

**Evaluation of STT technologies performance and database  
design for Spanish dysarthric speech**

*Thesis to be submitted in partial fulfillment of the  
requirements for the degree*

*of*

**Master on Language Analysis and Processing**

*by*

**Margot Madina González**

Under the guidance of

**Inmaculada Hernáez and Eva Navas**



Universidad del País Vasco      Euskal Herriko  
Unibertsitatea

# ACKNOWLEDGEMENTS

First of all, I would like to express my gratitude to my supervisors Inma Hernández and Eva Navas. Thank you for your guidance, patience and support during all this time. You have always been there to solve my doubts and lead me in this project.

This project would not have been possible without the participation of the people who constitute our speech database. I must express my gratitude for their help and dedication. I am aware that they have put so much effort and they have invested a lot of time in their role.

I would also like to acknowledge the assistance of Aitor in the creation of the implementation app to conduct some of the experiments. This step could not have been done without his help.

Thank you to all those who took some of their time to take the perceptual test. The information collected through the answers provided has been very useful in this project.

Last, but not least, thank you to my parents and Julen, for always being there.

# ABSTRACT

Automatic Speech Recognition (ASR) systems have become an everyday use tool worldwide. Their use has spread throughout these last years and they have also been implemented in Environmental Control Systems (ECS) or Speech Generating Devices (SGD), among others. These systems might be especially beneficial for people with physical disabilities, as they would be able to control different devices with voice commands, therefore reducing the physical effort they have to make. However, people with functional diversity usually present difficulties in speech articulation too. One of the most common speech articulation problems is dysarthria, a disorder in the nervous system which causes weakness in muscles used for speech. Existing commercial ASR systems are not able to correctly understand dysarthric speech, so people with this condition cannot exploit this technology. Some investigation tackling this issue has been conducted, but an optimal solution has not been reached yet. On the other hand, nearly all existing investigation on the matter is in English, no previous study has approached the problem in other languages. Apart from this, ASR systems require of large speech databases, which are currently very few, most of them in English and they have not been designed for this end. Some commercial ASR systems offer a customization interface where users can train a base model with their speech data and thus improve the recognition accuracy. In this thesis, we evaluated the performance of the commercial ASR system Microsoft Azure Speech to Text. First, we reviewed the current state of the art. Then, we created a pilot database in Spanish and recorded it with 3 heterogeneous people with dysarthria and 1 typical speaker to be used as reference. Lastly, we trained the system and conducted different experiments to measure its accuracy. Results show that, overall, the customized models outperform the base models of the system. However, the results were not homogeneous, but vary depending on the speaker. Even though the recognition accuracy improved considerably, the results were far from being as good as those obtained for typical speech.

**Keywords:** automatic speech recognition, dysarthria, intelligibility, Spanish

# List of Figures

4.1 Perceptual test for human evaluation . . . . .	39
5.1 R1 recognized commands . . . . .	43
5.2 C1 recognized commands . . . . .	44
5.3 M1 recognized commands . . . . .	44
5.4 W1 recognized commands . . . . .	44
5.5 AVERAGE recognized commands . . . . .	44
5.6 WER per speaker and model . . . . .	48
5.7 R1 BASE MODEL confidence . . . . .	50
5.8 R1 BASE MODEL WITH GRAMMAR confidence . . . . .	50
5.9 R1 CUSTOM MODEL confidence . . . . .	50
5.10 R1 CUSTOM MODEL WITH GRAMMAR confidence . . . . .	50
5.11 C1 BASE MODEL confidence . . . . .	51
5.12 C1 BASE MODEL WITH GRAMMAR confidence . . . . .	51
5.13 C1 CUSTOM MODEL confidence . . . . .	51
5.14 C1 CUSTOM MODEL WITH GRAMMAR confidence . . . . .	51
5.15 M1 BASE MODEL confidence . . . . .	52
5.16 M1 BASE MODEL WITH GRAMMAR confidence . . . . .	52
5.17 M1 CUSTOM MODEL confidence . . . . .	52
5.18 M1 CUSTOM MODEL WITH GRAMMAR confidence . . . . .	52
5.19 W1 BASE MODEL confidence . . . . .	53
5.20 W1 BASE MODEL WITH GRAMMAR confidence . . . . .	53
5.21 W1 CUSTOM MODEL confidence . . . . .	53
5.22 W1 CUSTOM MODEL WITH GRAMMAR confidence . . . . .	53
5.23 Easiest and most difficult words . . . . .	56
5.24 Implementation app recognition results . . . . .	58
5.25 Implementation app WER per speaker . . . . .	59

# List of Tables

<a href="#">3.1 Azure Custom Speech training use cases and data types</a> . . . . .	23
<a href="#">3.2 Amazon Web Services, how to upload data</a> . . . . .	28
<a href="#">5.1 WER value per model</a> . . . . .	47
<a href="#">5.2 Human evaluation results</a> . . . . .	54
<a href="#">5.3 Implementation app results</a> . . . . .	59

# List of acronyms

- ACC** Alternative and Augmentative Communication. [61](#)
- ALS** Amyotrophic Lateral Sclerosis. [2](#), [7](#), [20](#)
- APAC** Assessment of Phonology and Articulation for Children. [18](#)
- ASR** Automatic Speech Recognition. [viii](#), [1-4](#), [7-13](#), [16](#), [17](#), [19-22](#), [26](#), [28-30](#), [32](#), [42](#), [64-66](#)
- DHMM** Discrete Hidden Markov Model. [15](#)
- EAT** Electronic Assistive Technology. [8](#)
- ECS** Environmental Control Systems. [8](#)
- ED-EMA** Three Dimensional Electromagnetic Articulograph. [17](#)
- HMM** Hidden Markov Models. [8](#), [10](#), [11](#)
- HTK** Hidden Markov Model Toolkit. [8](#)
- IPA** International Phonetic Alphabet. [27](#)
- MND** Motor Neuron Disease. [10](#)
- OOV** Out-Of-Vocabulary. [25](#), [26](#)
- PAL** Personal Adaptive Listeners. [9](#)
- PBW** Phonetically Balanced Words. [18](#)
- PCC** Percentage of Consonant Correct. [18](#)

**SGD** Speech Generating Devices. [8](#), [65](#)

**TBI** Traumatic Brain Injury. [2](#), [19](#)

**U-TAP** Test of Korean Articulation/Phoneme. [18](#)

**VIVOCA** Voice-Input Voice-Output Communication Aids. [10](#), [11](#)

**VOCA** Voice Output Communication Aids. [10](#)

**WER** Word Error Rate. [20](#)-[22](#), [37](#), [41](#), [46](#)-[49](#), [54](#), [55](#), [58](#), [59](#), [62](#), [65](#)

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Dysarthria and applications of speech technology to dysarthria</b>	<b>5</b>
2.1	What is dysarthria?	5
2.2	Applications of speech technologies for dysarthria	7
2.2.1	Electronic Assistive Technology (EAT) and Environmental Control Systems (ECS)	8
2.2.2	Speech-Generating Devices (SGD)	10
2.3	Summary	12
<b>3</b>	<b>Related work and commercial Automatic Speech Recognition (ASR) systems</b>	<b>13</b>
3.1	Existing dysarthric speech databases	13
3.1.1	The Whitaker Database of dysarthric (cerebral palsy) speech (1993)	13
3.1.2	The Nemours database of dysarthric speech (1996)	14
3.1.3	Dysarthric Speech Database for Universal Access Research (UASpeech) (2008)	16
3.1.4	The TORGO database of acoustic and articulatory speech from speakers with dysarthria (2012)	17
3.1.5	Dysarthric Speech Database for Development of QoLT Software Technology (2012)	18
3.2	Evaluation of ASR systems	19
3.3	Currently available commercial ASR systems	22
3.3.1	Microsoft Azure Speech to Text	22
3.3.2	IMB Watson Speech to Text	24



3.3.3	Google Cloud Speech-to-Text	26
3.3.4	Amazon Web Services - Transcribe	27
3.3.5	Elhuyar Aditu	28
3.4	Summary	30
<b>4</b>	<b>Experiments</b>	<b>31</b>
4.1	Database prototype	31
4.1.1	Prompting items	31
4.1.2	Description of participants	32
4.1.3	Recording process	34
4.2	Azure Custom Speech to Text	35
4.2.1	Training	36
4.2.2	Testing	37
4.2.2.1	Finite State Grammars	38
4.3	Human evaluation	38
4.4	Implementation	39
4.5	Summary	40
<b>5</b>	<b>Results</b>	<b>42</b>
5.1	Azure Custom Speech to Text	42
5.1.1	Recognized commands	42
5.1.2	WER	46
5.1.3	Confidence	49
5.2	Human evaluation	54
5.3	Most difficult and easiest words	55
5.4	Implementation app	58
5.5	Questionnaire for participants	60
5.6	Summary	62
<b>6</b>	<b>Discussion and conclusions</b>	<b>64</b>
	<b>Bibliography</b>	<b>67</b>
	<b>A AZURE CUSTOM SPEECH TO TEXT MANUAL</b>	<b>71</b>

# Chapter 1

## Introduction

**ASR** systems are widely known nowadays. It is a technology that enables the recognition of spoken language and converts it into text. **ASR** can be seen as an alternative to the use of a keyboard, the control of mobile phones or the manipulation of other devices that everyone uses on a daily basis, such as light switches. One common implementation of **ASR** systems can be found in environmental control systems. These refer to systems in charge of controlling one's home appliances, for example by switching on and off the lights or television, opening and closing doors or pulling up and down the shutters. An example of this could be 'Amazon Alexa' or 'Google Home', as they are capable of controlling several smart devices by means of voice commands. Another common example of **ASR** implementation are virtual assistants such as 'Siri' and others. These virtual assistants are able to do tasks such as voice interaction, checking basic information, setting reminders, or streaming music. Naturally, all these systems are very convenient for all of us, as they make our lives easier. A group that can especially benefit from this sort of voice-controlled technologies are people with physical impairments or reduced mobility. Taking into account that the aforementioned systems replace the use of keyboard, people with functional diversity would be able to control their personal devices or home appliances with voice commands. This would be very favorable for them as they would not need to make movements that might be difficult for them or they would not need the help of others, therefore being more independent. However, many people with disabling conditions also present a disordered speech associated with neuromotor conditions. This obstructs the accurate recognition of their speech for **ASR** systems, which are usually not trained to recognize non-typical voices. Such is the case of dysarthria, a nervous system disorder

causing weakness in muscles used for speech ([Mayo Clinic \(2021\)](#)).

Dysarthria is one of the most recurrent disordered voices. Unlike typical voices, dysarthric speech tends to be slurred and slow, and people usually struggle to understand it. Dysarthric speech has a high degree of variability, not only between different people, but even within a single person. Also, there are different kinds of dysarthria depending on which area of the neuromotor system is damaged. Conditions that may lead to dysarthria are [Amyotrophic Lateral Sclerosis \(ALS\)](#), cerebral palsy, Parkinson's disease or [Traumatic Brain Injury \(TBI\)](#), among others. Due to communication problems derived from dysarthria, people suffering it may face social difficulties ([Mayo Clinic \(2021\)](#)). Dysarthric [ASR](#) lags years behind the progress that has been achieved for [ASR](#) for typical voices, that is, voices that do not present pathologies. The previously mentioned systems' accuracy drops when exposed to dysarthric speech, and people presenting it cannot benefit from them.

In order to develop an [ASR](#) system, there is a need to have big amounts of speech data. The systems are trained with large databases of typical speech samples, but not with samples of non-typical speech. Although there are some databases in English dysarthric speech, there are currently none in many other languages, such as Spanish. Moreover, the existing databases have not been collected for the purpose of training [ASR](#) systems; most of them have impairment severity assessment goals. Besides, even if databases like these existed, each person with dysarthria presents a great variation of personal features in their speech. These features may also vary depending on the energy levels of the speaker or the emotions they are expressing.

Broadly speaking, [ASR](#) systems can have two main functioning modes when referring to speaker dependency: speaker-independent systems and speaker-dependent systems. Speaker-independent systems do not require any training or adaptation to a particular user, as they have already been trained with speech samples from several subjects with no speech impairments. Commercially available [ASR](#) systems are speaker-independent, and therefore any user with a typical speech can use them. One option to improve the recognition accuracy of dysarthric speech with these systems could be the use of a dysarthric speech database to train them. However, as it has been mentioned, there is a great scarcity of data when it comes to this matter.

On the other hand, speaker-dependent systems require training with samples of the future user's speech. The acoustic model is aligned to the speech of the user and gets

a higher accuracy than speaker-independent systems for this specific user. The main drawback of these systems is that a big amount of training data is required. Although being the best solution to recognize dysarthric speech, this training requirements can be very demanding for people with speech impairments.

As Calvo et al. (2020) claim, speaker-adaptive ASR can be a promising approach to solve this problem. In this case, a small amount of data is collected from the user in order to adapt a speaker-independent model. Some commercial ASR systems offer the chance to adapt their base models to the user. This strategy is originally thought to be used with difficult accents or domain specific vocabulary, and usually consists of training a base model with the user's data or reducing the recognition scope. Such is the case of the commercial ASR system selected for this study: Microsoft Azure Speech to Text. This system offers their users a customization interface which can be used to train the language, acoustic and pronunciation models with the users' data. We thought that it could be possible to train this system with speech samples obtained from people with dysarthria.

Some investigation on dysarthric ASR is currently being undertaken. However, most of this investigation focuses on English dysarthric speech, and therefore cannot be extrapolated to other languages. Previous studies have determined the accuracy of commercial ASR systems with dysarthric speech; however, as far as we are concerned, none of them has used the customization interface of such systems to adapt them to specific users. Besides, no previous study has focused on Spanish dysarthric speech. There is currently no commercial option for dysarthric ASR. We felt the urge to experiment with a commercial speaker-adaptive system with Spanish dysarthric speech and see whether the obtained results are close to the ones obtained with a person with typical speech. Besides, we felt that there is a need for dysarthric speech databases, especially in other languages that are not English. By performing this study, we contributed to the visibility of people with dysarthria. We raised awareness of the need to create dysarthric speech databases and ASR systems that are able to understand them.

The main goal of this study is to evaluate the capacity a commercial ASR system to be personally and individually adapted to people with dysarthria. In order to do so, we will follow the next steps:

- First we will conduct an investigation of the current state of the art. This consists on listing the available dysarthric speech databases, accounting the

commercial **ASR** systems and whether they provide an adaptation interface, and reviewing the existing investigation on dysarthric **ASR**.

- Then, we will create a pilot database based on the needs we think should be met. In this case, it will consist of commands in Spanish for text creation and edition programs. This database will be recorded by 3 speakers with dysarthria and by a person with typical speech to be used as reference for the results.
- Lastly, we will train and evaluate the commercial **ASR** system with our database. Results of the speakers with dysarthria will be compared with those obtained by the reference speaker.

Based on previous results obtained by different studies, the following hypotheses were formed:

- The systems will not be able to adapt to dysarthric speech in their default mode, as it happens with typical speech. The models will have to be personalized individually for each speaker.
- Results obtained will be very varied, as each person's dysarthric speech is different. Results will therefore vary depending on the speaker and the type and grade of dysarthria they present.
- The chosen **ASR** system will not be able to fully adapt to dysarthric speech and therefore get the accuracy needed in order to work properly. This conclusion is based on accuracy results obtained by previous studies, which are very low. Although having a customization interface, the expected improvement to be obtained does not seem to be sufficient for **ASR** systems to work properly.

The next chapter will explain dysarthric speech in more detail, and will overview **ASR** services that are available for dysarthric speech. Chapter 3 will introduce the currently existing most well-known databases of dysarthric speech, the evaluation of **ASR** systems that have been conducted, and commercial **ASR** systems. The experiments conducted in this study will be explained in chapter 4, and the results obtained will be presented in chapter 5. Finally, the conclusions and discussion of this thesis will be compiled in chapter 6.

## Chapter 2

# Dysarthria and applications of speech technology to dysarthria

This chapter is devoted to explain the different types of dysarthria and their characteristics, based on the most commonly used classification. It will also overview current applications of speech technologies for dysarthria. It is worth highlighting that the 3 participants of our experiments were different in age and had different levels and types of dysarthria. Although the optimal solution would be to create speaker-dependent speech recognition systems, another option would be to create dysarthric speech databases with homogeneous groups of people.

### 2.1 What is dysarthria?

Dysarthria is a speech disorder caused by disturbances in neuromuscular control of the speech mechanism and resulting from impairment of any of the basic motor processes involved in speech production (Darley et al. (1975)). These disturbances lead to a weakness in the speech articulatory muscles. Although dysarthria does not affect the creation of grammatically correct sentences, it leads to a more imprecise articulation of words. Different types of dysarthria can be categorized depending on the affected area of the neuromotor system. Symptoms may also vary depending on the type of dysarthria. The most commonly used classification of types of dysarthria is the one developed at the Mayo Clinic (Darley et al. (1969)). This study collected thirty-second speech samples from 212 patients of seven different neurologic groups. Each patient had been unequivocally diagnosed as being representative of that diagnostic group. Three judges independently rated the samples on each of 38 dimensions

of speech and voice using a seven-point scale of severity. Based on the results, they identified six different types of dysarthria:

**Flaccid dysarthria:** it is related with lower motor neuron impairment. Characteristics of flaccid dysarthria may vary depending on the nerves and muscles affected and the reduced muscle tone (Enderby (2013)). The main symptoms are hypernasality, imprecise consonants, breathy voice, reduced phonation time, monopitch and monoloudness, poor intelligibility, and abnormalities in the tongue and lips when doing movements and at rest.

**Spastic dysarthria:** it is related with damaged upper motor neurons linked to the motor areas of the cerebral cortex. If damage is bilateral, the tongue and mouth will be completely impaired of movement; however, if the damage is unilateral, it will produce weakness of one side of the mouth and tongue. These lesions produce negative symptoms (losses of function) and positive symptoms (evidences of overactivity). Symptoms include a loss of voluntary movements, difficulty to swallow, imprecise consonants, monopitch and monoloudness, slow rate with short phrases and hyper nasality (although in a lesser degree than in flaccid dysarthria).

**Ataxic dysarthria:** it is primarily caused by cerebellar dysfunction. It is characterized by what is known as “scanning” speech (Charcot 1877), where each syllable is pronounced slowly and there is a pause after one or some syllables. The speech rate is slow and the consonants are also imprecise.

**Hyperkinetic dysarthria:** it is caused by disorders of the extrapyramidal system, which results in an increase of the movement. This type of dysarthria is often related with movement disorders such as dystonia or chorea. Therefore, hyperkinetic dysarthria can be slow or quick depending on the disorder it is associated with. Overall, the symptoms result from intermittent or continuous involuntary movements. There are imprecise consonants, distorted vowels, variable rate, prolonged intervals and phonemes and some inappropriate silences.

**Hypokinetic dysarthria:** it is caused by disorders of the extrapyramidal system, which results in a reduction of the movement. Spontaneous movements are poor and there may be abnormal involuntary movements. The speech rate is quite variable, there are monopitch and monoloudness and imprecise consonants. Both breathy and harsh voices can be present.

**Mixed dysarthria:** it is related to damage in more than one area, and thus has speech characteristics from two or more groups of dysarthria. It is common in both upper and lower motor neuron degeneration. The signs vary depending on the location and proportion of damage.

Although this is the most accepted classification of types of dysarthria, it should be kept in mind that dysarthric speech may also vary depending not only on the type of dysarthria, but also on the person. Two people with the same kind of dysarthria may present different features and difficulties in their speech. Besides, the time of the day and thus, the energy that the person has in that moment, also affects the final speech outcome.

Dysarthria may be caused by many different reasons and it can be congenital or acquired. It is congenital when the person is born with it or when brain damage happens during birth; this is the case of cerebral palsy, for example. These cases are often stable or chronic in their status. On the other hand, acquired dysarthria happens as a result of brain damage later in life, such as a stroke, brain tumor, Parkinson's disease or **ALS**, among others.

Dysarthria leads to several communication problems, and relationships with other people may be affected due to this reduction in communication skills. All these issues may result in social difficulty or depression (*Dysarthric speech: Symptoms and causes* (2020)). Speech and language therapy might be helpful. The quality and naturalness of speech may be improved in cases of mild dysarthria. In the case of severe dysarthria, therapy would be focused on improving intelligibility and communication competence; this might include alternative methods of communication (i.e. use of a speaking device) (Enderby (2013)).

## 2.2 Applications of speech technologies for dysarthria

Most commercial **ASR** systems do not meet the requirements to be used by people with dysarthria, as they still need to improve their accuracy considerably. However, there are already some services that have been specifically designed for people with



this condition. These services can be categorized into two different groups: **Environmental Control Systems (ECS)** and **Speech Generating Devices (SGD)**. This section will review some of the most popular services that have been created so far.

### 2.2.1 Electronic Assistive Technology (EAT) and Environmental Control Systems (ECS)

Many of the commercial **Electronic Assistive Technology (EAT)** systems and **ECS** have been designed to be used by people with typical voices; however, people who could benefit mostly from them are people with dysarthria. According to **Hawley et al. (2003)**, these systems should be speaker dependent, therefore trained with the user's utterances. They presented the STARDUST project, an approach that consisted of two parts: first, to develop a computerized training package and, second, to develop a speech recognition system with greater variability of speech utterances than common **ASR** systems. The training package would assist dysarthric speakers to improve the recognition likelihood and consistency of their vocalizations. The tolerance towards variability would be achieved by using the large corpus of data collected in the training phase. To set up the program, the recognizer had to be trained with some examples of words. The software would then determine a "best fit" utterance for each word, which would then become the target; the person would try to produce as close an articulatory approximation as possible. The "best-fit" is not necessarily the most "intelligible" word, but the one that best approximates the person's most likely production. In training mode, a word is displayed, and the user can play the "best fit", can speak the word or can move to the next vocabulary list. The training phase has three aims: to make each utterance as close as possible to the target, to imitate a stable target, and, since each utterance is recorded, to create a large corpus of data.

Eight adults were recruited for the STARDUST **ECS** trial (**Parker et al. (2006)**), five men and three women. These volunteers did not have a rapidly changing neurodegenerative condition. The command function words were determined depending on the device the users wanted to operate via the speech controlled **ECS**. Each vocabulary set consisted of ten words, which were recorded thirty times to build the first recognizer. Speaker dependent recognizers were trained for each individual with the **Hidden Markov Model Toolkit (HTK)** using continuous **Hidden Markov Models**

---

(HMM). Seven people completed the self-directed training phase. The pathology, severity or type of dysarthria did not seem to influence the final results. According to Parker et al. (2006), “speaker dependent recognizers illustrate that there are acoustically stable elements within the output of severe dysarthric speech”.

The *homeService* project (Christensen et al. 2013), aims to assist people with severe dysarthria by implementing a cloud-based environmental control system. According to the authors, “the only way to acquire substantial amounts of data is from a system which is being actively used”. This is due to the fact that prolonged speaking for people with dysarthria is usually exhausting; generally speaking, passive data collection from dysarthric speakers is limited. The *homeService* system has two main components: the *atHome* part and the *atLab* part. The *atHome* part is the collection of components implemented in the users’ home: a laptop to maintain the communication between the *atLab* and the *atHome*, a microphone for data capture, an infrared transmitter as remote control of the devices and an android tablet as a personalized, visual interface for the user. On the other hand, the *atLab* is a server at the university and comprises the main server which operates the ASR. It runs remotely “in the cloud”.

The cloud-based ASR servers allow researchers free access to maintain and update ASR models without having to modify the equipment in the users’ home. The ASR will be connected to the *homeService* users’ home by a broadband link. Each user has a dedicated ASR, previously loaded with personal acoustic and language models and grammars. The grammars are intended to restrict the vocabulary according to the given state the system is in. *homeService* systems are perceived as the first generation of Personal Adaptive Listeners (PAL).

Details about the system installation and how words are chosen are described by (Christensen et al. 2015). First, the command words are defined depending on the devices the user wants to control. Each command word should be recorded 5-10 times. The initial baseline models are trained in the lab, using the same data as training and testing. A grammatical hierarchy is also established, so that the system does not get confused with the commands and provides better results. Online and offline experiments are also carried out; online experiments could involve changing the acoustic models, and offline experiments could involve changing the training scenarios with the

data. There are two different data collection strategies: the enrolment data, obtained when the user reads the commands given, and the interaction data, the data recorded as the user interacts with the system once it has been adapted. The utterances of the interaction data are not inherently known; therefore, the recordings are saved and manually annotated afterwards. This system is focused on the “virtuous circle”. First, an “operating point” is set, a simple task that can provide good performance and that is useful for the participants. Users pronunciation consistency is improved by practice, which also leads to more data for **ASR** training. Therefore, the more the system is used, the better it gets and the bigger the corpus gets.

All interactions with the devices were recorded, which led to the creation of the *homeService* Corpus (Nicolao et al. 2016). As previously mentioned, most existing dysarthric speech databases were collected in controlled conditions. However, this is the first “spontaneous” dysarthric speech database, recorded in a real world environment. Besides, it contains examples recorded over several months, which enables the study of voice variations over time due to degenerateness.

## 2.2.2 Speech-Generating Devices (SGD)

There are two existing types of speech-generating devices or systems. The **Voice Output Communication Aids (VOCA)** usually take input from the user by means of a keyboard or switch-based interface and produce spoken output using synthesized or pre-recorded speech (Creer et al. (2013)). On the other hand, the voice-input voice-output communication aids **Voice-Input Voice-Output Communication Aids (VIVOCA)** take users’ voice as input and produce synthesized or pre-recorded speech as output. These systems usually output pre-recorded speech when the users have been able to record it themselves; this would be the case when the users have acquired conditions such as **Motor Neuron Disease (MND)**. However, when users have congenital conditions such as cerebral palsy, pre-recording with their typical voice is impossible; therefore, in these cases, synthesized voices are more widely used. In the study conducted by Creer et al. (2013), they produced synthetic voices for three male speakers with mild to severe dysarthria. Speech samples from the speakers were used to create synthetic voices that were close to the original one. As all three speakers had congenital conditions, a “restorative” process was needed for the samples. The technique followed was a **HMM**-based synthesis with data selection and imposition of

information from the average voice model. A qualitative evaluation was also carried out, which showed that one of the three speakers was pleased with the resulting personalized synthetic voice, as it conveyed many of his characteristics.

Hawley et al. (2012), developed a VIVOCA to recognize dysarthric speech and deliver the same message in synthesized speech. The system works as follows: users speak to a microphone, speech is processed and recognized words and phrases are passed to a message building module, and once the message is complete, it is passed to a speech synthesizer. In this study, HMMs were also used. In order to evaluate the system, researchers defined a vocabulary of input for each of the participants, which were then recorded around 20 times. The “best attempt so far” was determined by the system, and users aimed to replicate it. Once the system was trained, a series of tests were conducted. Nine people with moderate to severe dysarthria took part in the project, although a total of five completed the evaluation. The recognition accuracy during the training phase was 99%. When applied in real usage situations, the accuracy dropped up to 67%. Overall, most of the participants felt that the prototype should be improved, but that it could be a useful aid to their communication.

Bunnell et al. (2010) also proposed a procedure, named the ModelTalker project, to “bank” speech recordings from people with progressive condition. These recordings are meant to be used to create personalized synthetic voices. Another similar project is proposed by the Aholab group (Erro et al. (2015)); however, in this case, anyone can donate their voice. This way, both people with progressive conditions and people with congenital conditions can have a personalized voice.

Voiceitt (Voiceitt (2020)), is an ASR technology that translates non-standard speech patterns into clear speech in real time. The system is not language-dependent, unlike standard speech recognition systems, but speaker-dependent; instead of adapting the system to a new language, it is adapted to each individual’s communication. Its pattern classification technology is personalized for each speaker. The input used by this app is the users’ voice itself, and the output is a donor voice. Before using Voiceitt, a collection of known utterances is necessary. The user needs to provide a sample of five words, each repeated twice, so that the system learns how the user pronounces these words; as the user keeps using the app, more words are learnt from the

recordings. Then, clustering techniques are applied to the user’s phonetic inventory, so as to identify units of sound in the user’s speech (Murero et al. (2020)).

## 2.3 Summary

Dysarthria is a speech disorder characterized by a low articulation of words and phonemes. It is caused by a congenital or acquired disorder in the neuromuscular system. This leads to a weakness in the speech articulatory muscles, which makes it less consistent in comparison with typical speech. Depending on the affected area, dysarthric speech can be categorized into different kinds. However, it also varies depending on the person and other factors such as fatigue. Some speech features can be improved in dysarthric speakers by means of therapy.

The existing ASR systems designed for dysarthric speech include environmental control services and speech-generating devices. These systems should be speaker dependent. In all cases, the more the systems are used, the better they get. When training these systems, the “best-fit” is the closest one to the person’s most likely production, not the most “intelligible” one. Users are expected to replicate these “best-fit” words as best as possible; however, this is difficult for people with dysarthria, as they do not have a consistent word articulation. It is important that these systems are trained in a real world environment and over the time, so as to study the voice variations.

# Chapter 3

## Related work and commercial ASR systems

This chapter will introduce the most well-known databases of dysarthric speech, ordered chronologically. It will then review some evaluations of ASR systems that have been conducted with people with dysarthria. Finally, it will outline some of the most commonly used commercial ASR systems. The one chosen for this study, Microsoft Azure Speech to Text, will also be included in this section.

### 3.1 Existing dysarthric speech databases

There is a wide range of databases composed by typical voices; however, the publicly available databases for dysarthric speech are very limited. This section includes the biggest and most well-known databases for dysarthric speech.

#### 3.1.1 The Whitaker Database of dysarthric (cerebral palsy) speech (1993)

The Whitaker database (Deller Jr et al. (1993)) is presumably the first database of dysarthric speech. It got its name because it was collected during a Whitaker Foundation sponsored study into speech recognition technologies for persons with physical challenges. It was recorded in the Department of Speech and Language Pathology and Audiology at Northeastern University in Boston. The database is composed by a collection of 19275 isolated-word utterances. These recordings were spoken by six male persons aged between 28 and 48 years old with dysarthria due to cerebral palsy. The participants represented a wide spectrum of articulatory ability.

Additionally, a speaker with no voice disorders also took part in the study, and his data were used to provide a control group.

The vocabulary included in the Whitaker database has two parts: the TI-46 word list and the *Grandfather* word list. The TI-46 word list comprises utterances of the 26 letters of the alphabet, 10 digits (zero to nine) and the 10 “control” words (start, stop, yes, no, go, help, erase, rubout, repeat, and enter). The TI-46 word list had already been used in testing speech recognition algorithms, and was suggested as a standard by the Texas Instruments Corporation. On the other hand, the *Grandfather* is a list of 35 words selected from a passage frequently used by speech pathologists. It is called *Grandfather* because it begins with the sentence “Let me tell you about my grandfather...”. The words selected have an inherent phonetic diversity. Each word of the lists was uttered at least 30 times by each of the dysarthric speakers, although in most cases an additional block of 15 repetitions was also included. For the control data, there are 15 repetitions for the words in both lists.

All recording sessions were conducted in an isolated booth. Utterances were recorded on TDK type II cassettes using a TEAC W-450R stereo cassette deck with Dolby-C noise reduction. The data were filtered using an active band-pass, fourth-order Butterworth filter with a low-pass cutoff frequency of 4.7 kHz and a high-pass cutoff of 75 Hz. The sampling rate on the data was 10 kHz. Then, the endpoints of each utterance were “manually” located by means of a waveform editing program.

The aim of this database was to create a standard collection of speech which may be used to develop recognition technology for dysarthric speech (Deller Jr et al. (1993)).

### 3.1.2 The Nemours database of dysarthric speech (1996)

The Nemours database (Menendez-Pidal et al. (1996)) was designed to test the intelligibility of dysarthric speech before and after enhancement by various signal processing methods. The participants were 11 male speakers with varying degrees of dysarthria. They also had a non-dysarthric speaker record all the corpus as a control. The database was composed by 814 short nonsense sentences (74 sentences spoken by each participant). Each subject also recorded two paragraphs of connected speech, taken from the *Grandfather* passage and the *Rainbow* passage<sup>1</sup>. The form of the

---

<sup>1</sup>The *Grandfather* and the *Rainbow* passages are texts usually employed by speech therapists

nonsense sentences was “The X is Y-ing the Z”. X and Z were 74 randomly selected monosyllabic nouns; Y, on the other hand, was a set of 37 disyllabic verbs. This created 37 sentences, then, X and Z were swapped, thus creating another set of 37 sentences. The X, Z and Y words were thought to prove closed-set phonetic contrasts within an associated set of four to six words.

Participants were assessed according to the Frenchay Dysarthria Assessment (En-derby (1980)), which measures 28 relevant perceptual dimensions of speech grouped into 8 categories, namely reflex, respiration, lips, jaw, soft palate, laryngeal, tongue, and intelligibility. This led to a categorization of dysarthric speakers from “no abnormalities” to “severely distorted”.

The recording sessions were done by using a table mounted microphone connected to a digital audio tape recorder, and they consisted of three parts:

1. first, a speech pathologist conducted an initial assessment of the participant
2. then, the participant would record the 74 nonsense sentences
3. lastly, the participant would record the two speech passages.

In the nonsense sentences recording process, the experimenter read each sentence aloud first, and then the participant would repeat it. This assisted those subjects with limited eyesight or literacy. On average, each recording session would last between two and a half and three hours, including time for breaks.

Once the recordings were complete, the speech materials were digitized from the audio playback of the DAT recording using a 16 kHz sampling rate at 16-bit sample resolution with appropriate low pass filtering.

Additional perception experiments were conducted using the speech samples. For this, a minimum of 5 listeners heard each sentence 12 times; they had to identify the target words in the nonsense sentences.

The database was labeled at word and phoneme level. The word-level labels were assigned manually, whereas the phoneme-level ones were assigned using a Discrete Hidden Markov Model (DHMM) labeler. After doing so, the phoneme-level labels were inspected and corrected manually. They decided to use a DHMM for the labeling because it is appropriate for the recognition of a limited training size database. The main system was composed of a spectral analysis front-end, a vector quantizer system, and a DHMM Viterbi algorithm used to align the phonetic labels and to train the



39 phoneme models. In this paper, they tried several variations in the mentioned components so as to find the best possible combination to get the best accuracy. They believed that these techniques would be useful in the design of **ASR** systems for people with speech disorders.

### 3.1.3 Dysarthric Speech Database for Universal Access Research (UASpeech) (2008)

The UASpeech database (Kim et al. (2008)) includes speech produced by 19 speakers with cerebral palsy. Three subjects were recruited in Madison, though a collaboration with the Trace Research and Development Center, University of Wisconsin-Madison. The rest of the participants were recruited through personal contact established with clients of the Rehabilitation Education Center at the University of Illinois at Urbana-Champaign. Apart from audio data, video data was also recorded. The UASpeech database wanted to see whether an **ASR** system with both video and audio information would perform better than an audio-only **ASR** and, if so, whether the degree of benefit varied depending on the speaker.

In order to record the audio data, 8 microphones were arranged, 6 mm in diameter each, and with 1.5 inches of spacing between them. A Canon ZR500 was used to record the video data. The speech materials were displayed on a laptop. The AudioDesk (audio recording software of MOTU) was also run on the same laptop. The database was constructed with a variety of word categories. In the recording sessions, subjects would read three blocks of words. Each block contained 255 words, 155 of which were repeated across blocks and 100 uncommon words that differed across the blocks. The 155 words included 10 digits, 26 radio alphabet letters, 19 computer commands, and 100 common words. The uncommon words were selected from children’s novels digitized by Project Gutenberg, using a greedy algorithm that maximized token counts of infrequent biphones.

An intelligibility assessment was conducted using the recordings of the subjects. The aim was to observe whether different **ASR** architectures worked best for speakers with different categories of intelligibility. 225 words from the second block were chosen and randomly ordered. Five listeners were instructed to provide orthographic transcriptions of each word, together with a number (0 to 2) to indicate the degree of certainty. Based on the averaged percent accuracy, each speaker was classified in to one of four categories of intelligibility: very low (0-25%), low (26-50%), mid (51-75%)

and high (76-100%). The UASpeech database does not have any matching typical voice control subjects.

### 3.1.4 The TORGO database of acoustic and articulatory speech from speakers with dysarthria (2012)

This database (Rudzicz et al. (2012)) is the result of a collaboration between the departments of Computer Science and Speech-Language Pathology at the University of Toronto and the Holland-Bloorview Kids Rehab hospital in Toronto. To date, the TORGO database is the only publicly available one that combines the acoustics and endogenous articulation (e.g., tongue movement) of dysarthric speech. In the study, seven dysarthric speakers (4 male and 3 female) resulting from cerebral palsy were assessed. One subject with diagnosed ALS was recruited. They were matched with non-dysarthric speakers according to their gender and age (between 16 and 50 years old). Each experimental subject's motor functions were assessed according to the Frenchay Dysarthria Assessment (Enderby (1980)) (explained in section 3.1.2). The TORGO database contains approximately 23 hours of English speech samples, documentation and transcripts. Different types of stimuli were included in the data: non-words, short words, restricted sentences and unrestricted sentences.

Instrumental studies were also conducted through recording sessions. The first one involved the use of Three Dimensional Electromagnetic Articulograph (ED-EMA), which allows 3D recordings of articulatory movements inside and outside the vocal tract; this is achieved through sensors placed on the relevant locations of the speech articulators. The second one involved video recordings of facial markers; phosphorescent markers were placed on relevant landmark positions of the face, and using specialized software, their positions over time were extracted.

The acoustic data was recorded simultaneously by two microphones, one recording at 44.1 kHz and placed facing the participant at a distance of 61 cm, and a head-mounted electret microphone recording audio at 16 kHz. Both signals were temporally aligned using simple cross-correlation.

This database was conceived as a resource for developing ASR models oriented to people with dysarthria.

### 3.1.5 Dysarthric Speech Database for Development of QoLT Software Technology (2012)

This is a database (Choi et al. (2012)) designed as part of a national program to help better life for Korean people. There was a plan to create a database of about 600 middle or moderately severe dysarthric persons in 4 years, which goes as follows:

- in the first year, the creation of a database of 120 dysarthric persons of low or mild degree of disability and 40 typical speakers.
- in the second year, about a total of 106 persons. The idea was that after assessing the work of the first year, the degrees and areas of disability and ranges of age of the speakers would be readjusted.
- in the third and fourth years, the speech of another 200 persons would be added.

Before carrying out the final recordings, a small-sized prototype was created to be used as a pilot for discussing details about the final database. This prototype was the recording of 97 words uttered twice by 13 speakers with brain disability (12 males and 1 female). There were 61 words from Assessment of Phonology and Articulation for Children (APAC) and Test of Korean Articulation/Phoneme (U-TAP) and 36 Korean phonetic alphabets, or code words used for identifying Korean alphabet letters in voice communication. They were recorded with a notebook computer and a Plantronics Audio 750 DSP Stereo Headset.

The prompting items for the database recordings were composed by APAC 37 words for assessment of degree of dysarthria, 100 machine control commands and 36 Korean phonetic alphabets, 452 Phonetically Balanced Words (PBW), and 100 words and 5 sentences for investigating Korean consonants and vowels reflecting phonetic environment. The APAC words were later used to assess the speakers' severity of dysarthria. A speech therapist distinguished 4 groups according to Percentage of Consonant Correct (PCC): mild (PCC: 85-100%), mild to moderate (PCC: 65-84.9%), moderate to severe (PCC: 50-64.9%), and severe (PCC: less than 50%).

At the time of publication, the database was composed by recordings of 120 dysarthric speakers with cerebral palsy. They were 30 to 40 years old and were recruited from Seoul National Cerebral Palsy Public Welfare. The male and female speakers rate was 2:1. According to the PCC, the severity of dysarthria was mild for

65 speakers, mild to moderate for 23, moderate to severe for 8 and severe for 4 speakers. The speech data of the other 20 speakers were rechecked to find the intra-rater and inter-rater reliability. All recordings took place in a quiet office with a Shure SM12A microphone. Additionally, 40 typical speakers were also recorded as control group.

Although providing a plan for increasing the database in 4 years time, no recent papers regarding this dataset have been found. However, it is still the only dysarthric speech database in a non-English language.

## 3.2 Evaluation of ASR systems

Hux et al. (2000), present an early evaluation of 3 different ASR systems (Microsoft Dictation, Dragon NaturallySpeaking 3.0, and VoicePad Platinum) when used by a speaker with dysarthria. The main purpose of the study was to compare the accuracy and improvement pattern of these systems when presented with the speech of a TBI survivor with mild dysarthria. Microsoft Dictation and Dragon NaturallySpeaking 3.0 are used for continuous speech presentation, and VoicePad Platinum for discrete words. Another purpose was to compare the scores obtained from a speaker with and without dysarthria. There were two participants in this study: an 18-year-old female with ataxic dysarthria (due to TBI) and a 28-year-old female with no speech impairments. Both participants performed five data collection sessions for each of the three ASR systems. Each collection included the recitation of 10 read sentences and the generation of 10 novel sentences using one pre-selected word per sentence. Results of this study showed that the recognition accuracy of Dragon NaturallySpeaking was significantly better and more consistent for both speakers. However, it is worth highlighting that the results of this study are only based on one speaker with and one speaker without dysarthria. Each dysarthric speaker may display different speech characteristics; thus, the results obtained in this study cannot be generalized.

Ballati et al. (2018a) compared the extent to which three virtual assistants could be used by people with dysarthria: Siri, Google Assistant and Amazon Alexa. They relied on the dysarthric speech comprehension and the consistency of the answers from the assistants. In order to do so, they chose 17 sentences from the TORGO database that looked similar to the commands used to control the assistants. The

**Word Error Rate (WER)** was calculated for Siri and Google Assistant, as they provide the transcription of the request; however, Alexa only gives a binary indication about whether or not the sentence was understood. Therefore, Alexa could only be qualitatively compared to the other two assistants. Siri was the only assistant that tried to understand all sentences by providing a transcription for all of them. The consistency and appropriateness of the answers was similar for all three assistants. This study was conducted with a very low number of speech samples, since there are not many suitable sentences for virtual assistants in the TORGO database. On the other hand, it also lacked a control group, and results could not be compared with results obtained from non-dysarthric speakers.

A very similar study to the previous one, and conducted by the same researchers (**Ballati et al. (2018b)**), investigated the extent to which people with dysarthria could use smartphone-based assistants. In this case, they focussed on Italian dysarthric speech, and the chosen assistants were Siri, Google Assistant and Cortana. As there was no dysarthric speech database in Italian, they recorded 34 sentences from 8 patients (4 males and 4 females) with **ALS** at the Otolaryngology department of the Molinette hospital, in Turin, Italy. The speakers had different types of dysarthria: flaccid, spastic and unilateral upper motor neuron. The sentences were chosen among recommended questions from Amazon Echo and Google Home, and were modified to include all the phonemes of the Italian language. Results showed that the transcription was quite better in the case of Google Assistant, although the accuracy in transcription for all assistants was strictly dependent on the user. Besides, the answers provided by Google Assistant and Siri were more useful as they suited the question; in the case of Cortana, it usually provided the default answer. Similarly to the previously mentioned study, this investigation was conducted without a control group, and therefore these results cannot be compared to results obtained from speakers without speech impairments.

A study conducted by **De Russis & Corno (2019)**, compared the accuracy results obtained by IBM Watson Speech-to-Text, Google Cloud Speech and Microsoft Azure Bing Speech. The aim was to see whether **ASR** platforms were suitable for dysarthric speech recognition, analyze recurrent transcription errors and see if transcription alternatives could be used to improve the recognition result. The **ASR** systems were fed

with sentences chosen from the TORGO database. The **WER** was computed between the original sentence and the transcribed sentence. In terms of **WER** for dysarthric speech, Google Cloud Speech obtained the best performance. In some cases, the transcription alternatives were better than the actual transcription; therefore, the performance could be highly improved by choosing the right sentence among all the possible answers. However, this improvement was dependent on the knowledge of the “original sentence”, which is usually not the case in **ASR** systems.

The aim of the study conducted by **Calvo et al. (2020)** was to create a user-friendly speaker-dependent ASR system that could be used anywhere without the need of large amounts of data. A platform called mPASS was created as a collection of software tools enabling its users to create their own customized ASR system. It is an online-based application, where each user can train an acoustic model and create vocabulary. 20 subjects without dysarthria and 15 with dysarthria took part in the study. This system was tested in Italian and Polish, and it was used to recognize single discrete words and sentences. All participants recorded and tested 24 words and 5 sentences for 5 times in a clinical environment using the mPASS app. The recorded sentences were tested in the mPASS app and in Pocketsphinx, an open access commercially available speaker-independent ASR software. Results showed that the recognition accuracy rate for words was higher for both groups in the speaker-dependent customized system. The mPASS word recognition rate for the dysarthric speech group was 88.6%, while the sentence recognition accuracy rate was 98.5%. However, as it is highlighted in the study, these results might not indicate a real accuracy rate for sentences, as the target stimuli was very small (5 sentences), and usually the recognition accuracy rate decreases as the vocabulary increases. These accuracy results cannot be generalized. Moreover, the recordings were conducted in a clinical environment, which means that these results might not be replicated in noisier environments. Overall, enrolled subjects found mPASS useful and easy to use. **Calvo et al. (2020)** argue that, being a speaker-dependent software, intelligibility might not be crucial in determining accuracy as long as word articulation is consistent. Therefore, the greater the variability, the poorer the performance will be. Future studies need to investigate the correlation between accuracy and dysarthria severity.

### 3.3 Currently available commercial ASR systems

There is a wide range of commercially available **ASR** systems. These services leverage machine intelligence to combine grammar, language structure and the composition of audio and voice signals. These systems become better as they receive more input. Most of the existing cloud services support various input and output features. This section will review some of the most widely known **ASR** cloud services and comment on the customization options they might offer to their users.

#### 3.3.1 Microsoft Azure Speech to Text

Microsoft Azure (*Microsoft Azure* (2020)) supports 39 languages and 82 dialects for the voice service. Besides, each language has one or more base models, which are updated every 3-6 months in order to improve their accuracy. The input audios for recognition must be in 16kHz or 8kHz, 16-bit and mono sound. Supported input formats are WAV, PCM, MP3, OPUS/OGG, PLAC, ALAW in wav container, MULAW in wav container and ANY (for the scenario where the media format is not known). In addition to the standard Speech service model, users can also create custom models in the “Speech Studio” section of the system. Custom models are created by adapting a chosen base model to a particular customer scenario. In order to decide whether it is necessary to train a custom model, first, test data should be uploaded, in form of audio files. The system inspects the speech recognition quality of the test data. The Custom Speech portal provides a **WER**; the user can decide whether additional training is necessary based on this result. In case of being satisfied with the outcome, it is possible to start using the Speech service APIs directly. When not satisfied, the accuracy can be improved by a relative average of 5% to 20%; in order to do so, users should use the “Training” tab in the portal to upload additional training data. This additional data can be composed by audio files and their corresponding human-labeled transcripts, related text, or both. Besides, the related text can be of two types: sentences, or pronunciations. Sentences may include domain-specific vocabulary or product names, among others. Pronunciations include acronyms or words with undefined pronunciations. The audios and human-labeled transcripts must be uploaded together, so that the system establishes a correlation between each audio file and its corresponding transcription. The related text is used specially when training the domain-specific language, such as medicine or engineering, although it can be

used anytime. After doing so, another test should be conducted, to see whether the accuracy has been improved. This step can be repeated as needed.

The training stage of Microsoft Azure is meant to improve different accuracy issues, and each of them determines which dataset should be used to improve the accuracy. The documentation on custom models includes Table 3.1, which reports this information:

Use case	Data type
Improve recognition accuracy on industry-specific vocabulary and grammar, such as medical terminology or IT jargon.	Related text (sentences/utterances)
Define the phonetic and displayed form of a word or term that has nonstandard pronunciation, such as product names or acronyms.	Related text (pronunciation).
Improve recognition accuracy on speaking styles, accents, or specific background noises.	Audio + human-labeled transcripts.

Table 3.1: Azure Custom Speech training use cases and data types

Once all the steps in the training are completed, a custom endpoint can be deployed to be used with apps, tools and products. The model lifecycle could be summarized in three different steps:

- **Adaptation:** taking a base model and customizing it to your domain/scenario using text data and/or audio data.
- **Decoding:** using a model and performing speech recognition (decoding audio into text).
- **Endpoint:** a user-specific deployment of either a base model or a custom model that is only accessible by a given user.

As mentioned before, the base models used to create custom models are updated periodically. This might cause issues with compatibility, therefore, it is important to bear in mind that both base and custom models have an expiration timeline. For base models, the adaptation is available for 1 year (after import, it is available for



1 year to create custom models), and the decoding and endpoints are available for 2 years after import. For custom models, the decoding and the endpoints are available for 2 years; after this time, the custom model is not lost, but it should be retrained on the renewed model.

Apart from creating custom models, Microsoft Azure also provides the user with the chance of using batch transcription, a set of REST API operations that enables the transcription of large amounts of audio in storage. Batch transcription can be used with one or more individual files and also with whole storage containers. Both base and custom models are accepted. One transcription result file is created for each audio input. The batch pipeline also supports diarization, that is, the capability of recognizing two speakers. Since Azure Speech to Text has been the chosen speech recognition system to perform our experiments, a more detailed explanation on the training and testing phases will be provided in chapter [4](#).

### 3.3.2 IBM Watson Speech to Text

The IBM Watson Speech to Text service ([High \(2012\)](#)) offers three interfaces: a Web-Socket interface, a synchronous HTTP interface, and an asynchronous HTTP interface. The service also provides APIs that can be used for voice-automated chatbots, analytic tools for customer-service call centers or multi-media transcription among others. For most languages, the sampling rate should be 16 kHz for broadband and 8 kHz for narrowband.

The service supports many languages and dialects (11 languages and 13 dialects) and provides a customization interface, which can be used to augment speech recognition capabilities. It needs to be taken into account that this interface is not currently available for all languages. It supports both custom language models and custom acoustic models.

This system offers many different parameters that can be used to tailor the request. Some of these parameters are interdependent, or can only be used with the customized interface. For example, the parameter “keyword” is used to introduce an array of keyword strings that the service spots in the input audio. However, this parameter does not return anything unless the parameter “keyword\_threshold” is established, which indicates the minimum threshold for a positive keyword match. On the other hand, the parameter “customization\_weight”, for example, can only be used when

using custom language models, as it indicates the relative weight that the service gives to words from a custom language model versus words from the base vocabulary.

By means of the language model customization, the user can expand and tailor the vocabulary of a base model to include domain-specific terminology. The service's base vocabulary contains everyday words, but may lack knowledge of specific terms. Once the language model is created, corpora and domain-specific words can be added to it; after training the system with these new data, the interface will improve its accuracy in speech recognition for the chosen domain.

In the case of the acoustic model customization, the interface can be adapted to different environments and speakers. The model should be trained with audio resources that closely match the acoustic characteristics of the audio file we want to transcribe. Speech recognition can be highly improved by means of acoustic customization in cases like having a unique acoustic channel environment, when the speaker's speech patterns are atypical or when the speaker has a very pronounced accent.

The system also enables the use of grammars with customized models. The system will only be able to recognize words, phrases and strings that are compiled in the grammar. The use of grammars boosts the accuracy of the system, as the space for valid matches is limited.

The length of time for training both the language and the acoustic models will depend on the data available for the model. It is possible to create up to 1024 custom language models and 1024 custom acoustic models per account. The custom language model can be populated with corpora, grammars or individual words, which are stored in the "words resource"; here, the pronunciation and spelling of these words are defined. The user can add a maximum of 90 thousand **Out-Of-Vocabulary (OOV)** words to the words resource of each custom model; a total of 10 million words are accepted per model (including the service's base vocabulary and **OOV** words). The system includes information about n-grams (it is not a word-matching algorithm); the service analyzes the contents of the corpus file, tokenizing and parsing the data, extracts **OOV** words and adds them to the words resource.

The data used for the custom acoustic model should be at least 10 minutes and maximum 200 hours long, and should only include speech, not silence. All audio and archive-type resources are limited to a maximum size of 100 MB. The sampling rate of the audio resources must match the sampling rate of the base model. The audio

content should reflect the acoustic channel conditions of the audio to be transcribed. Besides, acoustic model customization alone cannot produce **OOV** words; thus, a language model based on transcriptions of the audios might be needed. The service analyzes the data asynchronously and extracts information about it, such as sampling rate, length or encoding. The training will usually take the same amount of time as the length of the audio data.

### 3.3.3 Google Cloud Speech-to-Text

Google Cloud Speech-to-Text ([Google Cloud \(2020\)](#)) has three methods to perform speech recognition: Synchronous Recognition (limited to 1 minute audio length), Asynchronous Recognition (480 minutes), and Streaming Recognition (5 minutes). The system accepts audio sample rates from 8000 Hz to 48000 Hz, although the ideal sample would be that of 16000 Hz. Moreover, it also allows the user to specify the model to be used (“video”, “phone-call”, “command\_and\_search” or “default”), so that results can be improved as the audio is processed with with a model trained for that particular type of source. This system supports a wider variety of languages than Microsoft Azure and IBM Watson (71 languages and 135 accents in total).

Among the features it offers, there is the “speech adaptation” feature, which is helpful to improve the accuracy of words and phrases that occur frequently in the audio data, expand the base vocabulary (e.g. by adding proper names or domain-specific words), and improve the accuracy of speech transcription when the audio is not very clear. Words or multi-word phrases should be added to the “SpeechContext” in order to increase the probabilities of the system recognizing them. Classes such as monetary units and calendar dates can also be added to the “SpeechContext” by means of class tokens. This resource is limited to 5000 phrases per request, a total of 100,000 characters per request and 100 characters per phrase.

The model bias can be increased by using the “boost” feature, assigning more weight to some phrases than others. The higher the value, the higher the likelihood that phrase is chosen as a transcription alternative.

One big drawback of using Google Cloud Speech versus Microsoft Azure is the lack of acoustic model customization in the former. This is a pivotal feature in order to enhance the accuracy of **ASR** systems and adapting them to people with dysarthria.

### 3.3.4 Amazon Web Services - Transcribe

Amazon Transcribe ([Amazon Web \(2020\)](#)) is a speech to text system available in 20 languages and 31 dialects. It also has a streaming transcriber for certain languages. The sampling rate should be 16 kHz (Wideband) or less (Narrowband). Amazon lets users create a custom vocabulary to help the system recognize words that are not being recognized or unfamiliar words that are specific to a certain domain. A custom vocabulary is a list of words, phrases or proper nouns. Each user can have up to 100 vocabularies in their account, and each custom vocabulary is limited to 50 Kb. The custom vocabulary is specified in a text file; it can be either a list or a four-column table. For lists, each word or phrase can be placed on different lines or be separated by a comma. Each entry must contain fewer than 256 characters and only characters from the allowed character set. On the other hand, each row from a table contains a word or phrase, followed by the optional [International Phonetic Alphabet \(IPA\)](#)<sup>2</sup>(to specify the pronunciation, [IPA](#) characters can be included here), SoundsLike (to mimic the pronunciation of the word with standard orthography) and DisplayAs (defines how the word or phrase should look in the output) fields.

Amazon Web Services also allows the user to create Custom Language Models, to train and develop domain-specific language models. Text data should be uploaded to Amazon Simple Storage Service (Amazon S3), provide Amazon with permission to access the data and choose a base model. The text data can include domain-specific text or audio-scripts. The provided text data should be as close as possible to the audio the user will later transcribe; the quality of these data is more important than the quantity. There are two ways to upload the text data to create a custom language model:

1. Upload your text as training data. Use training data to train the custom language model for the specific use case.
2. Upload the domain-specific text as training data and the audio transcripts as tuning data. Use tuning data to optimize the custom language model and increase its transcription accuracy.

---

<sup>2</sup>The [IPA](#) is used to provide a unique symbol to each of the distinctive sounds in a language

The maximum amount of data provided should be 2 GB for training data and 200 MB of tuning data. Amazon Web Services also provides a table (3.2) with advice on how to upload the data depending on the specificities of it.

<b>If you have</b>	<b>Upload this</b>
A large amount of domain-specific text and a Domain-specific text as training data.	Upload your much smaller amount of audio transcript text data transcription text as tuning data.
A minimum of 10,000 words of audio transcript text.	Audio transcript text as training data.
At least 100,000 words of audio transcript text and a large amount of additional domain-specific text.	Audio transcript text as training data. Typically, this method leads to the greatest possible increase in transcription accuracy. Follow the first method described in this table if this method doesn't produce the desired increase in transcription accuracy.
Domain-specific text only.	Domain-specific text as training data. We recommend any of the preceding methods of uploading your data over this one.

Table 3.2: Amazon Web Services, how to upload data

As opposed to custom vocabularies, custom language models do not only recognize individual terms, but they also use each term's context to transcribe the audio. Additionally, they also add words to their recognition vocabularies, so that users do not need to add the words manually. When having enough text data that represents the audio to be transcribed, training custom language models can produce improvements in accuracy over using custom vocabularies. Besides, language models and custom vocabularies cannot be used together. However, this service is only available in US English for now, therefore, it could not be used to enhance this project's transcriptions in Spanish.

### 3.3.5 Elhuyar Aditu

Aditu is Elhuyar's ASR system (*Elhuyar Aditu* (2020)), based on artificial intelligence and neural networks. For now, the system is able to transcribe in Basque and Spanish, although they are planning to expand their language offer. It accepts most audio

formats, and also offers the chance of recording oneself through the browser by means of a microphone. Among other purposes, it is meant to be used to generate subtitles. Aditu can handle audio and video files and also links to different webpages. Once the file or link is transcribed, the system returns different files to the user: a text file with the whole transcription, a file with the generated subtitles (in both VTT and SRT formats), and a JSON file containing each word with its duration, confidence rate and start and finish time. If the provided transcription is not correct, users can correct it by means of the online edition interface in Aditu. This interface displays the subtitles segmented and users can modify each of them separately, delete them, split them, join them or add new segments with bits of speech that might not have been recognized by the system.

Another available product in Aditu is the “Aditu live API”, the live ASR system. This can be done in two different ways: for long continuous speech by using secure web socket connections or for short speech sections by using HTTP POST calls. The transcription is available in monolingual and bilingual mode; however, in order to work, at least one transcriber needs to be available per language. A JSON message is received at the beginning of the transcription and each time the number of available transcribers changes. Therefore, for monolingual transcription 1 transcriber needs to be available, whereas for bilingual transcription there is a need for 2 transcribers (1 per language). The server sends recognition result to the client using a JSON format. The results include the hypotheses, transcription, confidence, likelihood and language of the hypothesis (for bilingual). By default, the server assumes that incoming audio is sent using 16 kHz, mono, 16 bit little-endian format. This can be overridden using the 'Content-Type' header.

One big disadvantage of Aditu is that it does not have a personalization interface, neither for language models nor for acoustic models. Therefore, each transcription should be corrected manually, but cannot be improved by training the system. They claim that, under optimal conditions, the accuracy rate can be over 95%. In order to get the best accuracy as possible, users should use a good microphone, standard language, one speaker at a time, adequate volume and pace, formal register, correct phrases and no background noise.

### 3.4 Summary

To the best of our knowledge, there are not available databases of dysarthric speech in Spanish. Besides, the existing databases in English or other languages have been designed as part of diagnosis tests, they were not designed for ASR or virtual assistants; also, samples were collected in a professional setting. Final results obtained by evaluation studies are likely to get higher results than they should; there is much overlap, as the sentences used in training and testing are usually the same. On the other hand, it is worth highlighting that none of the databases mentioned above include any recordings from children.

The experiments mentioned in section 3.2 try different ASR systems in order to see their accuracy when used by people with dysarthria. Results varied depending on the speaker. The consistency of the answers by virtual assistants depend on whether they are able to understand the speaker. Some improvement was observed in the transcription options provided by some ASR systems, although this meant that the “original sentence” had to be known. Most of them do not train the systems before being used; since they are all speaker independent, they cannot adapt to dysarthric speech. In the case of people with dysarthria, it is important to have speaker-dependent models or to adapt speaker independent models with pre-collected data. The only mentioned study that trained a system with the users’ speech obtained very high accuracy rates. However, they are not likely to be replicated in non clinical environments.

The currently available dysarthric speech databases are not designed for ASR training purposes. As far as we are aware, Elhuyar and Google Cloud are currently the only commercial ASR systems that work with Basque. However, taking into account that the aim was to improve the accuracy of an ASR system and personalize it as much as possible, Microsoft Azure was thought to be the most adequate ASR commercial system. It provides a customization interface for language and acoustic models and the chance to use different base models for customization.

# Chapter 4

## Experiments

This chapter is devoted to explain the experiments conducted in this thesis. First, the prompting items will be introduced, followed by a brief description of the participants. It will later describe the different experiments carried out.

### 4.1 Database prototype

The design of a pilot database was one of the objectives of this work. One of the most important characteristics of our database is that it is in Spanish. There are currently no available databases of Spanish dysarthric speech. Another important feature is that this database was recorded by the participants themselves in their own houses. This is an important point since previous databases were recorded in studios instead of in a real life use environment. This implies that the results we obtained from the experiments represent everyday life conditions. 3 people with dysarthria and 1 person with typical speech constituted the database.

#### 4.1.1 Prompting items

The prompting items were chosen mainly based on the necessities of the participants. A preliminary idea was to integrate the customized models in programs participants use on a daily basis, such as text creation and edition programs, or in the personal assistant of their personal computers. The prompting items chosen for this end were commands to control the *Word* program. It was a collection of 21 different commands chosen from among the keyboard shortcuts for *Word*. The whole list of commands used in these experiments is the following:



abrir documento	cursiva	pegar
alineal	deshacer	reducir imagen
ampliar imagen	encender	rehacer
apagar	guardar	seleccionar
cerrar documento	imprimir	subrayar
copiar	justificar	tabular
cortar	negrita	tachar

The aforementioned integration is not possible as both *Word* and personal assistants have their own default speech recognizer and do not allow the use of any other recognizer. However, we decided to stick to these prompting items as they are standard commands in text creation and edition programs. Besides, this way this database could be used in future experiments of implementation of speaker-dependent recognizers in commercial programs.

#### 4.1.2 Description of participants

Enrolled subjects for this study were 1 adult man, 1 adult woman and one child with dysarthria. These individuals comprised a group of heterogeneous dysarthric speakers. Recruitment of participants was made through personal contact with them or with their tutors (in the case of underage participant). Several people get in touch with the Aholab Laboratory to learn about the choices they have for their particular cases. The three people involved in this project got in touch with us because they were interested in the issue this project investigates. They found it very difficult, if not impossible, to use commercial **ASR** systems, as these systems do not normally recognize their voices. They all wanted to know whether there was a way of taking advantage of these systems and use them to control their computers with their voices. Participants volunteered themselves to take part in this study.

These subjects communicate with family and people close to them with their natural speech. In spite of having dysarthria, their speech is intelligible for people who know them. However, this is not the case with **ASR** systems. Apart from the already mentioned subjects, a person with typical speech was included in the database. Her results were meant to show the accuracy that a person with typical speech can obtain and to be used as a reference. We will now introduce the most relevant information about the subjects who participated in this study:

C1 is a 13-year-old girl who has congenital cerebral palsy. She has reduced mobility, although she does not need a wheelchair to move. She has an adapted mouse and keyboard in her personal computer. Her speech rate is slower than typical speech rate. She sometimes presents difficulties pronouncing /r/ and /n/ at the end of words, for example in *apagar* or *imagen*. She sometimes needs to make pauses in between syllables so as to pronounce the right phoneme. Her speech is slow and her syllables are long, especially the vowel sounds. She has a breathy voice, and her volume is quite high. She does not seem to have difficulties to sustain any sounds, so her respiratory muscles are quite powerful.

M1 is a 51-year-old male who has congenital cerebral palsy. He has reduced mobility and uses an electric wheelchair. He has an adapted mouse in his personal computer. He presents difficulties pronouncing /r/ at the end of words. He also has difficulties with the double *r* sound, which is present in many Spanish words such as *cerrar* and in words beginning with *r*. His speech stops in between certain syllables, when the last sound of the first syllable and the first sound of the next syllable are consonants. This is the case of words like *imprimir*, *subrayar*, *justificar* or *guardar*. He also has difficulties with some consonant clusters that are common in the Spanish language like /gr/ as in *negrita* and /cc/ as in *seleccionar*. His speech is slurred and somewhat nasal.

W1 is a 59-year-old woman whose dysarthria is due to a brain anoxia. She has reduced mobility and uses a wheelchair to move. However, she does not use any special hardware in her personal computer. Her speech is shorter than usual typical speech; she presents difficulties to sustain sounds. In general, she is able to differentiate vowel sounds from one another. She presents difficulties when pronouncing /c/ in its strong form, as in *cortar*, *copiar* or *cursiva*. This also affects the outcome of the following vowel sound, therefore in this cases /o/ and /u/ are not very clearly pronounced. This is not the case when pronouncing /c/ in its weak form, as in *cerrar*. However, she struggles with the pronunciation of the plosive /p/ sound, which is often perceived as the weak /c/. Therefore, *pegar* and *cerrar* are often misunderstood. She struggles with the sound /m/ both when coming before and after a sound, as in *imprimir*, when the sound appears before and after the vowel /i/. Consonant clusters such as

/gr/ are also difficult for her, as in *negrita*.

R1 is a 26-year-old woman with typical speech, that is, a speech with no pathologies. Her models have been used to see how a typical speech would perform in the same circumstances as a dysarthric speech. We call her models R1, standing for REFERENCE, because the results obtained by this speaker are a reference of what Azure Speech to Text is capable of. Results obtained in the R1 models should be considered a goal for the rest of the models.

### 4.1.3 Recording process

Ethical approval was obtained from the CEISH (Comité de Ética para las Investigaciones con Seres Humanos) of the University of the Basque Country (UPV/EHU) to conduct these experiments. Enrolled subjects signed informed consent forms. We decided to use the Nannyrecord platform for the recordings, as we thought it would be the easiest recording tool to use for non-experienced users. In the case of W1, GarageBand was used as the recording tool. This was due to the fact that her personal computer used a Macintosh software, which is not compatible with Nannyrecord. We wanted the users to be as autonomous as possible during the recording sessions, as they would be conducted in their homes and whenever they wanted. All recordings were performed in a home environment, so that the system was trained in the real future environment. This also maximized their comfort, as subjects did not need to attend recording sessions in remote locations. The need for interaction with this recording tool is minimal. In some cases, subjects were assisted by their relatives or assistants during the recording sessions. In other cases, subjects were completely autonomous and were able to conduct the whole recording process by themselves. Besides, each individual decided the duration of each recording session. The recording process varied depending on the recording tool they used. Participants that used Nannyrecord recorded a command at a time, that is, a command was displayed on the screen and participants had to utter it and record only that command. When finished, they could move on to the next one. If they felt that the recording was not good enough, they could overwrite the audio file. When using GarageBand, W1 uttered the complete list of 21 commands in a single audio file. Then, we split the whole audio into the different commands. Therefore, C1 and M1 provided us with a total of 210 audio files, each file containing a command. W1 provided us with a total

of 10 audio files than we then split into the total of 210.

The recording hardware tools were provided to two of the subjects; this consisted of a headset. In the third case, the subject had a personal desktop microphone, which was used for the recordings. The aim was to obtain heterogeneous samples from the same person. In order to do so, enrolled subjects were asked to conduct their recording sessions in different days and at different times of the day. This way, the variability of the speech would be greater and the system would not be customized based on a single recording session. Hence, the recording sessions could be spanned across the period of days or weeks, and the recording sessions could have any duration. The average duration of each recording session was between 10 and 15 minutes for C1 and W1 and between 20 and 25 minutes for M1<sup>1</sup>. If the system is trained with little variability in the subject's performance, results are not reliable, as the system could become accustomed to the performance of the subject at the moment of recording. The opposite effect could also be achieved; the recording process is very tiring for people with speech impairments, and long recording sessions are not feasible for them. If appropriate duration of recording sessions is exceeded, utterances may be more unintelligible and therefore the accuracy results would decrease. On the other hand, the order of the commands that were to be recorded was also altered. This way, we avoided the predictability of the words that could happen if the order was always the same.

## 4.2 Azure Custom Speech to Text

This section will describe the training and testing processes in Azure Custom Speech to Text. A detailed description of how to do it is provided in the appendix [A](#). In order to create a customized acoustic language model, it is necessary to have an Azure account. When creating the new model, the user must provide a name for the project, a description of the project (this is optional) and choose the language of the model. The data for the custom model may include *Audio* data, *Audio + Human – labeled transcripts* and *Related text*. The *Audio* data is used to inspect the accuracy of speech recognition with regards to a model's performance; it is used to test the

---

<sup>1</sup>This information was gathered from a Questionnaire the participants answered to, which will be introduced in [5.5](#)

accuracy of Microsoft’s baseline or a custom model. The *Audio + Human – labeled transcript* data can be used for both training and testing. The human-labeled data is used to evaluate accuracy and train the model for future use cases. For optimal results, it is recommended to provide word-by-word transcriptions for about 10 to 20 hours of audio; however, this is not a realistic approach for dysarthric speech, as people with this condition get tired faster than typical speakers when talking. The audios and human-labeled transcriptions must be packaged as a single .zip file, which must be less than 2GB. *Related text* data for training helps ensure correct recognition. *Related text* can be of two types: sentences or pronunciations. Sentences may include domain-specific vocabulary or product names, among others. Pronunciations include acronyms or words with undefined pronunciations. For the related text, each term or sentence must be written in a single line, and a single .txt file must be uploaded, which should not exceed 200MB. Pronunciation files, on the other hand, cannot exceed 1MB .txt files.

Since pronunciation in dysarthric speech tends to be problematic, it is likely to think that the pronunciation could be customized for each speaker. However, it is not recommended to use custom pronunciation files to alter the pronunciation of common words. Apart from this, it is currently only available in English (en-US) and German (de-DE).

It is important to check whether the desired base model supports training audio data, as otherwise, it will be ignored. It is recommended to begin with the related text when training a model, as this will already improve the recognition of special terms and phrases.

Once the dataset is uploaded, users can navigate to the **Testing** tab and visually inspect the audio only or audio + human-labeled transcription data or navigate to the **Training** tab and train a custom model. Models can be trained with *Audio + Human – labeled transcript*, *Related text*, or both. A base model among four must be chosen to this end. Once the training has been completed, different tests can be run, both with the training data or with the testing data. When the model is ready, users must go to the **Implementation** tab and add a connection point.

### 4.2.1 Training

The interface offers 4 different base models to train a customized model: 20200115, 20200619, 20201015, and 20210113. The system was first trained once with each base

model. The training data was always the same: 9 sets of 21 commands with their human-labeled transcripts (189 commands and 189 transcriptions), and a related text containing the 21 commands. This way, we could observe which base model worked best with the data provided by the users. For all cases, the best results were obtained with the 20201015 base model. Once the base model had been chosen, we proceeded to train it based on the jackknife (or *leave one out*) method. This means that each time a model is trained, one of the training samples must be left out. We trained the system 10 times per speaker, omitting one session of audios on each occasion. Therefore, 189 commands and 189 human-labeled transcriptions and the related text were used for each training. The omitted session of audios was then used for the testing. Using this method, we could evaluate the system better, as we could calculate the average results based on 10 different tests. It should be highlighted that the training data did not include a finite state grammar; that is, a grammar limiting the possible recognizable words to those included in it. This was performed in the testing phase, and will be later explained in the following section [4.2.2.1](#).

## 4.2.2 Testing

The testing can be performed with both the training data or the testing data. If the data provided for the tests is only *Audio* data, the platform will provide a transcription of the audio files. If the data used in the tests is *Audio+Human-labeled transcript*, the platform also provides a [WER](#) and specifies the insertions, the substitutions and the deletions. This is done by comparing the model's transcription with the reference transcription. The tests can be run with a maximum of two models simultaneously, which can be base models or customized models. Testing can also be performed by means of a Python script. This must be the case when willing to include a finite state grammar, as it cannot be included in the training data before customization. A Python script must also be used when transcribing directly from the microphone. The Azure Custom Speech Studio interface only allows conducting testings on previously recorded data, which must be uploaded to the system. Testing was performed by using a set of 21 commands. As previously mentioned, the audio files used in the testing were those omitted in the training phase. So, if the system had been trained with sessions 1 to 9, the testing was performed with session 10. A total of 10 tests were performed for each speaker. No testing was conducted with input received through

the microphone. This was performed later by means of an online app, which will be introduced later in this chapter.

#### 4.2.2.1 Finite State Grammars

Azure’s Speech Studio offers its users the chance to conduct the experiments directly in the interface. However, if users want to use a finite state grammar, a script must be used in a development environment. For this, we used a Python script. A finite state grammar limits the recognition to the words included in it. The system is not able to recognize any words that are not included in it. Even if the audio file tested has an out-of-grammar word, the system will pick the closest one to it from the grammar, or will output a blank space instead. Our finite state grammar included all 21 commands again. By limiting the recognition options to these commands, the possibilities of each command to be correctly recognized rises. Different tests were conducted using the finite state grammar. In an aim to observe the weight of the grammar in a base model, we tested the 20201015 base model, with no training, using only grammar, with all the commands. This was done 1 time per speaker. Then, we tested the customized models using the jackknife method again, and including the grammar.

### 4.3 Human evaluation

In an aim to observe whether humans were able to understand our participants, we conducted a perceptual test <sup>2</sup>. Each person had to listen to 30 commands in total (10 commands from each participant), and had to transcribe what they understood. Besides, they also had to indicate the difficulty of understanding each of the 30 commands; for this, they had to indicate the effort they made to understand from ”Muy poco” (very little) to ”Mucho” (a lot). These were then converted into numerical values from 1 to 5, 1 being the easiest and 5 the most difficult.

---

<sup>2</sup>[https://aholab.ehu.eus/users/margot/test/Listening\\_test.php](https://aholab.ehu.eus/users/margot/test/Listening_test.php)

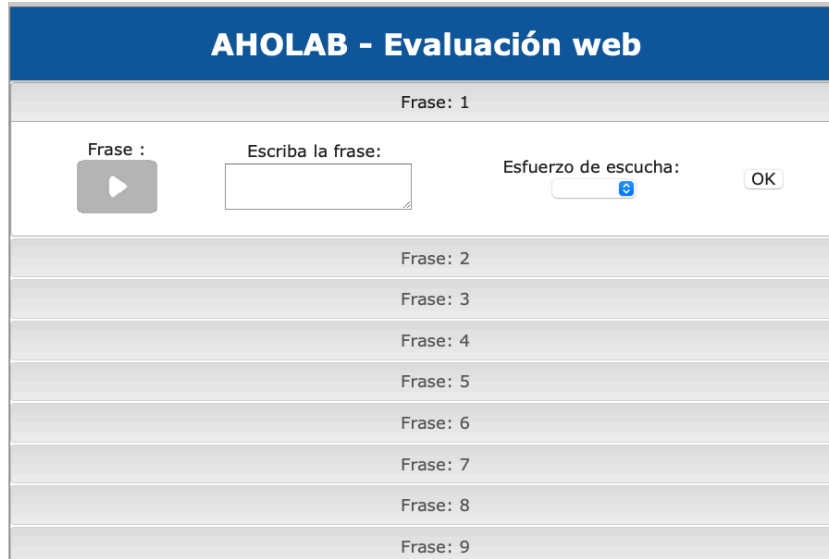


Figure 4.1: Perceptual test for human evaluation

Each command could only be listened once, so people had to be focused on the task on hand. When they did not understand something, they were asked to write three points (...). The commands for the task were chosen randomly from the whole database each time a person took the test. Before listening to the commands, people were asked about any hearing impairments they might have. They were also asked about any relationship with a person with dysarthria. We had previously theorized that people with experience in talking with people with dysarthria would be able to understand the commands better and therefore, get higher scores.

## 4.4 Implementation

As previously mentioned, a preliminary idea was to integrate the customized models in programs or in the personal assistant of the users' computers. These programs were meant to be those that the participants used on a daily basis, in this case, Microsoft's *Word*. However, *Word* and similar programs do not allow the implementation of external speech recognizers into them. The same happens with the default personal assistants of their laptops. Therefore, in an aim to make our findings more understandable for the users, we created an online app. Besides, all previous testings had been conducted with recorded audio files, but not with live microphone input. By means of this app, we would be able to observe the accuracy of Azure when used live.



Users introduce their own login data into the app; each user’s login data is linked to their customized speech model. The models used for this app were models trained with all the recordings of each speaker. These models had not been previously tested in Azure, as there was no left out session on this occasion. Once logged in, they can try the model with microphone input. The word or phrase the user utters is transcribed and displayed on the screen. Users will then tell the system whether the transcription is correct or not. When correct, the audio file and its correspondent transcription are saved into the Aholab database. When incorrect, users are asked to provide a correct transcription of what they said. Then, the audio file, Azure’s transcription and users’ corrected transcription are saved into the Aholab database. In future works, it would be ideal to be able to implement customized speech models into different programs so that users can control them through voice commands. Some text creating and processing programs already employ dictation systems. These systems transcribe everything the users say into the document. However, they have not been trained for people with dysarthria or other speech impairments. Besides, these systems do not allow full control of the programs with voice commands. This is a key feature that should be implemented, as many speech disorders are accompanied by a reduced mobility. Therefore, being able to control programs with voice commands and reducing the need to use a mouse would be of great advantage for these people.

## 4.5 Summary

The database used for our experiments is composed of a total of 630 commands produced by dysarthric speakers (210 per speaker) and 210 commands produced by a typical speaker. The prompting items are commands to control *Word*. All recording sessions were performed in the participants’ homes, at their pace.

All the models were trained using the 20201015 base model of Azure, since this was the model that gave us the best results. The training for each participant consisted of training the system on 10 different occasions, leaving one session of samples out on each occasion. For the training, a related text, audio files and their corresponding human-labeled transcriptions were used. The testing was also performed on 10 different occasions per speaker, using the left out samples. Besides, experiments using a finite state grammar were also performed, both with the 20201015 base model and the customized models of each speaker.

An implementation app was created in order to test the models with live microphone audio input. By means of this app, we also made our findings more understandable for the participants of the study, as they were able to test the trained models by themselves.

A perceptual test was conducted with people who were not involved in the project. This was made in order to see whether people obtained similar results to Azure. Due to the fact that speech recognition models cannot be implemented in commercial programs, we created an online app where the participants of the experiments could try their customized models. The models used in this app were trained with all the recordings from each speaker. Data obtained from this app was stored in the Aholab server to later calculate the **WER**.

# Chapter 5

## Results

This chapter will review the results obtained in the experiments explained in the previous section. We will also offer some theories on the reasons of these results. First, results obtained from the Azure Custom Speech to Text interface will be explained. Then, we will review the answers got in the human perceptual test. After this, the accuracy achieved in the implementation app will be analyzed. Finally, the answers of a questionnaire undertaken by the participants of the experiments will be summarized.

### 5.1 Azure Custom Speech to Text

#### 5.1.1 Recognized commands

**ASR** systems are not always able to understand and recognize every word humans say. Sometimes, they do recognize something, but it is not the right term. This section will look at the number of commands that have been correctly recognized, not recognized, and recognized with errors. As we will see, these results vary between people and speech recognition models. The graphics included here present the values for the Azure base model and the Azure custom model, both of them with and without finite state grammars. The values of the human evaluation are also included in the dysarthric speech models (C1,M1,W1), but not in the R1 models. There are three different values in the bars: correctly recognized, recognized with error(s) and not recognized. Correctly recognized commands are those that were perfectly recognized, with no errors. In this case, *abrir documento* should be recognized as *abrir documento* for example. Commands recognized with error(s) refer to commands that were either completely wrongly recognized or recognized with one or more errors. This is the case

for *abrir documento* recognized as *abrir un momento*, *cerrar* recognized as *pegar* or *rehacer* as *tienes sed*. Lastly, the not recognized commands are those that the system is not able to recognize and therefore outputs a blank space.

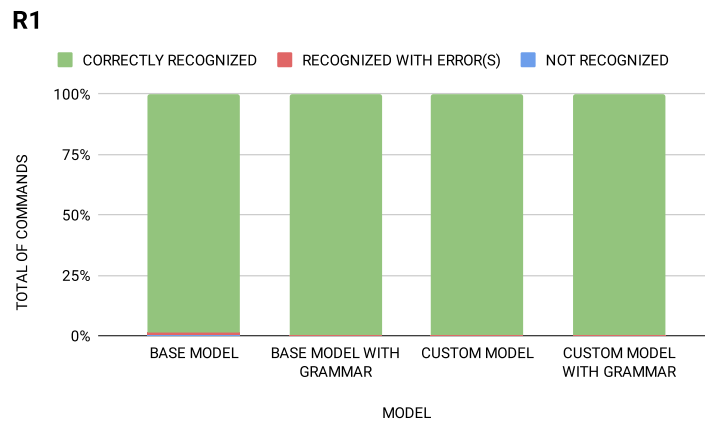


Figure 5.1: R1 recognized commands

Figure 5.1 shows the results obtained with the R1 models. As it can be observed, the base model failed in 3 commands of the total 210. The rest of the models performed perfectly, recognizing all of the commands. This means that the base model with grammar, the custom model, and the custom model with grammar improved by a relative average of 1,45% over the base model. As mentioned in 3.3.1, Azure states that the accuracy can be improved by a relative average of 5% to 20%. The relative improvement in this case is quite low because there was nearly no room for improvement. Besides, the base model’s performance was already very satisfactory. Graphics 5.2, 5.3, 5.4 and 5.5 show the results achieved with the 3 participants of the study. The fourth graphic represents the AVERAGE results obtained with C1, M1 and W1 (the R1 speaker is not included). This way, we can review which of the models works best for people with dysarthria.

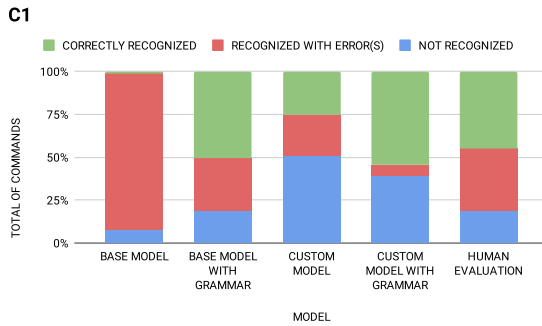


Figure 5.2: C1 recognized commands

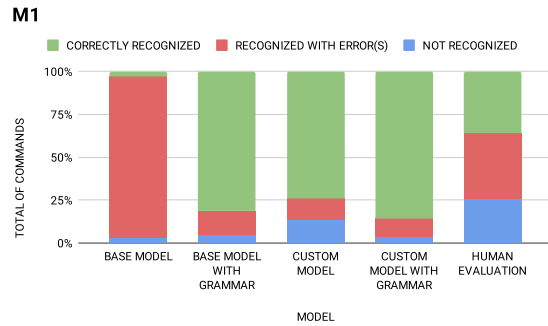


Figure 5.3: M1 recognized commands

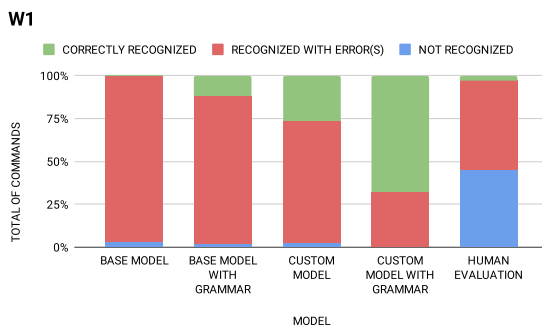


Figure 5.4: W1 recognized commands

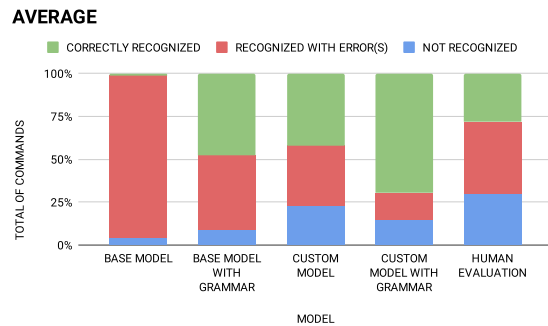


Figure 5.5: AVERAGE recognized commands

In the case of C1, the base model got many incorrectly recognized words. After the training, the amount of wrongly recognized words decreased considerably, while the correctly recognized words increased; however, the amount of not recognized words also incremented. The finite state grammar affects very positively the base model, and enables the correct recognition of a higher number of commands. Nonetheless, the best model was the custom model with grammar. This model obtained the highest number of correctly recognized commands and the lowest number of wrongly recognized commands, and improved by a relative average of 3700%.

For M1, the use of a finite state grammar in the base model was slightly better than the custom model. Customization in the case of M1 was very effective, as the

number of correctly recognized commands was much greater than in the base model. Besides, the custom model performs better than the human evaluation [\[1\]](#); this means that a trained base model is able to understand more commands than a human. The use of grammar made an impressive difference in the base model. The number of not recognized commands in the base model with grammar was a bit higher than in the base model, but the number of correctly recognized commands was much higher. Still, the best model was the custom model with grammar, with a relative improvement of 2900%.

The customization process for W1 was the less effective one. As the graphic shows, the customized model performed better than the base model with grammar, but there were still many wrongly recognized words. The number of not recognized commands was nearly the same for the base model, the base model with grammar and the custom model. Actually, W1 got the lowest number of not recognized commands among the 3 participants. The best result was obtained with the custom model with grammar; the base model did not recognize correctly any of the commands, while the custom model with grammar recognized 143 commands of 210.

Apart from the number of correctly recognized commands, the total of wrongly recognized commands and not recognized commands also varies between speakers. For example, the number of correctly recognized commands in the custom model was nearly the same for W1 and C1, but the not recognized commands were higher in the former.

Although the reasons for these results are not completely clear, a feature that may have influenced the outcome might be the speech rate of each speaker. C1's speech was slower than the rest, with a mean duration of 2,28 /pm 0,89. The R1 speech mean duration was 1,14 seconds /pm 0,29. It may be possible that the custom model of C1 expected the commands to be finished earlier, and stopped the recognition process too early. Besides, unlike the base model the custom model is influenced by the training data. These factors could have led to a higher number of not recognized words in the custom model. Nonetheless, C1 got the higher rate of correctly recognized words in the human evaluation probably thanks to the longer duration of the speech. The theories why M1 achieved the best results with the customized models, are that his

---

<sup>1</sup>The Human evaluation process will be introduced in section [5.2](#)

speech rate is relatively close to that of R1, being the mean duration 1,33 seconds /pm 1,28. Besides, his main pronunciation difficulties were consonant clusters and the need to make short pauses in between certain syllables. These mispronunciations might have been easier to get for the system than others. W1's speech is the shortest of all, with a mean duration of 1,06 /pm 0,35. It is worth highlighting that she was also the most difficult subject to understand for human evaluators. On the other hand, some of the commands in the custom model were confused with other commands. This might have been due to the fact that she has difficulties with some relevant consonants for this task and also with some vowels. Besides, a bit of echo can be perceived in her audio files, which might have affected the final outcome.

Overall, as we can observe in the AVERAGE graphic(5.5), the relative improvement of the base models thanks to the training is quite good. However, the highest score was obtained by training the base model and including a finite state grammar. It could be stated that, in order to get optimal results, Azure's recognizer should be trained and later tested including a finite state grammar. Nonetheless, if we compare the results obtained for C1, M1 and W1 with those of R1, we can observe that there is still much improvement to achieve.

### 5.1.2 WER

The previous section showed the number of correctly recognized commands, not recognized commands, and recognized commands with errors. This section will look at the total WER and at the sort of errors committed by the models. The WER represents the total number of wrongly recognized words. It is calculated as follows:

$$WER = (S + D + I)/N$$

$S$  is the number of substitutions,  $D$  is the number of deletions,  $I$  is the number of insertions,  $C$  is the number of correctly recognized words and  $N$  is the total number of words in the reference. The lower the value is, the better the output has been. The lowest value possible is 0. It should be highlighted that a very high WER might mean that there is an elevated number of insertions. The WER is composed by substitutions, deletions, and insertions. Substitutions occur when a term has been replaced by a different one. Deletions refer to those words that have been omitted and not replaced by another one. Insertions are words that have been added to the reference, they can be one or more.

Table 5.1 shows the WER committed by each of the models. The WER value for each of the models is the average of their 10 cross-validation tests.

MODEL	WER	STANDARD DEVIATION
R1 BASE MODEL	0,0120	0,018
R1 BASE MODEL WITH GRAMMAR	0	0
R1 CUSTOM MODEL	0	0
R1 CUSTOM MODEL WITH GRAMMAR	0	0
C1 BASE MODEL	2,132	0,157
C1 BASE MODEL WITH GRAMMAR	0,713	0,134
C1 CUSTOM MODEL	0,780	0,107
C1 CUSTOM MODEL WITH GRAMMAR	0,836	0,079
M1 BASE MODEL	1,884	0,270
M1 BASE MODEL WITH GRAMMAR	1,269	0,124
M1 CUSTOM MODEL	0,320	0,196
M1 CUSTOM MODEL WITH GRAMMAR	1,133	0,047
W1 BASE MODEL	1,256	0,183
W1 BASE MODEL WITH GRAMMAR	1,147	0,057
W1 CUSTOM MODEL	0,704	0,115
W1 CUSTOM MODEL WITH GRAMMAR	0,100	0,094

Table 5.1: WER value per model

The R1 base model has a WER of 0,0120, which was due to 3 wrongly recognized commands of all the 210. The rest of the R1 models have WER of 0, which is the lowest possible, and means that no errors were made. A key idea is that although the WER might be very high in some cases, it does not necessarily mean that the model it corresponds to is the worst. As previously mentioned, the WER is composed of different features. Two models might have the same number of correctly recognized commands, however the type of errors of each model might lead to different WER values. For example, the missed words produce substitution and deletion errors; these types of error will never be higher than the total number of words. However, substitution errors do not have a top limit. This means that, theoretically, if a model is really bad there could be much more words in the proposed model than in the reference, which would increase the total WER value. This idea will be better understood with the graphic 5.6, which shows the type of error per speaker in both their custom model and their base model. Each color corresponds to one type of error. It can be appreciated that the bars have different heights; this is because each



speaker has a different total number of errors. The sum of all the errors gives the **WER** per speaker and model. The high of the bar represents the **WER**. The **WER** contemplates the following:

- words that have been recognized with errors, such as recognizing *encender* as *enciendes* (represented by substitutions)
- words that have not been recognized and therefore omitted in the proposed model; this is the case when not recognizing a command or part of it, for example not recognizing *cortar* or recognizing *imagen* from *reducir imagen* (represented by deletion)
- words that have been invented and substitute the reference words, for example recognizing *cortar* as *compra* (represented by substitutions)
- extra words that are not part of the reference and therefore increase the total number of words, as in *copiar* recognized as *fue copiar* (represented by insertions)

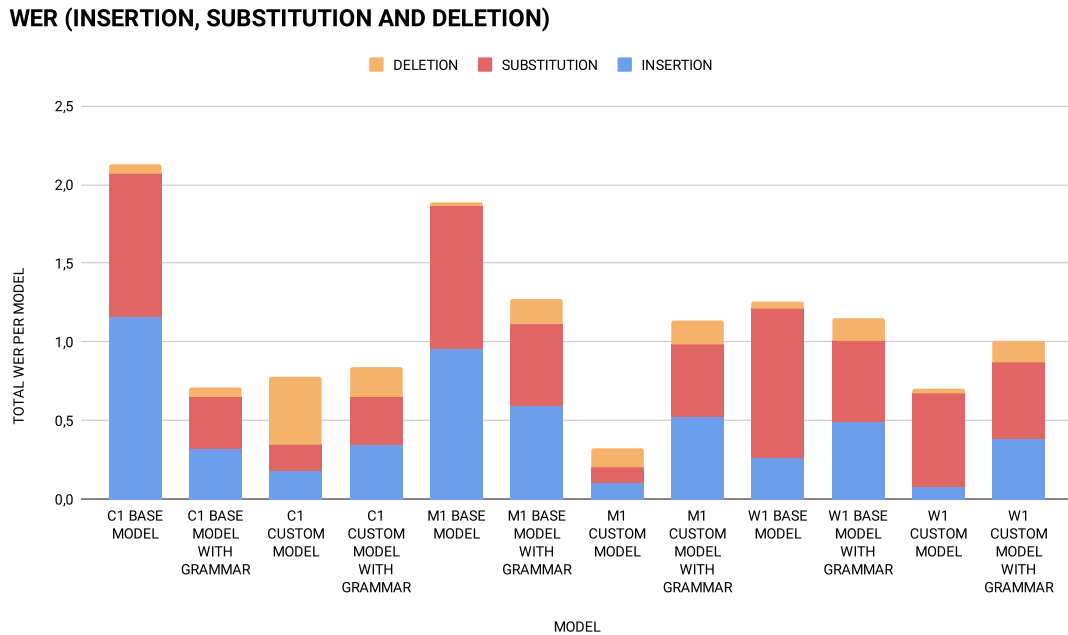


Figure 5.6: WER per speaker and model

As it can be observed in the graphic above, the total **WER** drops in dysarthric speech models after the customization. As seen in graphics [5.2](#), [5.3](#), [5.4](#) and [5.5](#), the base model with the worst results was that of W1, as it did not recognize a single command. However, we can observe that it is also the base model with the lowest **WER**. This is due to the types of error that conform the total **WER** value. Substitutions and insertions are the most common errors in the base models, while deletion is quite rare. However, in the case of C1 CUSTOM MODEL, we can see that deletion is the most common type of error. This means that although the system has been able to improve its substitution and insertion errors, there are still many words that it is not able to understand and therefore displayed a blank space in the recognition. On the contrary, we can see that W1 CUSTOM MODEL has nearly no deletions, thus the system is able to recognize most of the terms. Nonetheless, there is a great number of substitutions, which means that the system is recognizing many non existing incorrect words. After performing the training, we can see that the best custom model in terms of **WER** (without taking the R1 model into account) is M1. There are still errors in this model, but the total **WER** is the lowest of all three dysarthric speech custom models. In in graphics [5.2](#), [5.3](#), [5.4](#) and [5.5](#), we saw that the custom model with grammar got the highest number of well recognized commands for all speakers. In the case of figure [5.6](#), we can see that the custom models with grammar are not the models with the lowest **WER** value.

### 5.1.3 Confidence

Another interesting feature to bare in mind is the confidence level of the recognized commands. The confidence tells us how sure the system is about the recognized word(s). The confidence level goes from 0 to 1, being one the highest confidence level. We have calculated the confidence per word and represented the distribution of the confidence in histograms. In green, we have the confidence of the well recognized commands; in red, we have the confidence of the wrongly recognized commands.

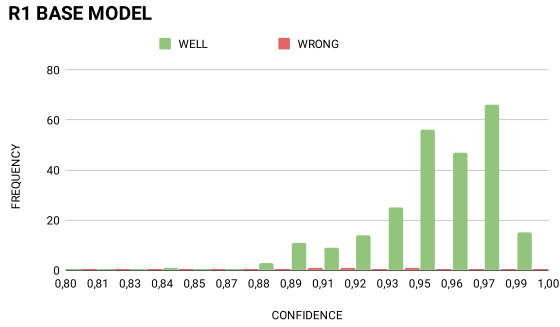


Figure 5.7: R1 BASE MODEL confidence

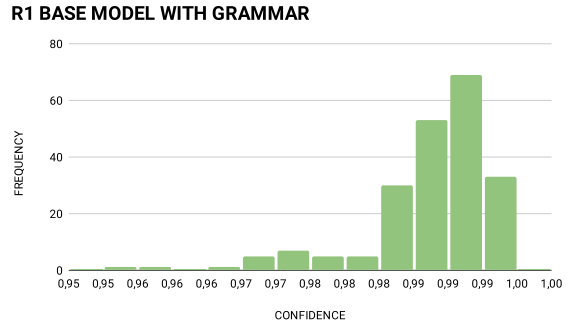


Figure 5.8: R1 BASE MODEL WITH GRAMMAR confidence

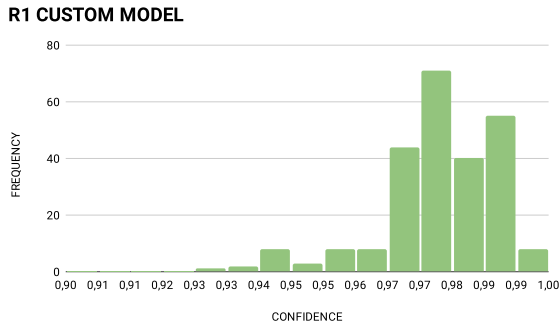


Figure 5.9: R1 CUSTOM MODEL confidence

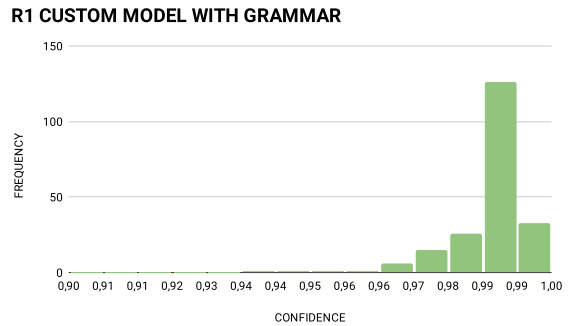


Figure 5.10: R1 CUSTOM MODEL WITH GRAMMAR confidence

In the case of R1 models, we can see that there are only 3 instances of wrongly recognized terms in the base model. The rest of the models performed perfectly, without errors. However, it is interesting to see that the distribution of the confidence is not the same for all models. In the R1 custom model with grammar, nearly all instances are concentrated in 0,99 of confidence. In the rest of the models, the confidence is more scattered, even though the confidence levels are very high.

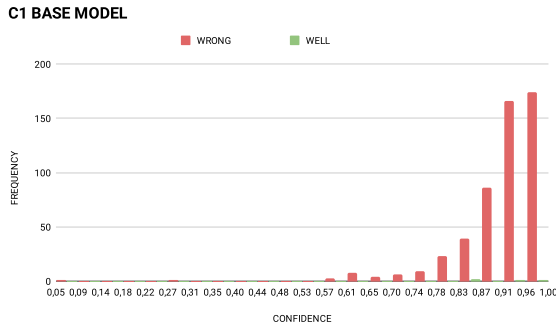


Figure 5.11: C1 BASE MODEL confidence

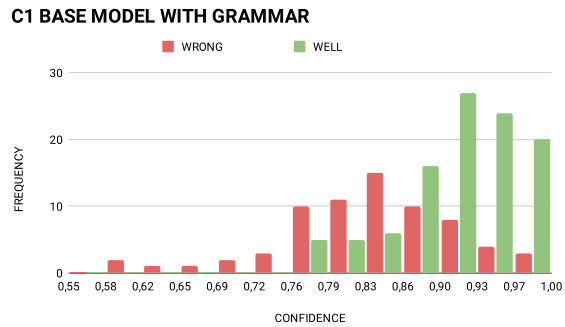


Figure 5.12: C1 BASE MODEL WITH GRAMMAR confidence



Figure 5.13: C1 CUSTOM MODEL confidence

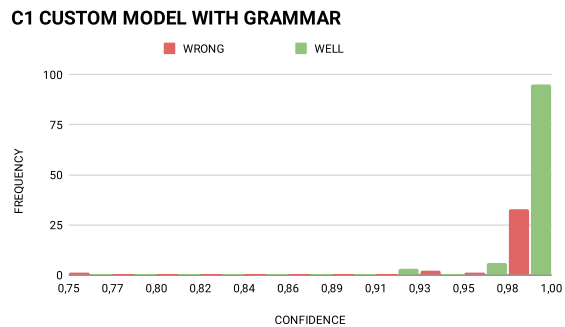


Figure 5.14: C1 CUSTOM MODEL WITH GRAMMAR confidence

In C1 BASE MODEL, the very few instances of well recognized commands are between 0,83 and 1. The wrongly recognized commands have very high levels of confidence. The clearest different between the instances of well recognized and the wrongly recognized commands for C1 is the CUSTOM MODEL WITH GRAMMAR. In this case, the confidence levels of both sorts of commands is very high; however, the number of well recognized commands is much higher than that of the wrongly recognized ones. In C1 BASE MODEL WITH GRAMMAR, we can see how the confidence is more spread between 0,58 and 1. It is worth highlighting that in C1 CUSTOM MODEL some words were correctly recognized with a very low confidence level, around 0,08.

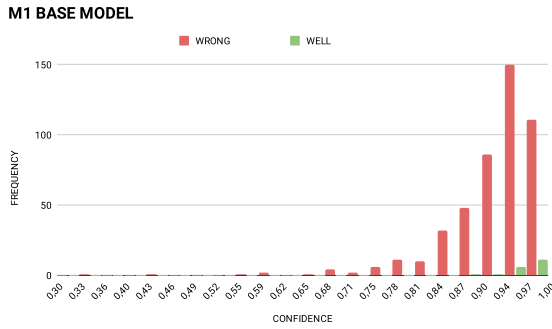


Figure 5.15: M1 BASE MODEL confidence

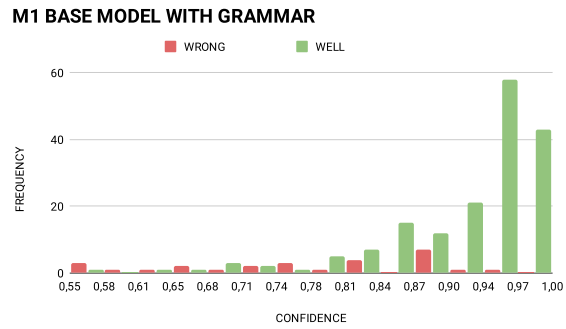


Figure 5.16: M1 BASE MODEL WITH GRAMMAR confidence

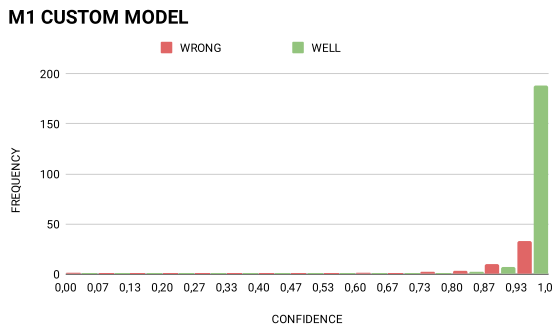


Figure 5.17: M1 CUSTOM MODEL confidence

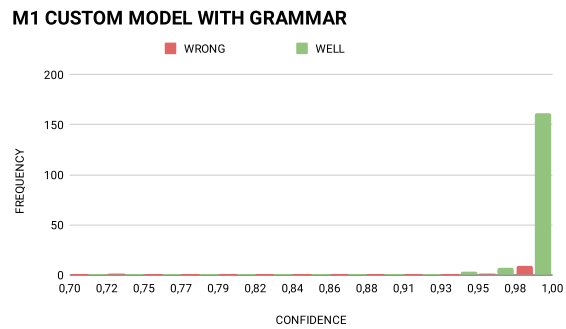


Figure 5.18: M1 CUSTOM MODEL WITH GRAMMAR confidence

A similar situation happens with M1. In the base model, all confidence levels are very high, the highest being those for well recognized commands. In M1 BASE MODEL WITH GRAMMAR, the confidence levels are more spread going from 0,55 to 1. M1 CUSTOM MODEL presents sporadic instances of wrong recognized words with very low confidence levels. M1 CUSTOM MODEL WITH GRAMMAR also presents this sort of instances, although in this case the confidence level is around 0,72. In both these models we can observe how the well recognized commands outnumber the wrongly recognized ones, although the confidence levels for both are very high.

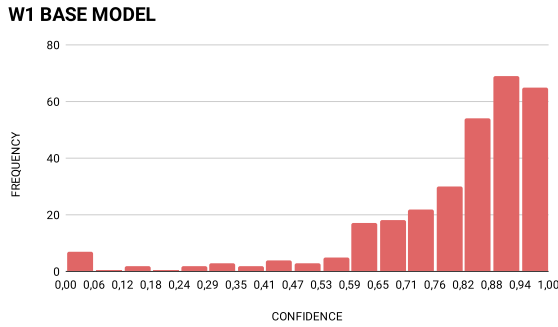


Figure 5.19: W1 BASE MODEL confidence

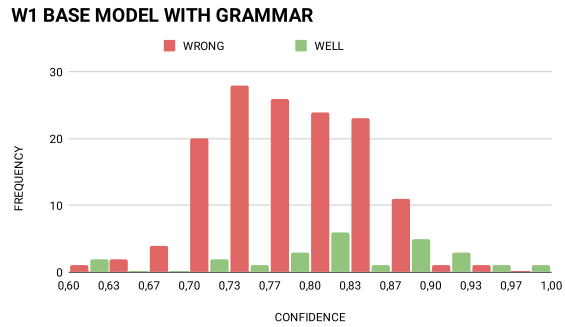


Figure 5.20: W1 BASE MODEL WITH GRAMMAR confidence

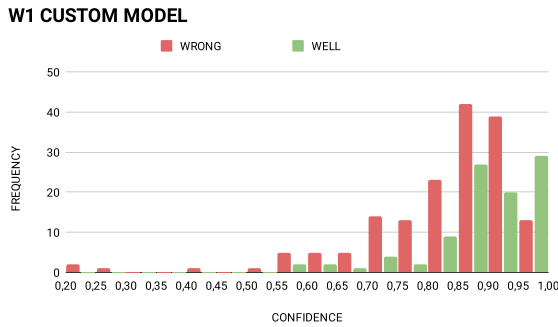


Figure 5.21: W1 CUSTOM MODEL confidence

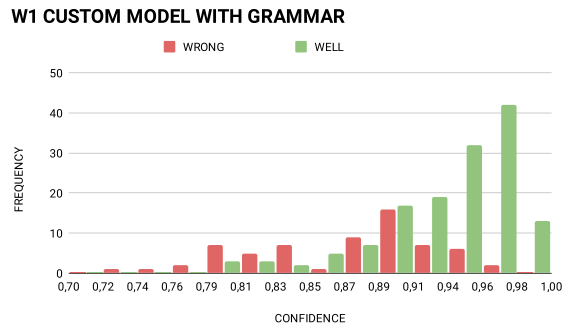


Figure 5.22: W1 CUSTOM MODEL WITH GRAMMAR confidence

W1 BASE MODEL did not recognize any commands, therefore, all confidence levels in the graphic belong to wrong recognized commands, which go from 0 to 1. Similarly, in W1 CUSTOM MODEL all values are spread between 0,20 and 1, although there are well recognized commands in this model. In W1 BASE MODEL WITH GRAMMAR, the confidence values are not that spread, but concentrate between 0,60 and 1. In this model, we can see how the highest confidence values correspond to well recognized commands. In the case of W1 CUSTOM MODEL WITH GRAMMAR, we can see how the confidence values have accumulated but not as much as in custom models with grammar for C1 and M1. Besides, we can see

that this speaker had more errors than the previous ones in the custom model with grammar.

So, we could conclude that overall the confidence levels are more scattered in the base model and the base model with grammar. When using the custom model or the custom model with grammar, we can see that the confidence levels rise and most instances accumulate in between 0,90 and 1. Although this is not always the rule, because we have seen how the confidence levels of W1 CUSTOM MODEL WITH GRAMMAR were more spaced than those of this same model for other speakers. We have also observed that in C1 CUSTOM MODEL, there were words recognized at 0,08 confidence. Ideally, well recognized words should get high levels of confidence while wrongly recognized words should get low levels of confidence. This is not the case in any of the presented models.

## 5.2 Human evaluation

A total of 85 people took part in the perceptual test. The aim of the human evaluation was to observe how difficult it is for humans to understand the commands uttered by the participants of the study. The audios employed in this perceptual test were the same audios used in the experiments. Apart from the **WER** value, there is also the "perceived difficulty" value. These scores are subjective, as people who did the perceptual test marked the commands they listened to based on their perception. Thus, this value is only included in the human evaluation, and not in the rest of the results. Table 5.2 shows the mean **WER** and perceived difficulty of the perceptual test per speaker and their corresponding standard deviations (S.D.).

	<b>WER</b>	<b>S.D.</b>	<b>PERCEIVED DIFFICULTY</b>	<b>S.D.</b>
C1 HUMAN EVALUATION	0,635	0,680	3,685	1,302
M1 HUMAN EVALUATION	0,749	0,692	3,880	1,283
W1 HUMAN EVALUATION	1,095	0,443	4,540	0,801
AVERAGE	0,827	0,646	4,030	1,210

Table 5.2: Human evaluation results

As it can be inferred from the results, C1 got the lowest **WER** score, which means it was the speaker with the most accurate results. It was followed by M1 and lastly by W1. C1 was also the easiest model to understand for the evaluators, followed by

M1 and then by W1. Therefore, a low difficulty level of understanding relates to a low **WER**, whereas a high difficulty level of understanding relates to a high **WER**. Among the 85 people who took the test, 10 had experience in talking to people with dysarthria. We were also interested in looking at their results and see whether they found the recognition process easier and whether they obtained a lower **WER**. For this group of evaluators the mean **WER** was 0,723 /pm 0,647 and the mean perceived difficulty was 3,64 /pm 1,339. Even though the mean values for the **WER** and the perceived difficulty are a bit lower than the mean values obtained for all 85 evaluators, the standard deviation values are pretty high; we cannot conclude that these people found it easier than the rest to understand the commands.

### 5.3 Most difficult and easiest words

This section will review the easiest and most difficult words to recognize in each of the different models. This categorization was made based on the number of times a command was correctly or incorrectly recognized in each model. Although the **WER** is the most commonly used way of measuring the accuracy of speech recognition, we believed that in this case it would be informative to count the number of correct recognitions per word per model. A set of words might get a very high **WER** when having several insertions, but this does not necessarily mean that this word has been well recognized less times than another one with a lower **WER**. This is why we considered the easiest word the one which had been recognized most frequently and the most difficult word that which had been recognized the fewest times. The following table **5.23** summarizes the findings:



<b>MODEL</b>	<b>EASIEST</b>	<b>MOST DIFFICULT</b>
C1 BASE MODEL	pegar	abrir documento, alinear, ampliar imagen, apagar, cerrar documento, copiar, cortar, cursiva, deshacer, encender, guardar, imprimir, justificar, negrita, reducir imagen, rehacer, seleccionar, subrayar, tachar
C1 BASE MODEL WITH GRAMMAR	encender, negrita, pegar	tachar
C1 CUSTOM MODEL	reducir imagen	alinear, deshacer, encender, justificar, rehacer, seleccionar, subrayar, tabular, tachar
C1 CUSTOM MODEL WITH GRAMMAR	abrir documento, encender, guardar, negrita, pegar	reducir, rehacer
C1 HUMAN EVALUATION	copiar	tachar
M1 BASE MODEL	abrir documento	alinear, ampliar imagen, cerrar documento, cortar, cursiva, deshacer, encender, guardar, imprimir, justificar, pegar, reducir imagen, rehacer, seleccionar, subrayar, tabular, tachar
M1 BASE MODEL WITH GRAMMAR	apagar, cursiva, encender, negrita, reducir imagen, seleccionar	ampliar imagen
M1 CUSTOM MODEL	apagar, cerrar documento, imprimir	seleccionar
M1 CUSTOM MODEL WITH GRAMMAR	abrir documento, alinear, copiar, cortar, cursiva, encender, guardar, imprimir, justificar, negrita, subrayar	ampliar imagen
M1 HUMAN EVALUATION	encender	rehacer
W1 BASE MODEL	-	-
W1 BASE MODEL WITH GRAMMAR	encender	abrir documento, ampliar imagen, apagar, copiar, cortar, cursiva, guardar, imprimir, justificar, pegar, seleccionar, tabular, tachar
W1 CUSTOM MODEL	encender	abrir documento, apagar, cerrar, copiar, cortar, reducir, tabular
W1 CUSTOM MODEL WITH GRAMMAR	justificar, negrita, reducir imagen, subrayar	cortar
W1 HUMAN EVALUATION	encender	abrir documento, alinear, cerrar documento, copiar, cortar, cursiva, imprimir, pegar, seleccionar, tabular, tachar

Figure 5.23: Easiest and most difficult words

In cases where more than one word was included, it means that those words were recognized with the same frequency. Some of the words categorized as the most difficult ones in the case of C1 coincide with her difficulties of pronunciation. Words like *alinear*, *tachar* or *rehacer* have /r/ sounds at the end, which is quite tricky to pronounce for C1. It is interesting, however, that *pegar* or *encender* also have this feature but were sometimes categorized as the easiest words. With regards to C1 HUMAN EVALUATION, the easiest and most difficult words coincide with those subjectively perceived as the easiest and the most difficult one by the evaluators.

The same happens in the case of M1, as the words categorized as the most difficult ones have consonant clusters, such as /cc/ in *seleccionar* and the double *r* sound, as in *cerrar documento* or *reducir*. He also had difficulties when the last sound of a syllable and the first sound of the next syllable are consonants; such is the case of *imprimir*, *justificar* and *subrayar*, all present in the list of most difficult words for M1 BASE MODEL. Once again, the words categorized as easiest and most difficult in M1 HUMAN EVALUATION coincide with what the evaluators subjectively thought to be the easiest and most difficult words.

In the case of W1 BASE MODEL, there were no correctly recognized commands, and therefore we could not establish an easiest and most difficult word. Among the words categorized as the most difficult ones, we can find *cortar*, *cortar* or *cursiva*; all of them have the strong /c/, a sound that she struggles with. In the case of W1 HUMAN EVALUATION, we see that the easiest word was *encender*, which coincides with what the evaluators subjectively perceived as the easiest. The evaluators perceived *abrir documento*, *alinear* and *guardar* as the most difficult words; two of those words are categorized as the most difficult ones.

This results prove that there is a correlation between the pronunciation difficulties of each speaker and the final output of the speech recognizer. The words categorized as the most difficult ones had features that are hard to pronounce for the speakers. Apart from this, the easiest and most difficult words coincide with those subjectively perceived as easy and difficult by the human evaluators.

## 5.4 Implementation app

Unlike all the previously mentioned experiments, which were tested with pre-recorded data, the implementation app was tested with live input from microphone. Since we did not need any left-out audio sessions to conduct the testings, the models used for this app were trained with all the data (210 audio commands with their human-labeled transcripts and the related text). The finite state grammar was not included in these tests. When using a grammar in a recognizer that takes input directly from the microphone, this grammar is not finite. These tests enabled us to evaluate whether results obtained directly from microphone were better, worse or equal to the ones obtained by means of pre-recorded audios. Graphic 5.24 summarizes the results obtained.

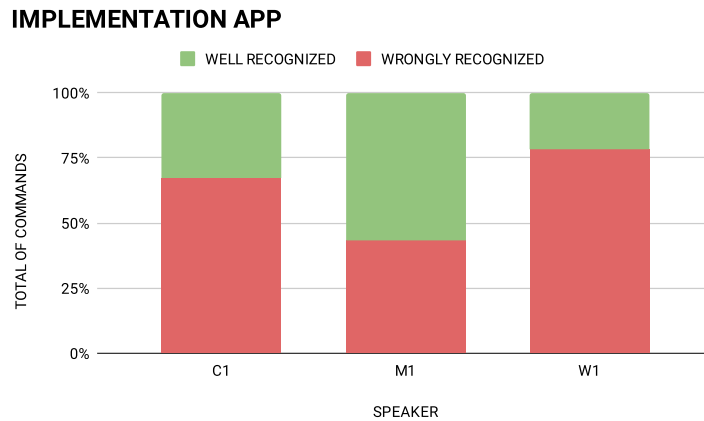


Figure 5.24: Implementation app recognition results

Figure 5.24 shows the number of wrongly recognized and well recognized commands obtained by each speaker in the implementation app; in this case, there were no "not recognized words". As it can be observed, M1 got the highest number of well recognized commands, followed by C1 and lastly W1.

In order to calculate the **WER** of these tests, we took into account the total commands that each speaker tried. Therefore, the **WER** values presented for these experiments are not a mean value, but a total value. The **WER** obtained by each speaker is shown in the table below:

MODEL	WER
C1 APP	1,06
M1 APP	0,75
W1 APP	0,61

Table 5.3: Implementation app results

C1 APP got the highest **WER**, followed by M1 APP and lastly W1 APP. In order to understand these values, it is important to look at the type of errors committed by the system for each of the speakers. This information is represented in figure **5.25**.

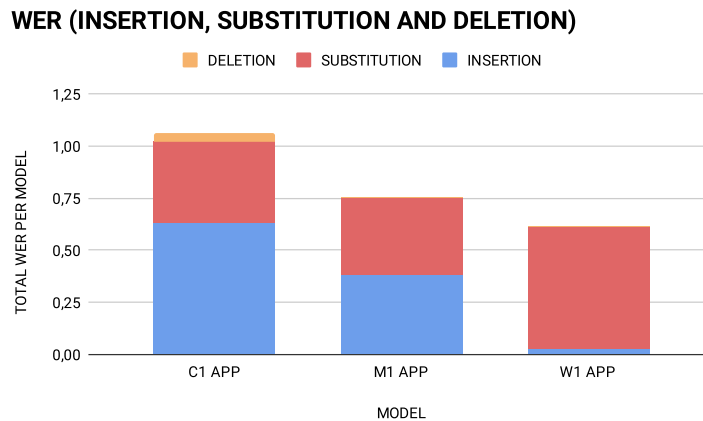


Figure 5.25: Implementation app WER per speaker

As is can be observed, C1 APP got the highest amount of insertions, while W1 APP got a very low number of them. The amount of substitutions and insertions for M1 was nearly the same. Very few deletions occur in the C1 APP model and there are no deletion occurrences in the other two models. These results are similar to the ones obtained for C1 CUSTOM MODEL, M1 CUSTOM MODEL and W1 CUSTOM MODEL, which were presented in figure **5.6**. In these models and the custom models from figure **5.6** the distribution of the types or errors is very alike. However, the number of deletions for speaker C1 is much lower in the implementation app results than in the custom model. On the other hand, the total **WER** for M1 in the custom model was lower than that of W1, while here it is higher.

It could be said that, overall, results obtained from input received from live microphone data are worse than those obtained from pre-recorded data. Test conducted

be means of this implementation app helped users understand our findings better. Apart from this, it also helped us see how the created custom models behave when being used live. This is a key point since, in the future, adapted speech technologies will have to deal with live microphone input from their users.

## 5.5 Questionnaire for participants

Participants of the experiments were asked to fill a questionnaire about the recording process and general questions about technologies adapted to dysarthric speech. Our aim was to get their perception of how easy or difficult the recording process was and how they felt during it, how they liked the implementation app and which technologies would they like to be adapted to dysarthric speech.

C1 found the recording process quite easy and very comfortable. She claimed that she did not feel tired in any of the recording sessions. She was accompanied during the recording sessions, and was provided some assistance; however, she felt that she probably could have done it by herself. Overall, her recording sessions lasted between 10 and 15 minutes. Besides, most of the commands were correctly recorded in the first attempt; this implies that the recording software did not find any interference and saved the file correctly. When asked about the implementation app, she said that although many of her commands were well recognized, it still needs to be improved. She claimed that the app recognized similar words on many occasions, but not the exact word; besides, sounds like breathing or coughing were sometimes transcribed as words.

M1 thought the recording process was quite easy and comfortable, although he said he felt tired in some of the recording sessions. He gave two reasons for this tiredness: the extra effort made in order to pronounce the commands correctly and the need to make repetitive movements with the mouse to proceed with the commands that needed to be recorded. He performed all recording sessions alone; however, he claimed that he felt the need of some help sometimes, for example to prepare the recording hardware or to control the recording software. Overall, his recording processes lasted between 20 and 25 minutes, and most recordings were successful in the

first attempt. He said he was positively surprised with the output of the implementation app, as he found that many of his commands were well recognized or nearly well recognized. He mentioned that it is very important to have a feedback with the users about any implementation apps in order to offer them the best experience as possible.

W1 felt that the recording process was very easy and comfortable. She did not feel tired during the recording sessions. On the other hand, she performed all recordings by herself and did not feel the need of anyone's help. Her recording sessions were between 10 and 15 minutes long. In her case, the recordings were never successful in the first attempt. As mentioned in [4.1.3](#), she recorded the 21 commands in a single audio file which was then split.

All the participants of this study agree on the fact that there are currently not enough adapted speech technologies for them. They all believe that there is a great need of investigation on this matter. They were asked about the fields in which these adapted speech technologies should be implemented. A list of possible implementations was provided to them, which included the following: environmental control systems (such as turning on and of the lights), control of computer programs, having conversations with virtual assistants (for example to ask them about basic information), and the dictate mode of text creation and editing programs such as *Word*. This last option enables the user to speak aloud while the program transcribes everything he/she says, and it is already available for typical speech. Participants could also suggest other fields in which they felt these technologies should be implemented. The most recurrent answers were the control of computer programs and the dictate mode of text creation and edition programs, followed by environmental control systems and having conversations with virtual assistants. Suggestions made by the participants included the urge to implement adapted speech technologies in the social ambit, in [Alternative and Augmentative Communication \(ACC\)](#) systems in face to face dialogues, controlling the television by means of voice commands, and implementing these technologies in mobile phone apps such as WhatsApp in order to transcribe voice messages. M1 also highlighted that it is very important to offer efficient alternative technologies to people with communication difficulties; in order to do so, it is vital that future user's opinions are taken into account in all the design phases.

## 5.6 Summary

It is undeniable that the recognition has considerably improved by means of training with speech from the dysarthric speakers and the use of finite state grammars. The enhancing achieved in dysarthric speech models is considerably better than the one obtained for the R1 model. This is because the R1 base model only failed in 3 commands out of 210, and therefore its improvement margin was much lower than the rest. However, the results obtained by the dysarthric speech models are far away from being as good as those obtained with the R1 model. The best results for dysarthric speech models were obtained by custom models with a finite state grammar in the case of the 3 speakers. Therefore, although an acoustic training of the system is necessary, a finite state grammar that limits the recognition possibilities to the words included in it boosts the improvement.

The types of **WER** error vary depending on the speaker and also on the model of each speaker.

When it comes to recognition confidence levels, we can see that in the case of the custom models with grammar, all values were very high, both for the well recognized and the wrongly recognized commands. This is possibly due to the fact that the grammar gives a finite list of options from which to recognize the audios. In the rest of the models, all confidence levels were more scattered.

The perceptual test conducted with 85 subjects showed that there is a correlation between the perceived difficulty of understanding and the **WER**. W1 got the highest **WER** and the evaluators found her the most difficult to understand. On the contrary, C1 got the lowest **WER** and was the easiest person to understand for the evaluators.

There is a correlation between what the speech recognizer is able to recognize well or wrong and the pronunciation difficulties of each speaker. The words categorized as most difficult ones had features that are tricky to pronounce for the speakers. On the other hand, the easiest and most difficult words coincided with those subjectively perceived as easiest and most difficult ones by the human evaluators.

Results obtained in the implementation app show how the trained system behaves when receiving live input from microphone. Overall, the results are not as good as those obtained in the tests conducted with pre-recorded data. However, the distribution of the type of errors is very similar in the custom models that were tested with audios and the implementation app models.

In general, the participants of this project felt that the recording process was easy and comfortable to conduct. All of them agreed on the need of creating adapted speech technologies for people with communication impairments. They also mentioned the areas in which they would like to find these technologies implemented.



# Chapter 6

## Discussion and conclusions

This work aimed to evaluate the possibilities of adapting the commercial Microsoft Azure **ASR** system for people with dysarthria. We have designed and recorded a pilot database with Spanish dysarthric speech, as there was none previously available. We have trained and tested a base model with data from 3 heterogeneous speakers with dysarthria. We have conducted experiments with and without finite state grammars and also with different input data: speech data from pre-recorded audios and live speech input from microphone. Nearly all existing dysarthric speech databases are in English, and have not been designed with the aim of using them to train **ASR** systems. The pilot database we designed contained 21 commands for the use of text creation and edition programs in Spanish. Apart from this, it was recorded in the users' homes; this is important because it is the environment in which the adapted **ASR** system is later tested. Besides, we have conducted a perceptual test with 85 human evaluators who transcribed audios of the 3 speakers and indicated their perceived difficulty when understanding them.

This commercial **ASR** system was not able to fully adapt to dysarthric speech. However, when customizing the base model, recognition accuracy results improved considerably for the 3 speakers. The results were not homogeneous, as they varied depending on the models used and the speaker. Among the experiments conducted, we discovered that the best results for all 3 speakers were obtained in the customized model with a finite state grammar. This means that the base model of Microsoft Azure can be customized with related-text, recorded audios and human-labeled transcripts and then add a finite state grammar in order to reduce the list of possible answers. When doing so, preliminary results were vastly improved. Although the outcomes were not as good as the outcome of the reference, the results obtained with some

models were good enough to consider the implementation of them. On the other hand, the **WER** and error types were different in all models. However, the **WER** types for each speaker were similar in the pre-recorded data tests and the input from microphone tests.

The confidence levels provided by Microsoft Azure were overall very high for well recognized and wrongly recognized words. This means that the system is very sure of every word it recognizes. Ideally, we should get a low level of confidence for the wrongly recognized words. Human evaluation results obtained from the perceptual test were worse than results obtained in the customized models with grammars. This implies that an **ASR** system with a proper training is able to outperform a human evaluation. If these models were implemented in **SGD** for example, people with dysarthria could transcribe what they say into a machine and later generate a synthetic voice for example, which could improve some of their social relations.

The integration of the customized models in computer programs is very difficult. Mainly because these programs do not allow the implementation of any external software in them or because they already have their own speech recognizer. In any case, we were able to create a webpage where we implemented the customized models and measured the accuracy obtained with live input from microphone. Results of this experiments were not as good as those obtained with the custom models with grammar and pre-recorded data. Nonetheless, the results we got outperformed the base model we tested with pre-recorded data. Besides, these findings are a proof of how this system would work in real life use.

The participants of our experiments answered a questionnaire about their experience in the recording process and general questions on adapted speech technologies. They claimed that the recording process was easy and comfortable to carry out. They all claimed that there are very few options for them and that there is a need of adapting more speech technologies that meet their needs. They also suggested different fields in which they would like to be able to implement the adapted **ASR** systems, such as in the use of computer programs, environmental control systems or in the social ambit.

It has been proved that a speaker-independent **ASR** system such as Microsoft Azure can be adapted to recognize dysarthric speech. Although the optimal solution in terms of accuracy would be to build a speaker-dependent model per user, this option has some drawbacks that need to be taken into account. A speaker-dependent

recognizer would need large amounts of data of the future speaker; the recording process until reaching decent accuracy results can be challenging for some people with dysarthria. Besides, these systems would be built especially for people with speech articulation difficulties, but probably not for the general public. Training an already existing speaker-adaptive recognizer can be a good solution to these problems. As it has been stated, the results of some of the models are good enough to be implemented.

There might be a relationship between the type and severity of the dysarthria and the accuracy results. However, the database acquired in this work included only three speakers and this number is not big enough to reach such conclusion. It is worth highlighting that the three participants of these experiments varied in age, gender and had different levels and types of dysarthria; they all had different pronunciation difficulties, and the results reflected that. Depending on the mispronunciations of the speaker, the system will need more or less training data. As we have seen, the three participants of these experiments had different pronunciation difficulties, and the results reflected that. Besides, the exact reasons behind the obtained results are not clear. We theorized that one possible feature affecting the outcome was the speech rate, since it was different for each speaker. The speaker with the closest speech rate to the reference obtained the best results. It would be convenient to execute more experiments in order to find common patterns. The closer the different voices are among themselves, the better the results would be. It might be possible that an homogeneous group of people shared some of the most recurrent pronunciation characteristics, and therefore, be able to create an homogeneous database with different speakers.

More investigation must be carried out on in the creation and adaptation of speech technologies for people with speech articulation difficulties. There is a clear need and demand of them and their implementation in different fields. Persons with dysarthria constitute one of the groups of people that could really benefit from adapted **ASR** systems. Their condition usually comes with a functional diversity; having access to these technologies would greatly improve their quality of life. If technology is to be used to improve our lives, we should think of who can benefit from its advantages the most.

# Bibliography

- Amazon Web* (2020), <https://aws.amazon.com/es/transcribe/>. Accessed: 2020-12-05. Amazon Web.
- Ballati, F., Corno, F. & De Russis, L. (2018*a*), Assessing virtual assistant capabilities with italian dysarthric speech, *in* ‘Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility’, pp. 93–101.
- Ballati, F., Corno, F. & De Russis, L. (2018*b*), ”Hey Siri, Do You Understand Me?”: Virtual Assistants and Dysarthria., *in* ‘Intelligent Environments (Workshops)’, pp. 557–566.
- Bunnell, H. T., Lilley, J., Pennington, C., Moyers, B. & Polikoff, J. (2010), The ModelTalker system, *in* ‘Proceedings of the Blizzard Challenge Workshop’.
- Calvo, I., Tropea, P., Viganò, M., Scialla, M., Cavalcante, A. B., Grajzer, M., Giardone, M. & Corbo, M. (2020), ‘Evaluation of an Automatic Speech Recognition Platform for Dysarthric Speech’, *Folia Phoniatrica et Logopaedica* pp. 1–10.
- Choi, D.-L., Kim, B.-W., Kim, Y.-W., Lee, Y.-J., Um, Y. & Chung, M. (2012), Dysarthric Speech Database for Development of QoLT Software Technology., *in* ‘LREC’, Citeseer, pp. 3378–3381.
- Christensen, H., Casanueva, I., Cunningham, S., Green, P. & Hain, T. (2013), home-Service: Voice-enabled assistive technology in the home using cloud-based automatic speech recognition, *in* ‘Proceedings of the Fourth Workshop on Speech and Language Processing for Assistive Technologies’, pp. 29–34.
- Christensen, H., Nicolao, M., Cunningham, S., Deena, S., Green, P. & Hain, T. (2015), ‘Speech-enabled environmental control in an AAL setting for people with speech

- disorders: a case study’, *International Conference on Technologies for Active and Assisted Living (TechAAL)* .
- Creer, S., Cunningham, S., Green, P. & Yamagishi, J. (2013), ‘Building personalised synthetic voices for individuals with severe speech impairment’, *Computer Speech & Language* **27**(6), 1178–1193.
- Darley, F. L., Aronson, A. E. & Brown, J. R. (1969), ‘Differential diagnostic patterns of dysarthria’, *Journal of speech and hearing research* **12**(2), 246–269.
- Darley, F. L., Aronson, A. E. & Brown, J. R. (1975), *Motor speech disorders*, Saunders.
- De Russis, L. & Corno, F. (2019), ‘On the impact of dysarthric speech on contemporary ASR cloud platforms’, *Journal of Reliable Intelligent Environments* **5**(3), 163–172.
- Deller Jr, J., Liu, M., Ferrier, L. & Robichaud, P. (1993), ‘The Whitaker database of dysarthric (cerebral palsy) speech’, *The Journal of the Acoustical Society of America* **93**(6), 3516–3518.
- Dysarthric speech: Symptoms and causes* (2020), <https://www.mayoclinic.org/diseases-conditions/dysarthria/symptoms-causes/syc-20371994>. Accessed: 2020-12-07. Mayo Foundation for Medical Education and Research.
- Elhuyar Aditu* (2020), <https://aditu.eus>. Accessed: 2020-12-05. Aditu. El reconecedor del habla de Elhuyar.
- Enderby, P. (1980), ‘Frenchay dysarthria assessment’, *British Journal of Disorders of Communication* **15**(3), 165–173.
- Enderby, P. (2013), Disorders of communication: dysarthria, *in* ‘Handbook of clinical neurology’, Vol. 110, Elsevier, pp. 273–281.
- Erro, D., Hernaez, I., Alonso, A., García-Lorenzo, D., Navas, E., Ye, J., Arzelus, H., Jauk, I., Hy, N. Q., Magariños, C. et al. (2015), Personalized synthetic voices for speaking impaired: Website and app, *in* ‘Sixteenth Annual Conference of the International Speech Communication Association’.

- Google Cloud (2020), <https://cloud.google.com/speech-to-text/docs>. Accessed: 2020-12-05. Google Cloud Speech-to-Text.
- Hawley, M., Enderby, P., Green, P., Brownsell, S., Hatzis, A., Parker, M., Carmichael, J., Cunningham, S., O'Neill, P. & Palmer, R. (2003), STARDUST; speech training and recognition for dysarthric users of assistive technology, *in* '7th European Conference for the Advancement of Assistive Technology (AAATE 2003)', pp. 959–963.
- Hawley, M. S., Cunningham, S. P., Green, P. D., Enderby, P., Palmer, R., Sehgal, S. & O'Neill, P. (2012), 'A voice-input voice-output communication aid for people with severe speech impairment', *IEEE Transactions on neural systems and rehabilitation engineering* **21**(1), 23–31.
- High, R. (2012), 'The era of cognitive systems: An inside look at IBM Watson and how it works', *IBM Corporation, Redbooks* **1**, 16.
- Hux, K., Rankin-Erickson, J., Manasse, N. & Lauritzen, E. (2000), 'Accuracy of three speech recognition systems: Case study of dysarthric speech', *Augmentative and Alternative Communication* **16**(3), 186–196.
- Kim, H., Hasegawa-Johnson, M., Perlman, A., Gunderson, J., Huang, T. S., Watkin, K. & Frame, S. (2008), Dysarthric speech database for universal access research, *in* 'Ninth Annual Conference of the International Speech Communication Association'.
- Mayo Clinic (2021), <https://www.mayoclinic.org/diseases-conditions/dysarthria/symptoms-causes/syc-20371994>. Accessed: 2021-06-11. Mayo Clinic, Dysarthria.
- Menendez-Pidal et al. (1996), 'The Nemours database of dysarthric speech', **3**, 1962–1965.
- Microsoft Azure (2020), <https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/>. Accessed: 2020-12-05. Microsoft Azure Speech to Text.
- Murero, M., Vita, S., Mennitto, A. & D'Ancona, G. (2020), 'Artificial Intelligence for Severe Speech Impairment: Innovative approaches to AAC and Communication', *Proceedings of the Second Symposium on Psychology-Based Technologies. PSYCHOBIT'20* **2730**.

- Nicolao, M., Christensen, H., Cunningham, S., Green, P. & Hain, T. (2016), A framework for collecting realistic recordings of dysarthric speech-the homeservice corpus, *in* ‘Proceedings of LREC 2016’, European Language Resources Association.
- Parker, M., Cunningham, S., Enderby, P., Hawley, M. & Green, P. (2006), ‘Automatic speech recognition and training for severely dysarthric users of assistive technology: The STARDUST project’, *Clinical linguistics & phonetics* **20**(2-3), 149–156.
- Rudzicz, F., Namasivayam, A. K. & Wolff, T. (2012), ‘The TORGO database of acoustic and articulatory speech from speakers with dysarthria’, *Language Resources and Evaluation* **46**(4), 523–541.
- Voiceitt (2020), <https://voiceitt.com>. Accessed: 2020-12-06. Voiceitt.

## Appendix A

# AZURE CUSTOM SPEECH TO TEXT MANUAL



**INDEX:**

- 1. Creating an Azure account and subscribing to the Speech service.....page 1**
- 2. Speech Studio.....page 2**
- 3. Training the system**
  - 3.1 Training data.....page 4**
  - 3.2 Training process.....page 6**
- 4. Testing process.....page 7**
- 5. Implementation of the model.....page 8**

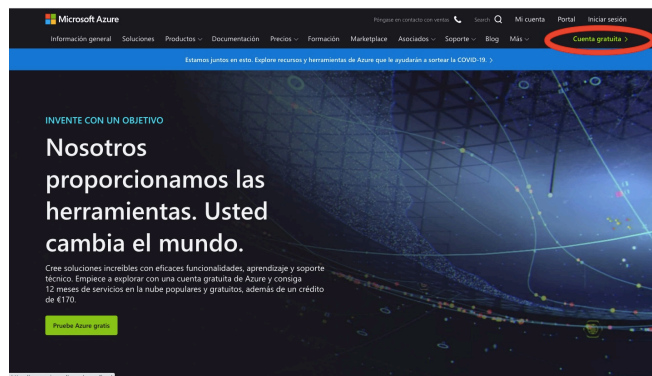
# AZURE CUSTOM SPEECH TO TEXT MANUAL

Microsoft Azure Speech to Text is a Microsoft's service feature that transcribes spoken audio into text. Azure's base model uses the Universal language model, which was trained using Microsoft-owned data. The base model is not always optimal for certain needs, for example when what we want to recognize is industry-specific vocabulary, or when the speakers have difficulties to speak. A custom model can boost the recognition accuracy of a base model. Azure enables its users to create and train custom acoustic, language, and pronunciation models. The aim of this manual is to guide people on how to create a customized speech recognition model in Azure Speech to Text.

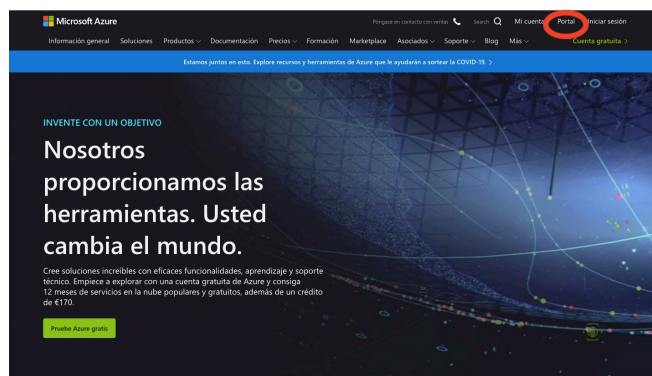
It will begin with instructions on how to create an Azure account and everything needed to start using Azure's Speech Studio, their customization portal for speech. We will then introduce the Speech Studio portal and explain how to create a new project. This will be followed by explanations on how to train a base model and how to test it later. The last section will be devoted to explain the use of the implementation of the customized models.

## 1. Creating an Azure account and subscribing to the Speech service

In order to create a customized acoustic language model, it is necessary to have an Azure account. This can be done in <https://azure.microsoft.com/es-es/> By having an account, we do not only have access to the custom speech service, but to all Azure services.

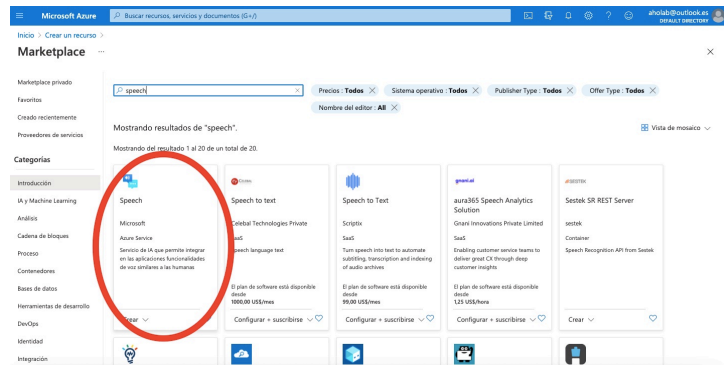
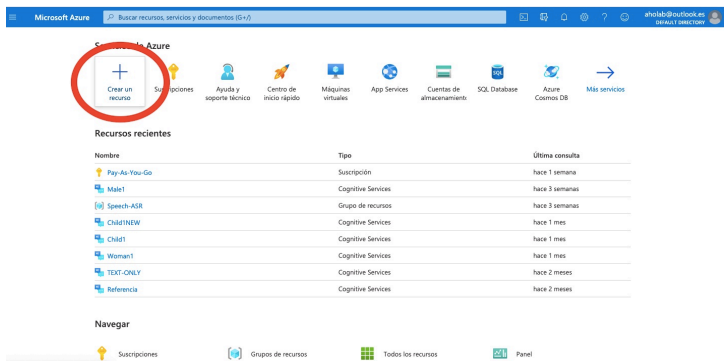


If we already have an account, we need to log in into the Azure portal.



If we still don't have a Speech service, we need to create one. For this, we have to click in "Crear un recurso". We look for "speech" and choose "Speech". This unified subscription gives access to speech-to-text, text-to-speech, speech translation, and the Speech Studio. "Speech" is one of the multiple subscription options that Azure facilitates to its users. We can create as many subscriptions as we like.

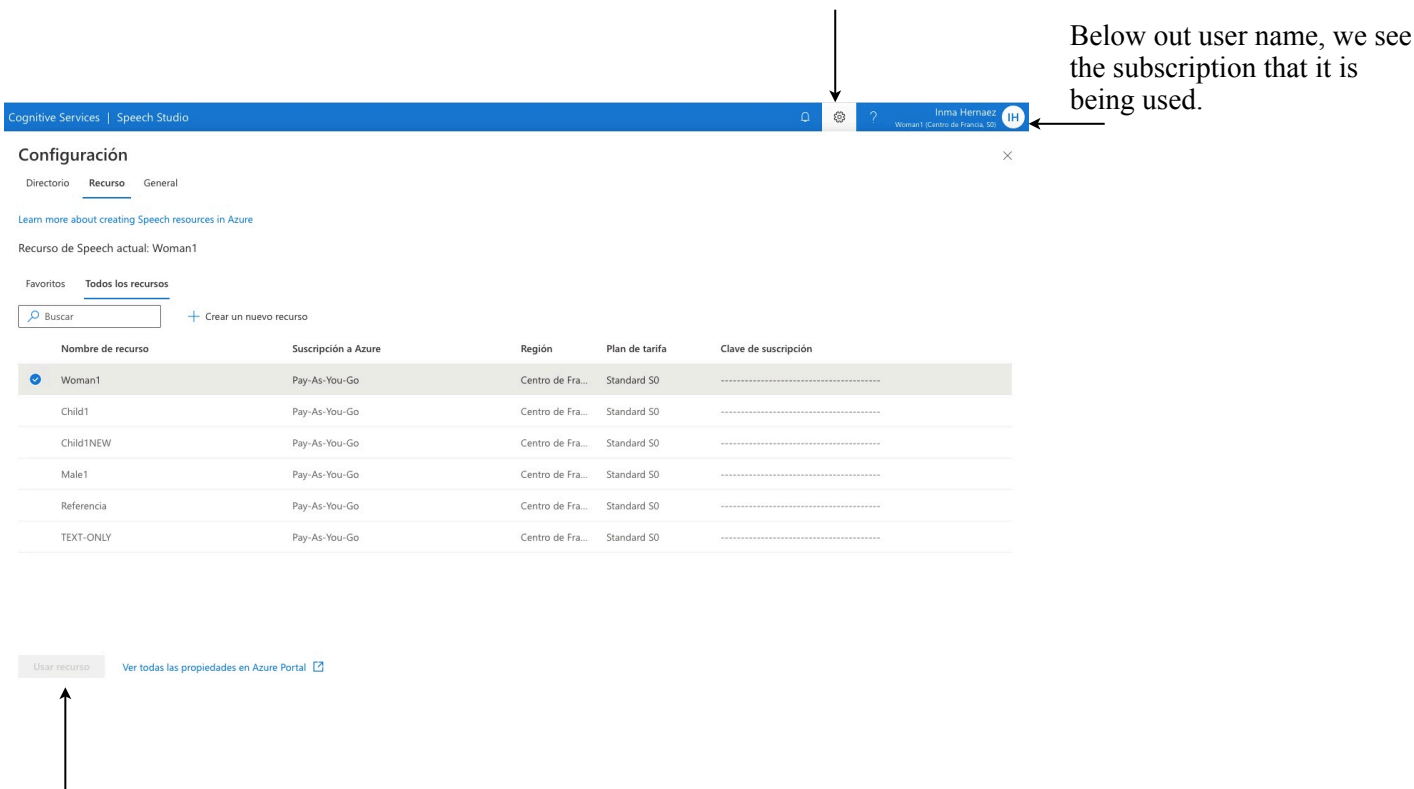
# Azure Custom Speech to Text Manual



## 2. Speech Studio

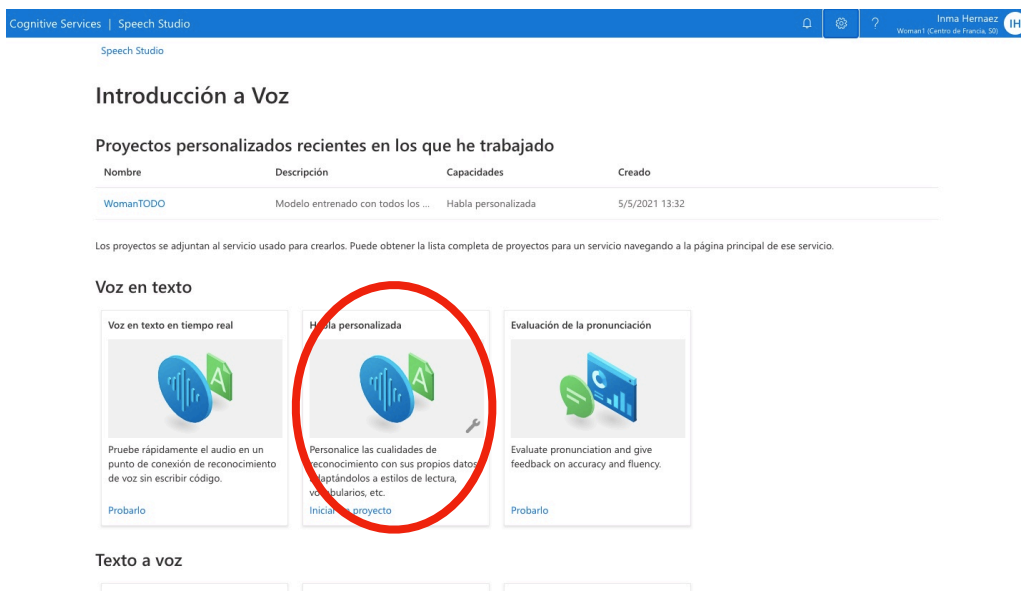
Speech Studio is Azure’s customization portal for the Speech subscription. This is the portal we need to deploy our speech-to-text custom models, text-to-speech custom models and assistants. We must first go to the Custom Speech portal ([speech.microsoft.com/customspeech](https://speech.microsoft.com/customspeech)). We should be able to see the subscription we have just created in the Custom Speech portal. If we have more than one subscription, we should be able to see all of them. We can also create new subscriptions through the Custom Speech portal instead of going back to the Azure portal, through “Crear un nuevo recurso”.

We can change from one subscription to another the subscription in the configuration part.



“Usar recurso”. This button us grey when the chosen model is the one that it is currently in use. It turns blue when we choose a different model from the one we are using.

In order to start creating custom models, we should pick one of our subscriptions and then create a new model. For that, we choose the subscription and click in “Usar recurso”. We will see all the resources available; in our case, since we want to customize a base speech-to-text model, we will choose “Habla personalizada”. If we already have created any project(s) previously, we will see them in this page, and we will be able to access them directly.

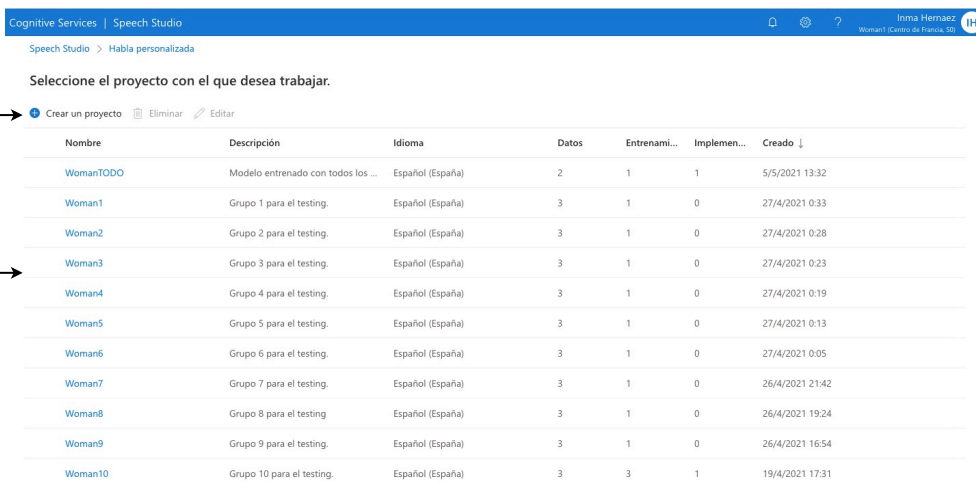


Project(s) that we have been working on recently.

Once we are in the subscription, we will click in “Crear un proyecto”. If we have already created a project, we will see them listed below. We can also access any of the models previously created. When creating the new model, the user must provide a name for the project, a description of the project (this is optional) and choose the language of the model. There is an unlimited number of projects that can be created with a single subscription.

“Crear un proyecto”. We must click here in order to create a new model.

Already created models. The most recent models appear first.



### 3. Training the system

Once we have created our project or we have entered in a previously created one, we can start training a base model. This section will introduce the types of data that can be used in Azure and the way they must be uploaded. It will later explain how to proceed with the training.

#### 3.1. Training data

First, we must upload the data we want to use. If we want to test a base model, we can simply upload the testing data and test it with a base model (this will be explained in the testing section of this manual). If we want to train a base model with our custom data, we should upload the data first, and train a base model before running any tests.

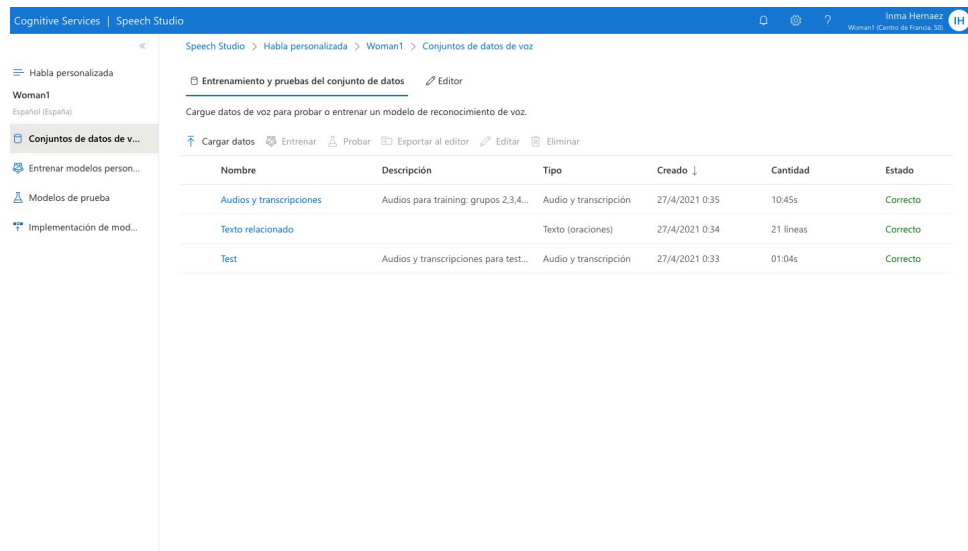
The data for the custom model may include *Audio* data, *Audio + Human-labeled transcripts* and *Related text*.

The *Audio* data is used to inspect the accuracy of speech recognition with regards to a model's performance; it is used to test the accuracy of Microsoft's baseline or a custom model.

The *Audio + Human-labeled transcripts* data can be used for both training and testing. The human-labeled data is used to evaluate accuracy and train the model for future use cases.

*Related text* data for training helps ensure correct recognition. *Related text* can be of two types: sentences, or pronunciations. Sentences may include domain-specific vocabulary or product names, among others. Pronunciations include acronyms or words with undefined pronunciations. All data types and how to upload them are explained below.

The different tabs that allow us to create a custom model. We can navigate through them as we please.



Uploaded data

Files should be grouped by type into a dataset and uploaded in a specific manner. Each dataset can only contain a single data type, must meet the requirements for the chosen data type and be correctly formatted before being uploaded.

- *Audio + Human-labeled transcripts* data can be used for both training and testing purposes. Azure recommends that each audio files includes at least half-second silence before and after speech in each sample. The audios must be in 16kHz or 8kHz, 16-bit and mono sound. Supported input formats are WAV, PCM, MP3, OPUS/OGG, PLAC, ALAW in wav container, MULAW in wav container, ANY (For the scenario where the media format is not known). Each training file cannot

exceed 60 seconds otherwise it will error out. The transcriptions for all WAV files should be contained in a single plain-text file. Each line of the transcription file should contain the name of one of the audio files, followed by the corresponding transcription. The file name and transcription should be separated by a tab (\t). Besides, transcription should be encoded as UTF-8 byte order mark (BOM). For example:

```
01Abrir.wav  abrir documento
01Ampliar.wav    ampliar imagen
01Apagar.wav    apagar
```

Once the audio files and their corresponding transcriptions are ready, they must be packaged as a single .zip file, which must be less than 2GB. More information on how to create human-labeled transcriptions can be found at <https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/how-to-custom-speech-human-labeled-transcriptions>

\*If we need to train a system with a lot of data and 2GB is not enough, we can divide the whole amount of data into different .zip files and then use of all them for the training. However, the testing can only be done with a single dataset.

- *Related text / Plain text data for training* is used to improve accuracy when recognizing specific terms or domain-specific jargon. The sentences used must reflect the spoken input. Each term or sentence must be written in a single line. This data must be uploaded as a single .txt file which should not exceed 200MB. Also, the format must be BOM UTF-8.

- *Pronunciation data for training* must be used when the audio files contain uncommon terms without standard pronunciation. However, it is not recommended to use custom pronunciation files to alter the pronunciation of common words. Pronunciation must be uploaded as a single .txt file and include the term we are interested in and its pronunciation. For example:

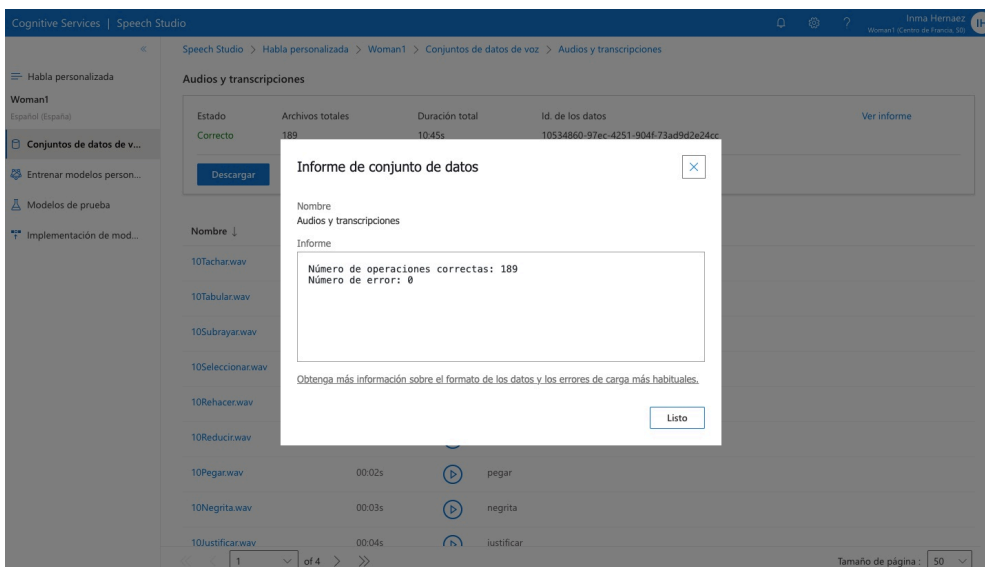
```
3CPO      three c p o
```

Customized pronunciation is currently available in English (en-US) and German (de-DE). Pronunciation .txt files should 1MB maximum and be in BOM UTF-8 format.

- *Audio data for testing* is used to inspect the accuracy of speech with regards to a specific model's performance. If we want to check the accuracy of a model we must use audio-human-labeled transcription data. The audios must be in 16kHz or 8kHz, 16-bit and mono sound. Supported input formats are WAV, PCM, MP3, OPUS/OGG, PLAC, ALAW in wav container, MULAW in wav container, ANY (For the scenario where the media format is not known).

More information on how to prepare data for custom models is available here: <https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/how-to-custom-speech-test-and-train>

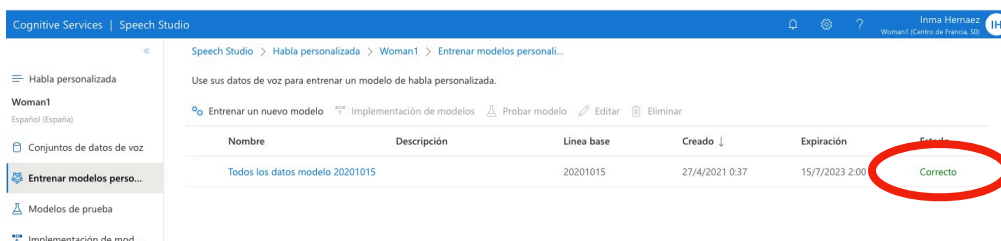
Before proceeding to the training, it is highly recommendable that we check that the data has been correctly uploaded. Even though our data shows as "Correcto", this does not mean that all data has been correctly processed. In order to do this, we can click on the name of the data and we will see all the information related to it. We then click in "Ver informe"; an additional table will appear on screen, with the phrases "Número de operaciones correctas" and "Número de errores". The former should display the total number of audio/transcriptions/lines we have uploaded, the latter should display "0". In case of having an error, we must find it (Azure normally tells us what sort of error it is), delete the uploaded data, correct the error and upload the whole data again.



### 3.2. Training process

The interface offers 4 different base model to train a customized model: 20200115, 20200619, 20201015, and 20210113. We can use the same training data with different base models and later test the customized models to see which one works best. Each model has a different expiring date, as the base models used to create custom models are updated periodically. This might cause issues with compatibility, therefore, it is important to bear in mind that both base and custom models have an expiration timeline. For base models, the adaptation is available for 1 year (after import, it is available for 1 year to create custom models), and the decoding and endpoints are available for 2 years after import. For custom models, the decoding and the endpoints are available for two years; after this time, the custom model is not lost, but it should be retrained on the renewed model.

To train a model with our data, we must first go to the training tab and then press “Entrenar nuevo modelo”. We will choose the “Escenario” (which usually only allows to use “General”), and the base model we want to use. We will then choose the data we want to use for the training, which might be *Audio + Human-labeled transcripts*, *Related text* or both. Finally, we will give our model a name and, if like, also a description. Once the model is trained with our data, we will see “Correcto” next to the model. The training phase can take from some minutes up to days, depending on the amount of training data we have.

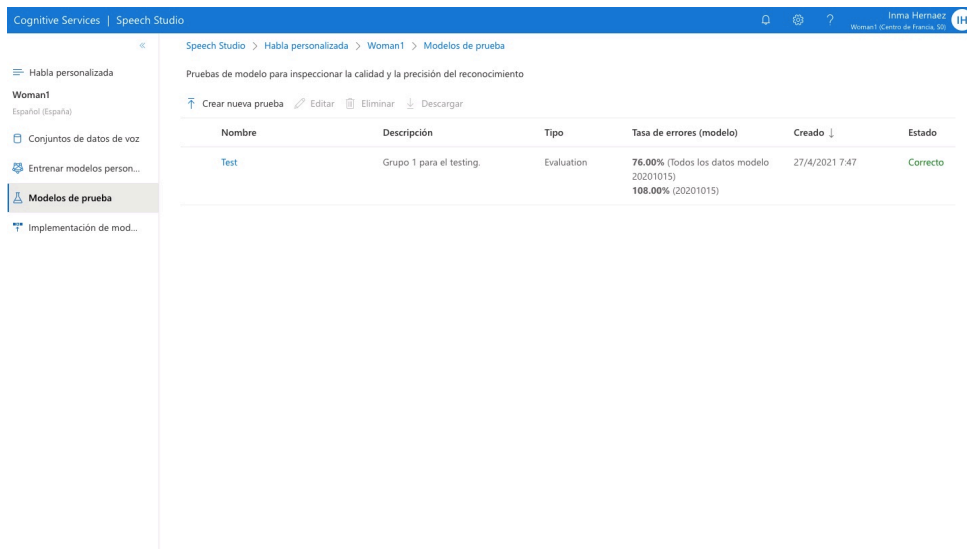


## 4. Testing process

The tests can be performed with base models or custom models. If we are not sure about whether training a model is necessary, we can run a test with different base models and see which one works best. If none of the results is good enough, we can pick any of the models (usually the one with the highest accuracy result) and train it.

In order to see how accurate our models are, we must perform one or more tests. The testing can be performed with both the training data or the testing data. If the data provided for the tests is only *Audio* data, the platform will provide a transcription of the audio files. We will obtain different results of our testings depending on the data we use for it. If the data used in the tests is *Audio + Human-labeled transcript*, the platform also provides a WER (Word Error Rate). This is done by comparing the model's transcription with the reference transcription. The tests can be run with a maximum of two models simultaneously, which can be base models or customized models. After performing a test with *Audio + Human-labeled transcript*, we can also download a folder with all the information related to this test. To run a test, we must go to the testing tab and choose “Crear una prueba”. We will see two options “Inspeccionar calidad” (for audio data) and “Evaluar precisión” (for audio-human-labeled transcripts data). Once we choose the type of test, we will see the data available for the testing; here we can only choose one dataset. We will finally have to choose one or two models that we want to test and give a name to our test.

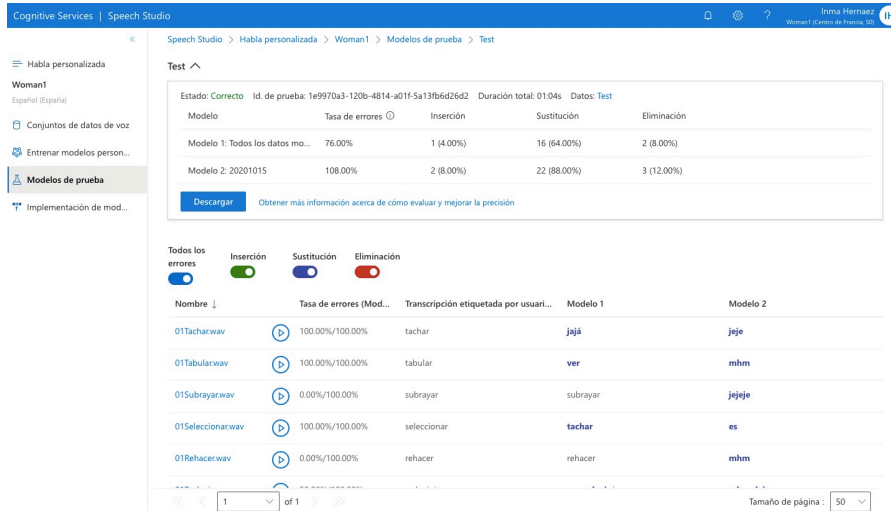
In the picture below, we can see that a test was performed with two different models simultaneously; in this case, one was the base model 20201015 and the other was a customized model trained with 20201015 base model. Since the test was an evaluation, the system compared the recognized sentences by each model with the human-labeled transcripts and thus provided a WER.



The screenshot shows the 'Modelos de prueba' (Test Models) section in the Azure Speech Studio interface. The breadcrumb navigation is 'Speech Studio > Habla personalizada > Woman1 > Modelos de prueba'. The main heading is 'Pruebas de modelo para inspeccionar la calidad y la precisión del reconocimiento'. Below this, there are buttons for 'Crear nueva prueba', 'Editar', 'Eliminar', and 'Descargar'. A table displays the test results:

Nombre	Descripción	Tipo	Tasa de errores (modelo)	Creado ↓	Estado
Test	Grupo 1 para el testing.	Evaluation	76.00% (Todos los datos modelo 20201015) 108.00% (20201015)	27/4/2021 7:47	Correcto





If we click on the name of our test, we will be able to observe the total WER, the insertions, the substitutions and the deletions for each of the models. We can also observe what each of the models recognized for each audio file and what kind of errors the models committed when compared to the human-labeled transcriptions. All the information provided here can be downloaded as .txt files for the recognized utterances, and JSON files for detailed information about the recognized utterance.

### 5. Implementation of the model

The implementation of a model consists of creating a connection endpoint in order to use our model outside the Speech Studio. The implementation can be done with any of the models, base or custom models.

The implementation is conducted for different objectives. On the one hand, it can be used as a final model, which can be used in our own recognizer or in other programs in which we are able to implement it, or to transcribe directly from microphone. On the other hand, we can use the connection endpoint to run different tests. The latter would be the case when willing to include a grammar in a testing. For this, we need to perform our tests in a Python/JAVA/C#/C++/Go/Node.js/JavaScript/Objective-C Swift/REST/CLI. Grammars consist of a list that limits the recognition options to the words included in it. Therefore, the system is not able to recognize any words outside the grammar. This boosts the possibilities of a word to be recognized. Grammars cannot be included in the training data before customization.

A script must also be used when transcribing directly from the microphone. The Azure Custom Speech Studio interface only allows conducting testings on previously recorded data, which must be uploaded to the system. This information is provided here: <https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/get-started-speech-to-text?tabs=windowsinstall&pivots=programming-language-python>

In order to use custom models in a script we first need to create a connection endpoint in our Custom Speech interface. We must go to the implementation tab and choose “Implementar modelo”, we will give it a name, a description if we like, and we will choose the model we want to implement (which might be a custom model or a base model). Once the implementation is finished,

we click on it and we will see the speech-key, the region and the connection endpoint ID, which is the information needed to implement our model.

