perspectives.

[36] Andrew Ng and Michael Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Adv. Neural Inf. Process. Sys*, 2, 04 2002.

[37] Luke Prananta, Bence Mark Halpern, Siyuan Feng, and Odette Scharenborg. The effectiveness of time stretching for enhancing dysarthric speech for improved dysarthric speech recognition. *CoRR*, abs/2201.04908, 2022.

[38] Kanishka Rao, Hasim Sak, and Rohit Prabhavalkar. Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer. *CoRR*, abs/1801.00841, 2018.

[39] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models, 2014.

[40] Frank Rudzicz. Adjusting dysarthric speech signals to be more intelligible. *Computer Speech Language*, 27(6):1163–1177, 2013. Special Issue on Speech and Language Processing for Assistive Technology.

[41] Haşim Sak, Andrew Senior, Kanishka Rao, and Françoise Beaufays. Fast and accurate recurrent neural network acoustic models for speech recognition, 2015.

[42] Ruslan Salakhutdinov and Geoffrey E. Hinton. Deep boltzmann machines. In *AISTATS*, 2009.

[43] speechpathologygraduateprograms.org. How speech language pathologists treat patients with dysarthria - what is dysarthria? `https://www.speechpathologygraduateprograms.org/dysarthria/`. Accessed: 1-7-2022.

[44] Anuroop Sriram, Heewoo Jun, Yashesh Gaur, and Sanjeev Satheesh. Robust speech recognition using generative adversarial networks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5639–5643, 2018.

[45] Patrick Tobing, Yi-Chiao Wu, Tomoki Hayashi, Kazuhiro Kobayashi, and Tomoki Toda. Non-parallel voice conversion with cyclic variational autoencoder. pages 674–678, 09 2019.

[46] Bhavik Vachhani, Chitralekha Bhat, and Sunil Kumar Kopparapu. Data augmentation using healthy speech for dysarthric speech recognition. pages 471–475, 09 2018.

[47] Junichi Yamagishi, Xin Wang, Massimiliano Todisco, Md Sahidullah, Jose Patino, Andreas Nautsch, Xuechen Liu, Kong Aik Lee, Tomi Kinnunen, Nicholas Evans, and Héctor Delgado. Asvspoof 2021: accelerating progress in spoofed and deepfake speech detection. pages 47–54, 09 2021.

[48] Emre Yılmaz, Vikramjit Mitra, Ganesh Sivaraman, and Horacio Franco. Articulatory and bottleneck features for speaker-independent asr of dysarthric speech. *Computer Speech Language*, 58:319–334, 2019.

[49] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2017.