Universität Zürich[UZH]

Master thesis
zur Erlangung des akademischen Grades
Master of Arts
der Philosophischen Fakultät der Universität Zürich

(Titel)

Author: Deborah N. Jakobi
Matriculation number: 16-054-165

Supervisor:

Institut für Computerlinguistik

Abgabedatum: (xx.xx.xxxx)

Abstract

All scripts and files produced along with this thesis are found in my GitHub repository: `https://github.com/theDebbister/masterThesis`.

Zusammenfassung

Alle Scripts und Dokumente, die ich im Zusammenhang mit dieser Arbeit erarbeitete, sind auf meinem GitHub zu finden: `https://github.com/theDebbister/masterThesis`.

Acknowledgement

Phillip Ströbel from the CL institute at the UZH for his help with the OCR technologies. Lysander Jakobi for writing the Hebrew orthographic transcription. Florina Vogel and Gazal Hakimifard for helping with the Farsi transcription.

Contents

List of Figures

List of Tables

List of Acronyms

IPA	International Phonetic Association
IPA	International Phonetic Alphabet
WER	word error rate
CER	character error rate
PER	phone error rate
G2P	grapheme-to-phoneme
FST	finite-state transducer
EM	Expectation-Maximization
seq2seq	sequence-to-sequence
JIPA	Journal of the International Phonetic Association
SIGMORPHON	Special Interest Group on Computational Morphology and Phonology
NLP	natural language processing
OCR	Optical Character Recognition
UZH	University of Zurich
WALS	World Atlas of Language Structures
ASR	automatic speech recognition
TTS	text-to-speech
HTR	Handwritten Text Recognition
NWS	*The North Wind and the Sun*
RNN	reccurent neural network

1 Introduction

With the advent of technologies that can process huge amounts of data, many linguistic tasks that were originally very tiresome and expensive to do, can now be accomplished much faster. Well known examples for this branch called natural language processing (NLP) are machine translation or search engines. A lot of available tools and consequently research done in this area is concerned with written text. For many scenarios like machine translation, large corpora of written text in many languages are collected that are used to train computational models that solve those tasks. Following, there is an ever-growing set of models that are trained on written text. Often the goal is to reach or outperform human solutions to those various tasks. Those corpora serve as example for how the task in question can be accomplished. It is therefore necessary that this training data represents language well enough for a machine to reach that performance. From a linguistic point of view, the questions comes up if focusing on written language only can ever represent human language adequately. Most of communication and daily language use happens through speaking. This is a first potential limitation of many (written) language technologies. It is not easy to find out what characteristics of a language can be observed in written representations. There are technologies like automatic speech recognition (ASR) or text-to-speech (TTS) that require the mapping of written to spoken language. Spoken language in those cases is mostly represented as phonetic transcriptions as those are easier to process. Those research foci do contribute to the questions how spoken and written language relate. But still, this does not answer the question of how well written language represents language in general. The representative power of written text is much less studied. This is where this current thesis connects to cutting-edge research. I am going present my attempt of studying a multilingual phonetic corpus and comparing it to its written-text version. I will try to answer the following question: **Is it essential for the study of multilingual corpora to perform analyses on phonetic text (i.e. speech representations) rather than only written text?** It becomes clear, when looking at these considerations, that there are a few huge topics addressed. None of these is trivial and can be answered easily. While this thesis cannot possibly discuss everything from the use of phonetic transcriptions up to the nature of human language use, the aim is to make

a step into the direction of quantifying the representative power of written text.

1.1 Research questions & goals

The text group of the Language and Space lab at the University of Zurich maintains a project that provides a multilingual corpus consisting of 100 language text samples [SPUR project] which is referred to as 100LC (see section 2.5). Those 100 languages are meant to be representative for all the world's languages which is explained in more detail in section **??**. It is therefore meant to give insight on relations, similarities, differences or properties of individual languages or language families. Specifically, their goal is to use quantitative methods like statistical modelling, machine learning and information theory to study language variation and compare languages. The goal is now to collect phonetic transcriptions of the corpus. The same analyses that are performed on the original written corpus can be performed on the phonetic texts and both can be compared. In order to add a phonetic corpus to the already existing one, various steps need to be performed which are outlined below:

1. **Data collection:** The given dataset contains no phonetic transcriptions of those 100 languages. The first step is to find already existing data.
2. **Phonetic transcriptions:** As existing data will not be available in sufficient amounts to perform meaningful analysis, the next step is to actually create phonetic transcriptions of as many languages as possible of the corpus. This will require to create and train a computational model.
3. **Calculations and Analysis:** Once the transcriptions have been obtained, the newly created phonetic corpus can be analysed and calculations can be performed.

The first two steps will already take up quite a lot of time which is hard to estimate in advance. It is not clear how much of the third step can be accomplished. I still think it is important to keep the bigger picture in mind and be aware of the overall importance of the topic. By performing these steps I am aiming at answering the following two questions:

1. What types of phonetic data is available and how can it be used?
2. Which computational models can be used to create phonetic transcriptions?
3. How can we use phonetic features to create phonetic transcriptions?

4. Is there any significant difference in comparing spoken or written languages?
5. Does written text represent language well enough to justify text-based research only?

Again, the last two questions are related to the third step above. Those are important questions but it needs a lot of work to actually answer them. The third questions is of particular importance as it is a new approach to creating phonetic transcriptions.

1.2 Thesis structure

The thesis is subdivided into six chapters of which the first chapter is this introduction. Chapter 2 covers the linguistic basics that are necessary to understand the topic. It presents the linguistic foundation of phonetics and phonology and an introduction to corpus linguistics or rather corpus phonetics. Chapter 3 sets the boundaries of the technical background. I will present an overview of the possibilities for automated creation of phonetic transcriptions. In chapter 4, I document my first practical experiments which are concerned with data collection. It contains a descriptions of the data I will later use to create models for phonetic transcriptions. Chapter 5 presents my own experiments to create phonetic transcriptions of the corpus. I will present all models that I created and their results. Chapter 6 summarizes my findings and presents ideas for future research.

2 Linguistic Background

From a linguistic point of view there are three high-level concepts that are important for this thesis. The first and most significant one is that of the **relation of written and spoken language and their representativeness**. As I have already mentioned, computational linguistics deals mostly with written languages rather than spoken language, but what does traditional linguistics say about the relation of written and spoken language? Both written and spoken language are valid representations of a language. As we will see later in this chapter, mapping a spoken language to its written representation is far from easy and never perfect. Elaborated writing systems like we are used to today came only much later compared to language in general [CrashCourse, 2021b; Hock and Joseph, 2019]. The question that gave rise to this thesis is if writing systems can capture language as such well enough or rather if we need spoken representations as well. Whenever we study language we look at samples of that language. It is simply impossible to study an *entire* language as we would need all texts that were ever produced in that language. Consequently, we need to ask ourselves how much and what material of a language is enough to study it properly. Baird et al. [2021] focus on answering the question of how much phonetic data is needed to represent a language well. We know that each language has specific sounds, its phoneme inventory, that are frequently used. The question is now how much data is needed to cover this entire phoneme inventory of a language. In order to answer that question, Baird et al. [2021] study *The North Wind and the Sun* (NWS) corpus. This corpus will become important in section 4.1.2 and I will go into more details about this study at that point. While it would exceed the scope of this thesis to go into too much detail about how well my data covers the languages, it is important to keep in mind that just because a computational model produces good results, this does not mean it actually represents a language well.

The second important concept is the analysis of large amounts of text in any language which is commonly referred to as **corpus linguistics**. Corpus linguistics allows for both qualitative and quantitative analysis of text. Although text can refer to written or spoken language, most corpora contain written text [McEnery and Hardie, 2011]. Multilingual corpora can be used to compare languages. Due

to recent technological advancement it has become possible to store large digital collections of speech recordings and their aligned transcriptions. These possibilities gave rise to a wider acknowledgement of corpus phonetics. Corpus phonetics deals with an abundance of linguistic variation. In addition to language, style or vocabulary variation, there are differences in dialect and idiolect, physiological state of the speakers and their attitude [Liberman, 2019; Chodroff, 2019]. Many methods and tools used in corpus phonetics are based on ASR algorithms or simple programming [Chodroff, 2019].

The third key concept is that of **multilingual analysis**. The 100LC underlying this thesis is multilingual. While a multilingual corpus in itself is not uncommon, the specific goal of the team maintaining the corpus is to compare the languages and study their similarities. Comparing languages and studying their similarities and differences is part of a well-established branch of traditional linguistics called comparative linguistics. For example, Gutierrez-Vasques et al. [2021] performed a study on the 100LC comparing different subword tokenizations. They compared 47 languages to find out how languages can be tokenized such that it leads to similar distributions among that languages. This analysis was performed on written language and is an example for what could be done on a phonetic corpus as well.

While there is a lot that could be said about any of these topics, I am not going into more detail about these. The point I would like to make is that the task of creating phonetic transcriptions of a multilingual corpus with the long-term objective of comparing written and spoken language is deeply rooted in linguistics and not at all trivial. In the following sections I will mostly talk about phonetics and phonology in general and writing systems.

2.1 Phonetics and Phonology

Given that phonetics and phonology is a sub-area of traditional linguistics and often only touched on superficially in computational linguistics, I will summarise the most important assumptions and terms concerning said field. A very important terminological distinction is between phonetics and phonology. While phonetics refers to the study of actual sounds, phonology refers to the study of sound *systems*. In phonetics, it is not so much important what the different sounds mean, but how they are produced and perceived and what different sounds a human being can produce and perceive at all. When it comes to human communication using spoken language, many of these sounds are not actually used to produce distinguishable meaning. This is why on the other hand, phonology is important to describe the set

of distinguishable sounds that make up a language. For example: the letter /r/ in English can be pronounced in many different ways in a specific word like 'request'. I can roll the /r/ like it is common in Spanish or I can pronounce it like an /r/ in German. None of those pronunciations produces a change in meaning. Others might say I have an accent or I speak a dialect but usually people will understand. This means that there exist many different *phonetic* sounds but only one *phonological* or *phonemic*. Physically, there are many sounds I can produce that map to the /r/, but none of them changes the meaning (at least not in English, in other language this can be different of course). Those sounds are referred to as phone and phoneme respectively. While there are infinitely many phones, there are only finitely many phonemes in a language. Also, in one language there is typically a set of phones that is used by the majority of speakers of this language that can replace phonemes and does not change the meaning. Sounds that can replace another sound without changing the original meaning are referred to as 'allophones'. Each language has therefore a set of phonemes, a phoneme inventory, and a set of allophones, each of which maps to a specific phoneme. How phonemes and allophones are used depends a lot on the dialect and idiosyncratic language use [Kracht, 2007].

Not all different possible sounds are actually considered qualitatively 'good' sounds of a language. Usually there is a subset of all possible phones that is accepted as 'good quality sounds' within all different dialects of a language [Kracht, 2007]. An obvious example being loudness: Although very silent speech produces correct phones, these are not 'good quality' as they simply cannot be understood. Or speaking in English with hardly any mouth and tongue movement. Although this produces understandable sound, it is not generally considered good speech.

It is important to note at this point that the terms phonetic and phonemic respectively phone and phoneme are sometimes used interchangeably. Their linguistic definition as given above is relatively clear while the definition on the computational side is often less strict. Strictly speaking phonemic transcriptions are not allowed to contain allophones but should write the respective phoneme of that language. This will not always be the case when it comes to data used in language technology [Lee et al., 2020].

In the following, I will outline different terms and concepts that appear a lot in research and literature concerned with phonetics and phonetic transcriptions.

Vowels and consonants Each phone can be described based on different categories. A well-known distinction is that between vowels and consonants. Both of these are again categorized differently. The schema for vowels and consonants is

inspired by the human vocal cavity. The terms to describe vowels sounds are based on the position of the tongue in the mouth and if the lips are rounded or not. Using those two categories enables us to distinguish every possible vowel. The position of the tongue in the mouth is described along to axes: back and front, and open and close. A vowel can then, for example, be described as close-back unrounded (which would be [ɯ]) or open-front rounded ([œ]). Figure 2 shows the vowel chart how it is usually represented in the International Phonetic Alphabet (IPA). More on this special alphabet and writing systems in general follows in section 2.4. Consonants are defined by the place and the manner of their production. The place, again, refers to the position of the tongue in the mouth and the overall form of the vocal tract. The vocal tract is used to block the air and make it flow in a specific way. The manner, on the other hand, describes the way the air is lead through the mouth or how it is blocked to produce a sound [CrashCourse, 2021a]. For dental sounds, the tip of the tongue is moved to the upper middle teeth. For palatal sounds, the body of the tongue is pressed against the hard palate in the back of the mouth cavity. These are examples for places of articulation. Examples for the manner are plosive or trill. A trill makes the tongue move in a vibrating way which consequently makes the air vibrate. A plosive first completely blocks the air and then pushes the air out of the mouth in a fast manner, a bit like an explosion, therefore the name. Both vowel and consonant categorization is rather intuitive and pictorial. The complete consonant chart is depicted in figure 2 as well. The exact description of each phone is not always very important when dealing with computational models for phonetic transcriptions. The key point is that that we can describe sounds uniquely using different features.

Syllables Phonemes, or letters, can be grouped into larger units called *syllables*. Syllables can be an entire word or a part of a word. English syllables typically consist of a group of consonants followed by a group of vowels or a diphthong followed by a group of consonants again. These parts are called *onset*, *nuleus* and *coda* respectively. For every syllable in every language it is true that the nucleus cannot be empty. The onset and the coda can be empty. Other than that, syllables are organized very differently in different languages [Kracht, 2007]. For computational analysis it is important that syllables are a way of segmenting written or spoken language. They are larger than individual letters or phonemes but often still smaller than individual morphemes or words.

Diphthongs A diphthong is a sequence of vowels that is considered as just one phoneme if is is within one syllable. If a syllable ends with a vowel and the next

one starts with a vowel, this vowel sequence is not called a diphthong. An example is the German word 'Chaos'. The two vowels in the middle are not a diphthong as there is a syllable boundary right in their middle: 'Cha-os'. A word like 'aus' contains a diphthong as it exists of only one syllable [Kracht, 2007].

Suprasegmentals Apart from individual sounds, there are features of spoken language like stress or intonation. Those are referred to as suprasegmentals. They are often related to syllables. For example, we can put stress on a different syllable or raise the pitch. Semantically, some suprasegmentals in some languages distinguish meanings, some do not. A special case are tones. Tones are a special way of intonation. In some languages like Chinese or many African languages tones are used to distinguish meaning while in most European languages, the concept of tones does not exist [Kracht, 2007].

2.2 PHOIBLE

A way to analyse or use phonetic corpora is to use phonetic features to represent each phoneme. These features are a list of properties that are overlapping with the phonetic description of each phoneme that I have mentioned before when talking about vowels and consonants. It is a minimal list that can be used to describe unique phones. There exists an online database called PHOIBLE [Moran and McCloy, 2019] that contains over 3,000 phonemes for more than 2,000 languages. PHOIBLE includes a feature system that can describe each phoneme uniquely. In total, there are 37 features that are used to describe the phonemes. Each of the features can take on three values: either '+' (applies to this phoneme), '-' (does not apply) or '0' (not applicable). In addition to the phonetic features each phoneme has other features like for example what allophones are used for it in a specific language. The entire PHOIBLE inventory is structured around languages. It lists all phonemes including features for all languages that are in there. This means that one phoneme can be listed for more than one language, but the features will always be the same. All the PHOIBLE phonetic features are listed in table 1. Some of them might sound familiar from the linguistic background chapter. The exact description of each and every feature is not important at this point. The main point is that these features allow to uniquely describe each phoneme.

For this present thesis, PHOIBLE does not only serve as a phonetic feature database but as a phonetic reference database in general. As it contains a very complete list of phonemes for each language it is a great help to clean phonetic data and detect

tone	tap	strident	epilaryngealSource
stress	trill	dorsal	spreadGlottis
syllabic	nasal	high	constrictedGlottis
short	lateral	low	fortis
long	labial	front	raisedLarynxEjective
consonantal	round	back	loweredLarynxImplosive
sonorant	labiodental	tense	click
continuant	coronal	retractedTongueRoot	
delayedRelease	anterior	advancedTongueRoot	
approximant	distributed	periodicGlottalSource	

Table 1: This table displays all the features that are use in PHOIBLE to describe one phoneme. Each feature is either '+' (applies to this phoneme), '-' (does not apply to this phoneme) or '0' (not applicable).

mistakes or uncommon transcriptions.

2.3 Mappings of written and spoken language

Unlike spoken language that was a part of human interaction all the time, writing systems only developed over time. There are different writing systems that developed in different places at different times. The structure of the spoken language, the cultural context or the tools that were at hand to write are a few of many factors that influenced the emergence of a specific writing system. In general, we can think of writing systems as mappings from spoken language to written language. The systems used to represent sounds in different languages do not uniquely map a letter, or a grapheme, to one specific phoneme. Most of the time, there is a standard pronunciation of each letter that is trained by reciting the alphabet. However, in reciting the alphabet there is a vowel added to the consonants in order to pronounce them more easily. This means that reciting the alphabet is somewhat artificial when compared to what sound is actually produced in natural spoken language. These explanations make clear that the mapping of written text to spoken text in various languages is complex. When taking a step back, we can see that a single grapheme can represent either a phoneme, a syllable or a word. The history and development of writing systems is an entire independent study area. For this thesis it is mostly important to be aware of the independently developing systems. Not all scripts can be treated the same and this most certainly has implications on models to create

phonetic transcriptions. Each major mapping will be presented below [CrashCourse, 2021b].

Alphabet When a grapheme maps to roughly to one phoneme, we call this an alphabet. In German, for example, the writing system consists of the Latin alphabet. The Latin alphabet is used for many different languages in western Europe and those languages that were influence by colonisation. There are other alphabets like the Cyrillic or the Greek alphabet. Having an alphabet does not mean that each grapheme, or letter in this case, maps to exactly one phoneme. In fact, one grapheme can have many different realizations as example 2.1 shows.

(2.1) The examples show the different realizations of the English grapheme sequence 'ough' [CrashCourse, 2021a]. The first part is the grapheme sequence and the second part the phonetic transcription.

(a) tough [tʌf]

(b) cough [kɒf]

(c) though [ðəʊ]

(d) through [θruː]

(e) bough [baʊ]

(f) brought [brɔːt]

The above examples show that it is not possible to have a one-to-one mapping from one grapheme or a sequence of graphemes to one phoneme or a sequence of phonemes with in the English language. Let alone within all languages that use the Latin alphabet. In addition, alphabets typically have diacritic marks that can be used to extend the main letters. Just as with single graphemes, also diacritic marks cannot simply be mapped to a phoneme.

Abjad A special variant of an alphabet-language is abjad. Abjad represents only consonants and no vowels. This means that vowels need to be added while reading. Again, this means that there is a lot of ambiguity as it is not always clear which vowel should be added if there is no context. Semitic languages like Hebrew or Arabic make use of abjad.

(2.2) Hebrew examples that are first mapped to Latin alphabet then to the Latin alphabet including vowels.

(a) בצלם bzlm bzelem

(b) בצלם bzlm bzalam

Example 2.2 shows that in this case each grapheme maps to a consonant but it can be completed with different vowels. The words presented above do not have the same meaning depending on the vowels added although there grapheme sequence look exactly the same.

Syllabary In syllabaries, a grapheme represents a syllable instead of a single sound. Examples are the Japanese Hiragana and Katakana. Both of these examples do not have any internal ambiguities in their pronunciation as one grapheme maps to exactly one phoneme. However, in the case of Japanese, in addition to the syllabaries they use a logographic system as well which is ambiguous.

Logographic systems Logographic systems represent entire words or morphemes as graphemes. Chinese is an example for a logographic system. We cannot break down Chinese signs into single morphemes or letters. One letter, which is called a logogram, is typically pronounced in the same way regardless of the context of the sign. Logographic systems are less ambiguous than alphabets for example. What makes a logographic system like Chinese difficult to pronounce is the sheer amount of sings that exits. For each logogram, one must know how to pronounce it and in Chinese you need a few thousand logograms for daily usage only.

What all of these mappings have in common is that they are no reliable source of pronunciation as the examples above show [Kracht, 2007]. Many of the pronunciation rules of a language are based on convention. Speakers of a language just *know* how to pronounce a word. Still, there can arise heated debates about the correct pronunciation of certain words. Just think of Swiss German dialects. Apart from these conventions, spoken and written languages change differently over time. Spoken languages are typically more flexible and ready to change while their written representation often stays the same [Moran and Cysouw, 2018]. This can lead to official governmental interventions like the German orthography reform of 1996 that intended to adapt the German spelling to represent the German pronunciation more adequately. Also, major inventions like printing machines gave rise to standardization of writing systems as reading and writing became more common [CrashCourse, 2021b].

2.4 The International Phonetic Alphabet (IPA)

An exception to the above explained characteristics of an alphabet are phonetic alphabets like the IPA where each grapheme is intended to represent exactly one phone [CrashCourse, 2021b; Kracht, 2007]. As usual, reality is more complex than what we wish it to be. Even with the IPA there are inconsistencies. Figure 2 shows the full IPA chart including all characters that the International Phonetic Association (IPA) decided to use. Although the IPA seems very complete there are still sounds that cannot be represented using the IPA. This becomes clear when, for example, looking at the vowel chart (see figure 2). The tongue does not 'click into place' for the vowels on the chart. Vowel characterisation happens on a continuum. This means that it is always possible to characterize a vowel as in between two vowels on the chart. The IPA is not the only transcription convention but it is very common (at least in this present research setting).

Apart from different character sets that can be used to represent sounds, there are different levels of detail. Not all transcriptions represent the phonetics in equal detail. Generally, there is the distinction of broad and narrow transcription. These two go back to the linguistic distinction of phone and phoneme. Broad refers to a phonemic description. Following the linguistic definition in chapter 2, this means that the transcription does not transcribe speaker specific pronunciations or dialectal variations. It uses the phonemes in the phoneme inventory of the language in question. This kind of transcription is therefore less complex and usually easier to create and understand. Narrow transcriptions are phonetic. They present every speaker individual or dialectal sounds as exactly as possible. Although the spoken text in narrow and broad transcription sounds only minimally different, the two texts can diverge greatly. It is important to treat broad and narrow transcriptions as two different kinds of transcriptions.

(2.3) pɪˈkʊ kəˈz̩əf

(2.4) pɪˈkʰʊ kʰəˈz̩əf

Example 2.4 is a narrow (phonetic) transcription of the beginning of the Mapudungun version of the short story *The North Wind and the Sun*. The same text is transcribed broadly (phonemic) in example 2.3. As becomes clear in this example, the narrow transcriptions is longer as it contains more different characters. In this case it is only the superscript h that is different. The problem, with especially the narrow transcriptions, is that the transcriber still needs to define what narrow means in a specific case. One could argue that there are as many narrow transcriptions

of a language as there are speakers of that language. This becomes tricky when given a task to automatically transcribe text. The training data might employ one definition of narrow, while there are texts in the test set that might follow another definition. This is mostly important when talking about data preprocessing and cleaning.

2.5 The 100 language corpus

As mentioned in the introduction, the basis of the data used in this thesis is a corpus provided by the SPUR lab at the University of Zurich (UZH). The corpus contains 100 languages which are proposed by Comrie et al. [2013]. This is an online book that contains different chapters each of which shows a different linguistic feature including a map which shows the distribution of that feature over the world's languages. While the number of languages presented on the individual maps depends on the amount of research done in a specific area, the sum of all maps gives quite an impressive overview on the structure of nearly half of the world's languages. Out of the 2676 languages a sample of 100 languages was chosen. This sample does not contain too many languages from one area, neither does it contain too many languages from one family. Those are the 100 languages that are in the corpus. Not considering the aforementioned criteria of maximizing genealogical and areal diversity can lead to misleading results in multilingual analysis. Figure 1 shows the distribution of the corpus on a world map. The different icons show the genus of the languages which is a classification of languages defined by the World Atlas of Language Structures (WALS) team that maintains the language description collection. The interactive map can be viewed online [100-language-sample]. Table 9 in the appendix A shows all languages that are in the 100 language corpus. None of the text samples are provided by WALS. The entire corpus is provided by the SPUR team that collected the corpus over the last few years and is continuously working on and with it.

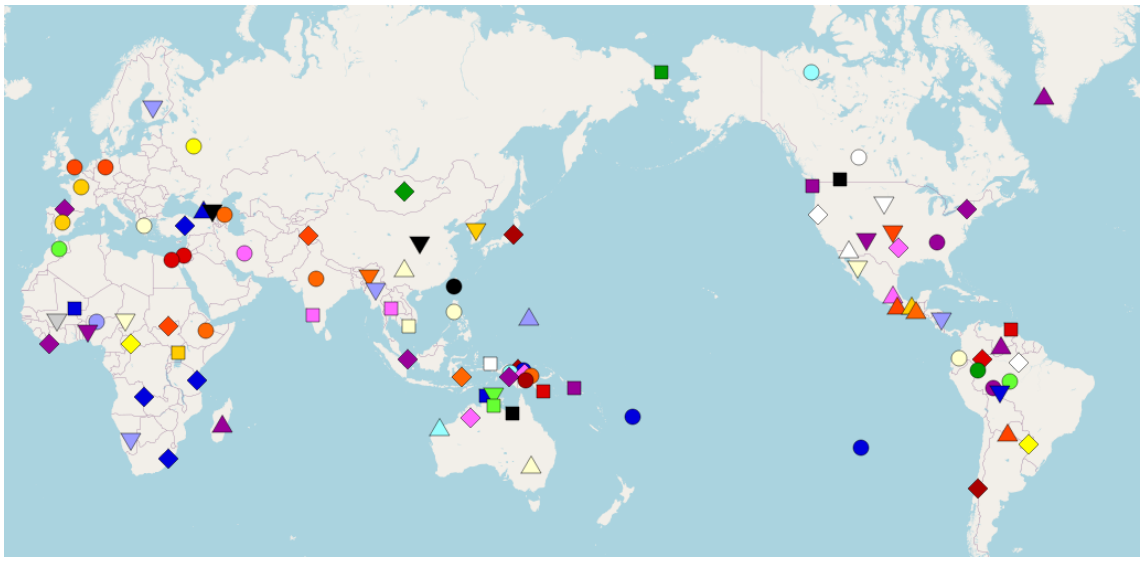

Figure 1: WALS - Map that shows the 100 Languages

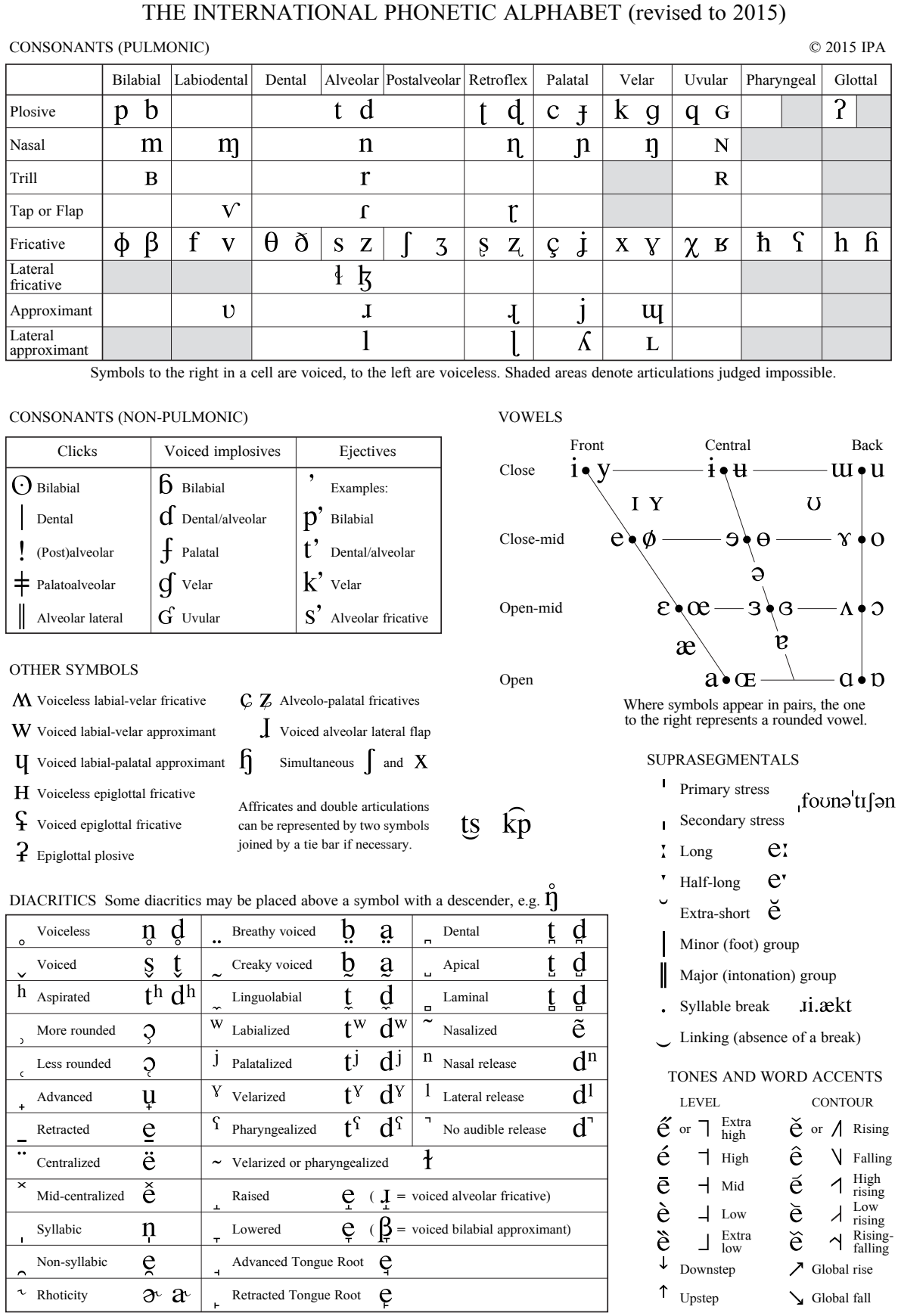

Figure 2: This is the full IPA chart, last updated in 2015

3 Technical Background

This chapter presents the technical background that is needed for this thesis. It explores everything around automated models that can be used to create phonetic transcriptions. I will first set the basis and dive into evaluation metrics and general architectures and frameworks that are commonly used and then present current state-of-the-art models. Table **??** shows the state-of-the-art models for grapheme-to-phoneme (G2P) modelling. Phonetic transcriptions are often used in ASR and **stt!** (**stt!**) technologies. There are systems that can produce speech directly from orthography and question the necessity of phonetic transcriptions. When only little data is available, the training data might not be enough to train an orthography-to-speech system, making the intermediate step of creating phonetic transcriptions necessary. Another reason for creating phonetic transcriptions is that its usage is not limited to speech applications [Mortensen et al., 2018]. They might also be used to compare languages on speech basis. In order to do that, there needs to be a lot of knowledge about how language works.

3.1 Evaluation metrics

In order to judge whether a computational system performs wells or not we need a appropriate metric. Mostly, you need a reference solution and a system solution that can be compared to the reference and then the difference can be quantified and be given a score. The evaluation of phonetic transcriptions depends on whether the system output and the reference are sentences or single words, i.e. character sequences. The most common metric to evaluate the former is the word error rate (WER). The lower the score, the better the model. If the WER is 0, this means that the texts are exactly the same. The idea behind the score is that we can capture the cost that it takes to transform the system output into the reference phoneme sequence. The following formula is used to calculate the WER:

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

In equation 3.1 the S stands for substitution, I for insertion, D for deletion and N denotes the total number of words in the reference sequence. Those numbers can be calculated by using an algorithm to get the edit distance. It is quite common to multiply the resulting number with 100 [Gorman et al., 2020]. Note that the WER can be more than 100%. This happens if, for example, the are a lot of additional insertions or deletions in the system text. If the system output and reference are character sequences, the score is called character error rate (CER). It is calculated in the exact same way as the WER, but instead of words everything is calculated on character basis. In the phonetic transcriptions setting, the CER is typically replaced by the phone error rate (PER) to match the correct terminology. The calculations are not changed. In a multilingual setting, it is sometimes necessary to have a score for the entire system covering more than one language. In such cases it is custom to use a macro-averaged WER or CER. This means that the sum of the scores for each language is divided by the number of languages [Leung, 24.6.2021].

In the case of G2P conversion, the WER actually just reflects the rate of wrongly predicted words, as one sequence consists of only one word. It is therefore just 1 (or 100) minus the accuracy. The accuracy is the rate of the correctly predicted words (which means we divide all correctly predicted words by the total number of words). It is important to note, the the WER and the PER suggest slightly different things. If each word has exactly one wrong character this means that the WER would be 100 as all words are predicted wrongly. Now, if the words are really long, the PER score would be very good compared to the really bad WER. If the words are rather short, the PER would get worse as well but the WER would just stay the same. So, when analysing the results it is important to pay close attention to how those two metrics relate.

3.2 Automated phonetic transcription

In this section I set the stage for understanding the technical background of G2P models. Today's technologies allow to build models that create phonetic transcriptions automatically given an original text. There are several approaches which I will discuss below. Creating phonetic transcriptions is essentially a sequence-to-sequence (seq2seq) task. Like other NLP tasks its goal is to transform a sequence of characters into another sequence of characters. In particular, it is very similar to machine translation which is such a task as well [Rao et al., 2015]. In the present case, the input sequence is a sequence of graphemes. These can look very differently depending on the script (see section 2.3). The output sequence is a sequence of

phonemes[1]. This process is typically referred to as grapheme-to-phoneme (G2P). There are a few problems and characteristics of the G2P task that are important:

- The input and output sequences are not always of the same length. It is difficult to align input and output which measn to map one or more grapheme to one or more phoneme. Not all systems rely on aligning input and output but often it is needed to analyse the results (for example to create confusion matrices).
- Due to the open-vocabulary situation and the impossibility to cover all possible words, all systems must be able to deal with rare and unseen words [Rao et al., 2015; Bisani and Ney, 2008].
- Most of the research done in this area is limited to the English language. This is not uncommon in NLP research. The availability of English data resources and the unavailability and struggles to find data in other languages heavily influences this research. Ashby et al. [2021] report that the SIGMORPHON (Special Interest Group on Computational Morphology and Phonology) G2P shared tasks in 2020 and 2021 are the first attempt to tackle multilingual G2P.

In the following, I explain the most important model types. Often, those different types are used in combination as the different models have different advantages which can be used very well in combination.

3.2.1 Look-up dictionary

The simplest version of a G2P model is a look-up table where a grapheme sequence is stored together with its phonetic transcription. Such a dictionary is expensive to create and needs a lot of storage and has a very limited coverage. Although such a system is no longer useful on its own it can still be used in addition to other, for example, statistical models [Bisani and Ney, 2008].

3.2.2 Rule-based models

The first systems to create phonetic transcriptions of text were rule-based systems. Although rule-based systems are outperformed by more recent neural models [Ashby

[1]Please refer to section 2.1 in order to understand the terminological implications of phoneme. As it is common in research, I will stick to the term *phoneme* although strictly speaking it is not always correct. Phoneme in this case just refers to any symbol that is used to represent a sound.

et al., 2021; Gorman et al., 2020], I will introduce them as they were an important step towards G2P modelling. Additionally, rule-based models can be used together with other models to reach a better performance. Rule-based transcriptions models are built using linguistic pronunciation rules. For example, if in a language, a certain word initial grapheme is always pronounced the same way, we can store a rule, that states that if that grapheme is encountered word initial it should be transcribe using a specific phoneme. This rule must of course be stored in a machine readable way but the idea is the same. In order to be able to create such a system, one needs to collect pronunciation rules first. While there are only few languages where such rules are ready and available for the general public there are many languages where those rules need to be created first. In order to create the rules in the first place, a lot of linguistic expertise is needed. Apart from this initial effort to create the rules, a problem with rule-based approaches is the maintenance of the systems. To maintain the system, experts need to keep track of language change which is time consuming and expensive. Most languages are irregular in their pronunciation and those irregularities need to be tracked as well. Another drawback of such systems is that they might fail when presented with unseen or rare words [Bisani and Ney, 2008]. Many earlier systems published considered only one language and were not multilingual (see e.g. Toma and Munteanu [2009]).

Epitran Epitran is an example of a relatively new rule-based system. It makes use of the fact that there are languages that are more or less regularly pronounced and presents a rule-based system for G2P conversion for mostly low-resource languages. The system has the ability to provide a solution for every possible word and is consistent within its transcriptions. Epitran for all languages except English and traditional Chinese works with a map file that allows to map graphemes to phonemes. Additional pre- or post-processing can be applied that follow context specific rules [Mortensen et al., 2018].

3.2.3 N-gram Models / Statistical models

N-gram models, statistical models or joint-sequence models were used before neural models took over the field. These are sometimes referred to as traditional models. One reason why they were outperformed by neural models is that it is necessary to construct alignments between graphemes and phonemes. This is needed because one grapheme can be realized as multiple phonemes and vice versa. It is not possible to simply have a one-to-one alignment. Joint-sequence models were often used with different versions of the Expectation-Maximization (EM) algorithm [Lo and Nicolai,

2021]. The main intuition about those models is that they, in some way or other, try to statistically model the relationship between graphemes and phonemes. Typically, those models consist of two parts: first an alignment model. Second, a model that captures the relationship between graphemes and phonemes using clearly defined statistical methods. A very common model is the joint-sequence model.

Joint-sequence model The term *joint-sequence* hints already at the underlying architecture of those models: the idea is to process *joint* sequences of input and output symbols. In order to do that they must be aligned. We can then concatenate those alignments, which means to concatenate the grapheme and the phoneme which it is mapped to, and receive what is called a graphone. Using the concept of finite-state transducer (FST), we can build a model. FSTs are similar to finite-state automatons. But instead of just telling whether a certain sequence belongs to a certain language (which is pattern matching), they can output none or many symbols at every step as well. This means that in the process of iterating over the sequence, they produce another sequence. Knowing that, it is easily understandable that this works well for our given G2P task. The idea is now, that we model the joint-probability of the input graphemes and output phonemes or rather our graphones. Doing this we can use the EM algorithm to find a mapping of graphemes to phonemes that is most probable. As those graphones consist of n-grams of both the input and output sequence, they themselves can be considered a n-grams which is why those models are sometimes referred to as n-gram models [Bisani and Ney, 2008; Lo and Nicolai, 2021].

3.2.4 Neural models

Neural G2P models have been reported to outperform most other models [Lee et al., 2020]. Many researchers experiment with different variants of LSTM models [Lee et al., 2020; Hammond, 2021; Gautam et al., 2021; Rao et al., 2015]. But there are also other models that have been used. The most important of those will be introduced in the following.

Sequence-to-sequence seq2seq is not a model type as such but rather an architecture. seq2seq models include an encoder and a decoder or more than one of each depending on the exact implementation. Both encoder and decoder can be the same model type or different ones but all of them are reccurent neural networks (RNNs) or some variant of it. The output of the (last) encoder RNN is used as input for

the decoder RNN. What makes such a model architecture powerful is that they can map an input sequence to a output sequence of a different length and different type. Generally, there is a difference between models that assume conditional independence between each output step (e.g. Hidden Markov Models) and there are models that do not make this assumption but condition the current output on the entire sequence before. Depending on what model type is used as encoder or decoder considering the entire preceding input can get tricky if the input sequence is really long. This is true especially for RNN models. Apart form that, seq2seq models have to wait until the full input sequence is processed before they can start decoding. This does not work well for input that gets continuously longer [Kostadinov, 02.05.2019; Sutskever et al., 2014].

RNN The important intuition about RNNs is that they process each unit of an input sequence one unit at a time. In our case, we can think of one unit as one grapheme in the input sequence. Instead of only outputting something at each time step, a hidden representation is passed on to be processed in the next time step. This sequential processing is crucial as it means that each unit gets information about the units preceding it. This makes sure that information about the preceding context is included. The output of such an RNN is therefore a manipulated version of the input sequence of the same length. A special kind of RNN is the LSTM. LSTM means long short-term memory. As their name suggests they include what we could call a memory. Instead of just adding some representation of the preceding units at each time step, LSTMs can more flexibly decide what information is added and what information should be forgotten [Olah, 29.01.2022; Kostadinov, 12.02.2017]. Many earlier neural LSTM models use a connectionist temporal classification layer to include alignment information [Lo and Nicolai, 2021]. This is a special methodology that can deal with n-to-m alignments which is the case in G2P modelling.

Transformer Transformers are seq2seq models as well. A central concept and what makes transformers powerful models is their self-attention mechanism. Also, they can process the entire input sequence in one go such that each unit is processed by a separate part of the model instead of just using the same cell for every unit. This allows to have dependencies between the different input units which means that we can model any dependencies within a sentence or a grapheme sequence. This is the reason why we call it *self*-attention as the attention is focused on other parts of the input sequence, but still on the sequence itself [Alammar, 03.01.2022].

Neural Transducer Neural transducers, as presented by Jaitly et al. [2016], extend previously used seq2seq models. They can treat more arriving input without having to redo the entire calculation for the entire updated sequence. At each time step, the neural transducer can output zero to many output symbols.

ISO396-3	BS20						DeepSPIN20		IMS20		BS21	CL21	UBC21		DP21
	LSTM		transformer		pair n-gram							CL	UBC-1	UBC-2	
	WER	PER	WER	PER	WER	PER	WER	PER	WER	PER	WER	WER	WER	WER	
ady^{B}	28.00	*6.53*	28.44	*6.49*	32.00	*7.56*	24.67[3]		25.00	*5.79*	**22.00**	**22.00**[23]	25.00	**22.00**	
bul^{A}	31.11	*5.94*	34.00	*7.89*	41.33	*9.05*	-		22.22	*4.85*	**18.30**	18.80[6]			
cym (wel_sw)^{B}											**10.00**	**10.00**[1]	13.00	12.00	
ell (gre)^{B}	18.89	*3.30*	18.89	*3.06*	21.78	*4.05*	-		**18.67**	*2.97*	21.00	20.00[13]	22.00	22.00	
eng(_us)											41.94				**37.43**
fra (fre)^{A}	6.22	*1.32*	6.89	*1.72*	13.56	*3.12*	**5.11**[3]		6.89	*1.60*	8.50	7.50[456]			
hbs^{A}											**32.10**	35.3[7]			
hin	6.67	*1.47*	9.56	*2.40*	12.67	*4.05*	-		**5.11**	*1.20*					
hun^{A}	5.33	*1.18*	5.33	*1.28*	6.67	*1.51*	-		5.11	*1.12*	1.80	**1.00**[67]			
hye (arm_e)^{A}	14.67	*3.49*	14.22	*3.29*	18.00	*3.90*	-		12.67	*2.94*	7.00	**6.40**[7]			
ice^{B}	10.00	*2.36*	10.22	*2.21*	17.56	*3.62*	-		**9.33**	*2.04*	12.00	10.00[13]	13.00	11.00	
ita^{B}											**19.00**	31.00[3]	20.00	22.00	
jpn(_hira)^{A}	7.56	*1.79*	7.33	*1.86*	9.56	*2.07*	**4.89**[4]		5.33	*1.26*	5.20	5.00[7]			
kat (geo)^{A}	26.44	*5.14*	28.00	*5.43*	37.78	*6.48*	-		24.89	*4.57*	**0.00**	**0.00**[4567]			
khm^{B}											34.00	32.00[13]	31.00	**28.00**	
kor^{A}	46.89	*16.78*	43.78	*17.50*	52.22	*15.88*	24.00[13]		26.22	*4.38*	16.30	**16.20**[4]			
lav^{B}											55.00	**49.00**[23]	58.00	**49.00**	
lit	**19.11**	*3.55*	20.67	*3.65*	23.11	*4.43*	-		20.00	*3.63*					
mlt(_ltn)^{B}											19.00	12.00[1]	19.00	18.00	
nld (dut)^{A}	16.44	*2.94*	15.78	*2.89*	23.78	*3.97*	-		**13.56**	*2.36*	14.70	14.70[7]			
rum^{B}	10.67	*2.53*	12.00	*3.62*	11.56	*3.55*	**9.78**[3]		10.22	*2.23*	10.00	12.00[3]	14.00	10.00	
slv^{B}											49.00	50.00[1]	56.00	**47.00**	
vie^{A}	4.67	*1.52*	7.56	*2.27*	8.44	*1.79*	**0.89**[2]		1.56	*0.48*	2.50	2.00[57]			

Table 2: The table lists the SOTA models from the SIGMORPHON tasks in 2020 and 2021. Superscript: model numbers. Numbers in bold: best WER. Languages in bold are in the 100LC corpus. ^{A}: 10,000 training samples in 2021. ^{B}: 800 training samples in 2021. Model explanations: table **??**.

Model name	Author	Model Architecture
CL21	SIG21: Clematide and Makarov [2021]	These are seven models in total. LSTM-based neural transducer with pointer network-like monotonic hard attention trained with imitation learning. All models 1-7 are majority-vote ensembles with different number of models (5-30) and different inputs (characters or segments).
UBC21	SIG21: Lo and Nicolai [2021]	UBC-2 outperforms the baseline. They analysed the errors of the baseline and extend it by adding penalties for wrong vowels and wrong diacritics. Errors on vowels actually decreased. UBC-2 achieved the best macro average for low-resource languages in 2021. UBC-1 included syllable prediction which did not improve the results.
DP21	SIG21: Gautam et al. [2021]	Dialpad-1: Majority-vote ensemble consisting of three different public models (weighted FST, joint-sequence model trained with EM and a neural seq2seq), two seq2seq variants (LSTM and transformer) and two baseline variations.
BS21	Ashby et al. [2021]	The baseline in 2021 is a neural transducer trained with imitation learning similar to a model submitted in 2020 [Makarov and Clematide, 2020]. As they are so similar, I did not include the model twice, but only the newer one.
DeepSPIN20	SIG20: Peters and Martins [2020]	DeepSPIN-2,-3,-4: Transformer- or LSTM-based seq2seq models with sparse attention. Add language embedding to encoder or decoder states instead of language token.
IMS20	SIG20: Yu et al. [2020]	IMS: Self training ensemble of one n-gram-based FST and three seq2seq (vanilla with attention, hard monotonic attention with pointer, hybrid of hard monotonic attention and tagging model).
BS20	Gorman et al. [2020]	The baseline for the task in 2020 consisted of three different model types. An LSTM, an transformer and a pair-n-gram model that is based on a weighted FST. All of them were outperformed by other models except for Lithuanian. The LSTM performed best among these three.

Table 3: This table presents the state-of-the-art G2P models. The results for these models can be found in table 2

3.3 State-of-the-art G2P models

As I have to decide what model I will use to train on my language set, I will now have a look at the state-of-the-art models that can be used for G2P conversion. The Special Interest Group on Computational Morphology and Phonology (SIGMORPHON) [Sigmorphon, 2021] regularly organizes shared tasks concerned with morphology and phonology. For the years 2020 and 2021 they organized a G2P conversion task [Ashby et al., 2021; Gorman et al., 2020]. The tasks represent a first attempt at creating benchmarks for multilingual G2P conversion. Although

there is other research on G2P, many recent publications have been made within the SIGMORPHON shared tasks. In the next sections, I will summarize the results and insights from G2P research with a focus on those shared tasks.

3.3.1 Model architectures

An essential part of G2P modelling is the actual model. In section 3.2, I explained the most important theoretical basics. For this part here, I familiarized myself a bit more with strategies that work well in practice and what some concrete problems are. The methodologies mentioned below are to the most part task-agnostic. This means that they often improve results on most NLP tasks and are not specifically developed for the G2P task. Still, I think it is insightful to be aware of the great variety of approaches that nowadays technology offers.

Ensembles Many models that are used and achieve peek performance for G2P modelling are ensemble models. An ensemble is essentially just a pool of different models that are trained on the data with different settings or they are completely different models. The way such a model can be used for inference is that all of the models process the input and present their predicted results. Out of all possibilities, one prediction will be chosen that is then the final model output. In order to get the final output, an ensemble needs some kind of decision algorithm to output the best result. A disadvantage of ensembles is that the models need a lot of storage. Also, it is to some extent a bit of a *brute-force* approach as it could lead to preferring quantity over quality.

Learning edit actions Instead of learning the output phoneme sequence, a model can also learn how to edit the input sequence in order to get the output sequence. Such a model would then output an edit sequence which can be applied to the input sequence in order to obtain the final phoneme output. The model therefore learns to create sequences of edit actions. The problem with this approach is that there are many possible sequences of edit actions that produce the same result. For example, it is always correct to delete every unit in the input sequence and then insert every unit from the output sequence. But this does not tell us anything about how graphemes and phonemes relate. To this end, we would also want to use substitution actions to see whether one grapheme is always substituted by the same phoneme. Imitation learning is proposed as a solution for this problem. Easily put, imitation learning is a variant of reinforcement learning. The idea is that the model learns

to imitate the behaviour of an expert (for example, a human expert that provides correct samples of the task in question) [Ai, 19.9.2019].

Multi-task learning What has worked well for G2P models is to use multi-task learning. This means that the model is not only trained on one task but on multiple tasks that are related. In the present case, a model was trained on phoneme-to-grapheme conversion as well [Gorman et al., 2020].

Neural models Not surprisingly, models that achieve peek performance are almost exclusively neural models [Gorman et al., 2020]. Due to their ability to process increasingly longer sequences and the above discussed techniques like attention, the are ideal for almost all NLP tasks. What type of neural model is chosen also depends on the amount of data available. Transformers are suggested to work better for larger datasets, while they are outperformed by LSTMs on medium-size datasets (a few thousand training pairs) [Gorman et al., 2020].

Reduce vocabulary size Some syllabary languages like Korean allow the decomposition into smaller units that make up the signs. Many other languages that do not use the Latin alphabet allow to be written with Latin letters. If a reduction of the vocabulary size is possible in one of these ways, it almost always improves performance as smaller vocabulary sizes are easier to handle for models [Gorman et al., 2020].

3.3.2 Data manipulation

A model is only as good as the data that is used to train it. While this is a very basic paradigm, in reality assuring data quality is not always easy. In this section I list a few strategies that are used to preprocess and prepare G2P data and how to deal with too little available data.

Data quality Authors mostly include a section about their preprocessing and what should be done to ensure high quality datasets. The list given below is an incomplete list of potential problems and measures taken in different settings for G2P data:

- **Exclusion of words with less than two Unicode characters or less than two phone segments** [Ashby et al., 2021]

- **Separation by script** [Ashby et al., 2021]: It is very straightforward why this is done. There is no obvious connection between the different scripts of a language and its pronunciation. It makes sense to treat different scripts as different languages.
- **Exclude foreign words with foreign pronunciations** [Ashby et al., 2021]: Foreign words in a language with their original pronunciation can add phonemes that are not in that language's phoneme inventory. If they were to be included it would make sense to include a pronunciation adapted to the actual language.
- **Words with multiple pronunciations in word lists**: Ashby et al. [2021] excluded those words, however, it might also be possible to add Part-of-Speech (PoS) tags or other linguistic information to distinguish these words.
- **Consistent broad transcriptions** [Ashby et al., 2021]: With broad transcriptions it is important to be consistent and not use allophones. Ashby et al. [2021] did this specifically for Bulgarian.
- **Linguistic variation and processes** [Ashby et al., 2021]: Some transcriptions include examples for monophthongization or deletion which are ongoing linguistic processes but should not be part of a dataset representing a standard variation. Ashby et al. [2021] dealt with monophthongization by choosing the longer to two transcriptions as this logically exclude the monophthonged version. This does of course only work if there are more than one pronunciations available.
- **Tie bars**: Ashby et al. [2021] notice that some languages (English and Bulgarian) have inconsistent use of tie bars. This can be correct by replacing all inconsistencies by the tie-bar-version.
- **Errors in the transcriptions**: Gautam et al. [2021] noticed many errors in the WikiPron English data. They identified errors by looking at the least frequent phones and then check the word-pronunciation pairs where those phones occurred in. As the number of phones in a language is often known this can be used to check the phones in the datasets and identify uncommon ones.

Especially the task of finding errors in the transcriptions is quite tricky. It requires a lot of knowledge about the phonology and phonetics of a specific language.

Low-resource setting Apart from a few well-studied examples, for most languages there is only little data available. It is therefore highly interesting and

important to find solutions of how to deal with lack of data. Hammond [2021] submitted a system to the 2021 SIGMORPHON edition focusing on data augmentation methods. The primary goal of their approach was to test how successful a minimalist data augmentation model would be, knowing it would most probably not outperform any of the other models. They identified two approaches that might improve low-resource models. The first one is to use as much as possible of the development set for training. The second, to train all languages together differentiating the languages only by a tag added to the word representations. The model they used was purposefully a very simple model that does not use a lot of resources. They used a seq2seq neural net with a LSTM decoder and encoder. Both LSTMs have two levels.

Yu et al. [2020] propose a data augmentation model for low-resource settings. The methodology applied in their approach is ensemble learning combined with a self-learning strategy. They use their ensemble to make predictions on unlabelled data. This newly created data is then added to the training data and the models are trained for another epoch on the extended data. This strategy worked well and produced good results.

Results in a low-resource setting are still bad when only using 800 samples for training. More research needs to be done in data augmentation techniques and improving the systems to cope with only little available data.

3.3.3 Error and result analysis

In this section I will list different types of analysis that have been performed on a trained model to improve future research.

Broad and narrow transcriptions In the SIGMORPHON tasks, there are great differences in the performance of models for different languages. One possible explanation is that the datasets were a mix between broad and narrow transcriptions. As narrow transcriptions contain much more detail, it can be argued that this is more difficult for any system [Ashby et al., 2021]. This assumption still needs to be analysed more closely. This differing performance for various languages calls for the questions what makes a language hard to pronounce. Especially as for Georgian all models from the SIGMORPHON task reached a WER of 0.0. Interestingly enough, the WER for the language in the high-resource setting, English, reached one of the highest WERs.

Linguistic error analysis Lo and Nicolai [2021] chose to perform an error analysis and try to minimize the frequent errors of a model in a multilingual low-resource setting. The analysis showed that often the model gets vowels and diacritics wrong. They extended the model in a way such that wrong vowel and diacritics predictions are punished more than other errors. Compared with the unchanged model, this extended model reached a better performance for some languages. The predictions with their model shows an improvement in vowel prediction. A further analysis showed that many errors still happen with vowels. Vowels get often confused with similar vowels. Their conclusion is that many of these errors make sense in a linguistic sense. They also tested augmenting the input data with syllable boundaries which did not improve the results.

Another type of linguistic analysis that can be performed is to analyse the data and check for uncommon pronunciations or language internal ambiguities. If a model produces a lot of wrong output because of ambiguities or uncommon data, then this is not necessarily the models fault but just a language inherent inconsistency. As languages are generally ambiguous, this type of analysis is very insightful to find out about *real* errors of the model. These are errors that could be derived from the data, but the model did not get there. Ashby et al. [2021] did such an analysis for the SIGMORPHON 2021 task which showed that many errors are due to language internal ambiguities.

Include linguistic information What has been suggested by Gorman et al. [2020], is to make use of phonetic resources or rule-based systems to improve the quality of current models. The advantage of such an approach is that it is specifically tailored to the problem at hand and not at all task-agnostic. Makarov and Clematide [2020] confirm this suggestion as they performed an error analysis which showed that including linguistic information such as PoS-tags might be useful.

As is always the case with such research there are many different aspects that can be tuned in order to improve model results. For my thesis I will have a closer look at how we can use phonetic features to improve models.

3.4 CMUSphinx

For my personal experiments, I decided to use the CMUSphinx seq2seq G2P model. This model has been used in the SIGMORPHON task. It was not used on many languages but promised a good performance which is why I decided to use this

model for this present thesis. the CMUSphinx model is a transformer-based acas2s model implemented with tensorflow. There exists a pre-trained version of the model for acg2p, however, they use a transcription format other than IPA which means it cannot be used for our dataset [GitHub, 03.02.2022].

3.5 Unicode and the International Phonetic Alphabet

When it comes to representing characters in a machine-readable format things get very tricky, very quickly. In order to understand this fundamental problem it is necessary to understand the basic concept behind unicode and encodings in general. Moran and Cysouw [2018] present a neat overview in their book. As discussed in chapter 2, there are many different kinds of what we typically call letters, graphemes, characters or signs[2]. Just as a human writer must be able to uniquely identify each different graphemes, so must a computer. The most widely spread standard to represent scripts is called Unicode. Graphemes are mapped to unique numbers that can be rendered differently depending on the font and the context. There are different stages of representation until a graphemes can be represented on screen:

CODE POINT A unique numerical, non-negative value usually expressed as a hexadecimal number (U+0000). Allows one-to-one mapping between letters and codes. Each code point has a set of properties attributed to it. Properties like the script, uppercase or not, etc.

CHARACTER An abstract representation of the shape of the grapheme. Can in theory not be represented visually, as this includes a font. A Unicode character is *not* the same as what we would call a grapheme in different writing systems.

GLYPH The rendered and therefore visual representation of one or more Unicode characters that can be identified by its code point(s). A glyph is rendered in a specific font in a specific context. No matter how different it looks to the user, for Unicode all different representations of one code point are exactly the same. Sometimes one character is represented as two glyphs. It is important to note here, that the exact visual representation of a glyph is not at all defined by its code point. This means that the exact same glyph can represent more than one code points. This happens sometimes in the IPA. An example is the post-alveolar click:

[2]Please note that I will from now on use grapheme to denote the smallest meaningful element of any writing system. Grapheme does not imply any specific writing system nor does is take the Unicode background into consideration. If I wish to distinguish the Unicode specifications I will use the correct Unicode term as described in this section.

- ! : this glyph represents an exclamation mark with unicode code point U+0021.
- ! : this glyph represents an post-alveolar click with unicode code point U+01C3.

It is striking that these two look exactly the same. Things like this become important when, for example, I want to count the different characters in a text.

Unicode code points are often organized in blocks. A block can, for example, contain all letters of the Latin script. Those blocks are helpful although not always consistent. The IPA is represented in a basic block but many IPA symbols are actually found in other blocks. Confusion often arises from the fact, that one human-perceived grapheme is sometimes represented as more than one code point.

Grapheme clusters A grapheme cluster is one visual letter that is represented in Unicode as more than one code point. This is the case for diacritic marks. A problem with grapheme clusters is that some diacritic marks, so marks that cannot really exists without any base character are underspecified. This means that when we want to split into clusters, we do not know if that character belongs to the left or the right base character. This is the case for many characters in the IPA. Unicode does not specify these but leaves it to the user to create tailored grapheme clusters.

Precomposed characters Note that sometimes, grapheme clusters can be precomposed and the combination of those two or more characters is assigned a new number. These clusters can be problematic if in a specific context, the graphemes should not be clustered but read separately. An example is the German 'ä'.

Additional complexity is added through the possibility of Unicode to create Unicode locales. These allow users to specify language- or writing-system-specific cases. An additional challenge is that of picking the right font. Our standard font format can only contain about half of all the Unicode code point. It is therefore simply not possible to display the entire set of Unicode characters with one font. Many problems encountered with displaying writing systems are somehow connected to the font rather than Unicode itself [Moran and Cysouw, 2018]. Moran and Cysouw [2018] list a few more 'pitfalls' that one might encounter when dealing with Unicode.

For the present thesis, this topic is relevant for multiple reasons:

1. The IPA contains many special characters and many diacritics.

2. The language data is available in many different scripts.

It is crucial that all data files, be it phonetic or 'normal' scripts, are formatted and read correctly. Or rather that the encoding and processing is made transparent as often there is not one correct way of how to treat IPA characters.

Segments library Moran and Cysouw [2018] present a python library called segments[3] that can be used to process phonetic text. It includes a method to segment IPA strings into grapheme clusters that make sense in a IPA context. This means that, for example, diacritics are put on the base character but also that characters connected by a tie bar are displayed as one unit. I will make use of this library later when processing the data.

3.5.1 Unicode normalization forms

The above explanations make clear that there are considerable differences in what a human reader perceives and in what happens in the background. Unicode therefore provides normalization forms that can help to process written data. Unicode publishes extensive explanations along with their standards which also includes those normalization forms. I will therefore not explain everything in full detail as this is done so already online [Inc., 27.08.2021]. What is important is that each normalization form results in very different behaviour if a text is processes. There are two important aspects to normalization. One is that we can have decomposed or composed characters. The second is that we have a compatibility form and a non-compatibility form. In a decomposed string, we split the characters into their individual components. This means that characters with diacritics are split up into two or more characters. This means that a characters that had originally assigned one code point can in a decomposed form have more than one code points. In a composed normalization usually precomposed forms are kept. This means that some parts can be similar or the same to the decomposed version but if for a character there exists a precomposed version, this one is usually used. If a normalization is according to compatibility decomposition, this means that any formatting is removed such that we receive the underlying character in its original form. Superscript characters are then shown normal. How exactly these normal forms work is not always equally important, but what is absolutely crucial to make sure that when characters are compared or counted, the same normalization form is used. The names for these normalization form are as follows:

[3] `https://pypi.org/project/segments/`

NFD : Canonical Decomposition

NFC : Canonical Decomposition, followed by Canonical composition

NFKD : Compatibility Decomposition

NFKC : Compatibility Decomposition, followed by Canonical Composition

4 Experiments: Data Collection

The first practical part of this thesis is concerned with data collection. Although phonetics is an important sub-area in linguistics, phonetic transcriptions are hard to find. If there are any transcriptions available, there are various hindrances that prevent it from being used as is. In the following, I outline the different data types which are available and the different strategies that are used to convert the data into one well-formatted corpus. There is quite a famous set of phonetic texts which is a collection of short stories called *The North Wind and the Sun*. Those stories were the starting point for my search for data. The reason for this is that the short stories are available in many different languages. As the corpus this thesis is based on is a large multilingual corpus, the availability of data in many languages was one of the key criteria to look choose the datasets.

4.0.1 Transcription Sources & Formats

Phonetic transcriptions of various languages are available from different sources in different formats. From what I have found out in my research phonetic transcriptions are available as either full texts or a word lists.

Full Text For the task of G2P conversion, phonetic transcriptions in the form of fully transcribed texts would be ideal. As became clear, it is hardly possible to find those. There is plenty of material describing how different languages can be transcribed but those rarely contain fully transcribed text. If they do, it is mostly limited to one or a few sentences. Additionally, some texts include short descriptions where certain pronunciations rules are explained which are not included in the transcriptions (especially stress). Apart from NWS short texts described in section, there are no full texts available in multiple languages that could be used.

Pronunciation Dictionaries Another data type that is found quite often are word lists. Those are sometimes referred to as pronunciation dictionaries. However,

these often mean that there are words mapped to an audio representation which is not what is meant in this present case. Pronunciation dictionary in this present case refers to the mapping of an orthographic word to its pronunciation using phonetic symbols. Although such lists are very handy, especially as they can easily be used to train a transcription model, transcriptions of individual words and of entire texts are not exactly the same. There are two major problems:

- Pronunciation depends on the context of the word in question. Word forms are ambiguous and sometimes their pronunciation differs given on their specific context. add example
- Phonetic boundaries are not always equivalent with word boundaries. Spoken language sometimes merges certain words which leads to one phonetic unit. There are phonetic symbols to represent such merging which often happens in, for example, French.

Transcription conventions No matter how the transcriptions are formatted, a crucial factor is the transcription convention. The IPA is a well-known convention but there is a lot of data that is available using a different convention. Although it might be possible to convert one convention into another convention this is very tedious and often those conventions are made specifically for one language or a group of related languages.

4.1 Data used for this thesis

Based on the outcome of my research on the availability of phonetic data, I decided to use two datasets for my experiments in this thesis. I will introduce both of them below.

4.1.1 WikiPron

A very recent project that publishes pronunciation lists is WikiPron. The WikiPron project [Lee et al., 2020] is an open-source Python mining tool to retrieve pronunciation data from Wiktionary. Their database contains 1.7 million word/pronunciation pairs in 165 languages. Both, the database and the tool, are freely available online. Apart from the mining tool and the database, WikiPron can be used for grapheme-to-phoneme modelling. The WikiPron data for one language is always

Iso 639-3	Language name	Type WikiPron	# Words WikiPron	Type NWS
cmn	Chinese	broad	133,686	unk
deu	German	broad	34,145	broad
deu	German	narrow	10,984	narrow
ell	Greek (Modern)	broad	10,547	unk
eng	English US	broad	57,230	broad
eng	English US	narrow	1,633	narrow
eng	English UK	broad	60,422	-
eng	English UK	narrow	1,284	-
eus	Basque (Goizueta)	broad	1,742	broad
fin	Finnish	broad	69,015	-
fin	Finnish	narrow	69,008	-
fra	French	broad	56,911	unk
hin	Hindi	narrow	9,563	-
hin	Hindi	broad	10,812	unk
ind	Indonesian	broad	1,555	unk
ind	Indonesian	narrow	2,637	-
jpn	Japanese (Hiragana)	narrow	19,689	-
kat	Georgian	broad	15,123	broad
kor	Korean	narrow	14,141	unk
mya	Burmese	broad	4,631	broad
rus	Russian	narrow	402,586	unk
spa	Spanish (Castilian)	broad	60,677	broad
spa	Spanish (Castilian)	narrow	52,190	narrow
spa	Spanish (Latin America)	broad	48,649	-
spa	Spanish (Latin America)	narrow	41,845	-
tgl	Tagalog	broad	3,321	-
tgl	Tagalog	narrow	1,915	-
tha	Thai	broad	15,050	unk
tur	Turkish	broad	1,789	unk
tur	Turkish	narrow	1,812	-
vie	Vietnamese	narrow	15,240	unk
zul	Zulu	broad	1,677	-

Table 4: In this table I list all languages that I am using for my experiments.

structured the same. It is a tsv-file that has graphemes as a first column and corresponding phonemes as the second column. While the graphemes are just listed as-is, the phonemes are split using the segments library which I shortly presented in

section ??. The phoneme segments are separated by white spaces. The WikiPron data has already been used in shared tasks which means that results on this data can easily be compared to other results. For a shared task in 2021 organized by SIGMORPHON, the WikiPron data was improved and additional scripts were added based on feedback and findings from a similar task in 2020. One major improvement was concerned with languages written in different scripts. WikiPron supports now the detection of different scripts and languages can be sorted according to those scripts. For some languages, there is a filtered version of either the broad or the narrow transcription available. Whenever possible, I used this one.

4.1.2 The North Wind and the Sun

Quite a well-known phonetic corpus is a collection of short stories. The Journal of the International Phonetic Association (JIPA) continuously published different phonetic transcriptions of a short story called *The North Wind and the Sun*. The story is a fable said to be written by Aesop and has been translated into many languages. Additionally, for many languages there exits a phonetic transcription. As I have mentioned before, many of these texts are available already transcribed and free to use [GitHub, 04.02.2022]. Table ?? shows the languages for which the short story is available and which are also in the 100 language corpus. As I have shortly hinted at in chapter 2, Baird et al. [2021] performed an analysis on these texts to find out how many tokens we need to cover a phonetic inventory of a language adequately. What they have found out in their study is that those short stories are by far not long enough to give a good picture of the phonetics of a language. This is of course problematic if those texts are used to demonstrate how a language works phonetically. That said, I will use these stories with care and only as an additional dataset to have more variety.

4.1.2.1 Transcription of NWS stories

A collection of the NWS stories is available in a handbook of the JIPA which is only available as a pdf scan of the original book [Press, 2010]. Luckily, most of those texts have been transcribed and made available by Simon Greenhill [GitHub, 04.02.2022]. At least the phonetic part of it, the original version was still not available already transcribed. Before I knew about the availability of these texts, I did some research on Optical Character Recognition (OCR) for IPA texts and manual transcription. While OCR is technically possible it turns out to be very difficult for IPA characters. There are tools that include IPA character recognition like the ABBYY FineReader

Iso 639-3	Type	Variation	Language
arn	broad and narrow		Mapudungun
cmn		Pekinese	Mandarin Chinese
deu	broad and narrow	North German	German
ell			Modern Greek
eng	broad and narrow	Americsn	English
eus	broad and narrow	Goizueta	Basque
hau	narrow		Hausa
heb			Modern Hebrew
hin	narrow		Hindi
ind			Indonesian
kat	broad and narrow		Georgian
kor			Korean
mya			Burmese
pes			Western Farsi
spa	broad and narrow	Castilian	Spanish
tha			Thai
tur	broad	Istanbul	Turkish

Table 5: The table shows a list of all the short stories *The North Wind and the Sun* that are available as phonetic text and whose languages are in the 100 language corpus.

which can be acquired for a fee. The CL institute at the UZH owns a version of the ABBYY tool but this version does not include the IPA module. I ran the ABBYY version on a JIPA pdf containing said phonetic transcriptions but the result could not be used. Mostly diacritics and special phonetic symbols were not correctly transcribed. These are exactly those characters that make it difficult to transcribe IPA manually, so there is not point in using this tool. There are also open source tools. One of which is called tesseract. tesseract does not include the IPA alphabet. It is possible to train the model to include the IPA alphabet but this would need appropriate training data[1].

As it was not possible to use OCR to get the texts, a next approach is to transcribe them. I experimented with a software called Transkribus[2] to manually transcribe the pdf scans. The software allows to make use of neural Handwritten Text Recognition

[1] `https://github.com/tesseract-ocr/tesstrain`

[2] https://readcoop.eu/de/transkribus/

(HTR) models. There exists no pre-trained model for transcribing IPA charachters, but I trained my own while transcribing some of the documents. On the website they mention that, ideally, training needs 5,000 - 10,000 words already transcribed. Although my available data is not nearly enough to train a reliable model (the short stories have around 40 - 100 tokens), it was a great help to transcribe. As the scans where not handwritten but machine typed text, the model still reached a surprisingly good quality. As an example: For the Hebrew transcription, the model reached a WER of 34.52 and a CER of 6.11. The two main mistakes were made for two characters that were not even in the training data. The quality of the scans differed quite a lot which had an influence on the performance of the model as well. After transcribing another document I trained the model again and transcribed a few more documents. The transcriptions got continuously better such that in the end I did not take me nearly as much time as in the beginning. Most of the errors resulted from characters that had not been in the previously described documents.

It was interesting to see that it is relatively easy to transcribe IPA text when using the right software. The original texts were easier to transcribe as they do not contain as many diacritics or other special characters. Still, for some texts like Chinese or Hindi, I needed help from native speakers as it is nearly impossible to find the matching sign if the language is unknown.

4.2 Pronunciation dictionary coverage

As I am using two different datasets I want to find out how these two datasets relate. I order to do that , I am using the WikiPron lists to write the NWS stories. In a way, I used the WikiPron data as a look-up table to create phonetic transcriptions for the NWS stories. By comparing the text produced by the WikiPron data to the reference transcriptions form the JIPA articles, I could calculate the coverage, the WER and the PER score. The comparison and the analysis of the metrics gives the following insights. In addition, I manually checked some errors which gave me insights about the word lists in general:

- For some NWS transcriptions it is not clear whether their transcription is narrow or broad. On the other hand, sometimes there is no broad or narrow word list available for a specific language but only one of those. For the short stories where the type was unclear, I tried both word list types if those are available. The analysis shows that transcriptions written with broad lists give better results if the type of the reference is unclear. See, for example,

Indonesian (ind) or Hindi (hin) in table 5. As this is the case, it makes sense to treat the unknown transcriptions as broad.

- The IPA allows to transcribe intonation segments. In German, those correspond mostly to punctuation marks like end of sentence symbols or commas. But this must not be true for every case. It needs to be decided if those should be kept or potentially deleted.
- The pronunciation dictionaries sometimes included duplicates with different pronunciations. This is not surprising but still it needs to be handled well. A solution is to simply delete duplicate words.
- In order to do this very simple experiment, it is necessary to tokenize the texts. This works well for languages using the Latin script. For languages like Chinese or Korean this is more difficult to accomplish. However, this issue needs to be tackled to create G2P models anyway. I will therefore not explore this issue here.
- Interestingly enough, for some language the coverage is really low although there are quite a lot of words in the word list. This is the case for German (narrow) where the coverage is only at 22% and the word list contains more than 10,000 words. A manual analysis showed that some frequent words like 'sich' and 'und' are not in the word list. It becomes clear that a few thousand words are not enough to reach reasonable coverage. 10,000 - 15,000 words seem to be a lower bound for covering around 50% of this short text.
- For most of the languages the PER is lower than the WER. This is a good sign, as it suggests that if a word is in the list, it is at least partially spelled the same as in the reference text. Still, it is surprising that for broad Spanish, the WER is actually lower than the PER. Results like this show that even if the words are covered, their phonetic transcription might be spelled differently. This is related to the fact that the IPA is not really standardized. It always depends on the person who transcribe a text.

The results from this experiment are summarized in table 5. Generally it is good to see that most texts are at least partially covered by the pronunciation dictionary. It will later be interesting to see how the neural models perform on these short texts.

Iso 639-3	Coverage	WER	PER	Type ref	Type list	# Words list
cmn	85.15	93.07	59.26	unk	broad	133,686
deu	75.00	72.22	52.67	broad	broad	34,145
deu	22.22	97.22	84.85	narrow	narrow	10,984
ell	26.32	84.21	87.74	unk	broad	10,547
eng	92.04	83.19	37.27	broad	broad	57,230
eng	7.08	100.00	108.35	narrow	narrow	1,633
eus	5.75	96.55	97.95	broad	broad	1,742
ind	22.22	96.30	88.04	unk	broad	1,555
ind	1.85	100.00	101.69	unk	narrow	2,637
kat	44.29	90.00	73.85	broad	broad	15,123
mya	7.50	97.50	97.70	broad	broad	4,631
spa	64.95	46.39	48.64	broad	broad	60,677
spa	35.05	98.97	65.92	narrow	narrow	52,190
tha	83.85	88.20	40.62	unk	broad	15,050
tur	18.46	100.00	88.89	unk	broad	1,789
tur	6.15	100.00	92.53	unk	narrow	1,812
hin	31.75	100.00	84.00	unk	narrow	9,563
hin	57.94	93.65	69.81	unk	broad	10,812
kor	18.64	100.00	51.97	unk	narrow	14,141
fra	86.11	51.85	40.76	unk	broad	56,911
vie	96.58	79.49	48.66	unk	narrow	15,240
rus	94.79	95.83	56.30	unk	narrow	402,586

Table 6: The table shows the coverage, WER and PER when the pronunciation dictionaries are used to write *The North Wind and the Sun*.

4.3 Language profiles

Once I collected my datasets, I wanted to find out what characters they include. I collected phoneme and grapheme profiles of the data and compared it to the PHOIBLE dataset. A profile lists all used characters or tokens (this depends on your chosen settings) for a text and lists its frequency. All profiles were collected for each language and each dataset separately. For each language and each data type I got these three lists:

- Grapheme list: contains all graphemes in that language. Characters that need a base character like diacritics are shown together with their base character.

- Phoneme list: contains all phonemes in that language. Again, diacritics and similar characters are shown with their base characters.
- Phoneme cluster list: Phonemes can be clustered into bigger sound groups. How to do this, is an ongoing discussion, but I used the segments library to get the clusters (compare Moran and Cysouw [2018])

Having this overview for the characters for each language allowed me to compare the character vocabulary to the characters or character clusters available in the Phoible dataset. This comparison showed that quite a few characters are missing that are included in the WikiPron data and the short stories. My observations are listed below.:

- Characters that are not included in the official IPA chart: There are sometimes characters included that are no part of the IPA. There might be a reason why the authors of the transcriptions decided to use this special character to denote a particular sound, but this is not always known. A possibility is to try and map it to a character that is available in Phoible and that represents a similar sound (or even the same sound actually).
- Tie bars: The creators of Phoible decided to exclude tie bars because they add no real value to the transcriptions[3].
- Stress marks: Stress marks are not represented in Phoible as they do not represent a sound. They are included in the NWS short stories. The same is true for other suprasegmental elements.
- Tones: Even within the IPA there exist different conventions of how to represent tones. Some are better suited for different languages. Apart from different ways of representing tones, it is not always sensible to have tones represented. Mostly, tones are not written as speakers of that language know how to pronounce the tones. So, the questions is whether it is necessary to include the tones at all. When looking at the written representation it does not matter what the tones are as the basic phonemes do not change. This is of course different when the phonetic representation is mapped to a spoken representation.

I conducted this preliminary analysis at an early stage of my thesis. In chapter 5 I will clean the datasets to use them for training and testing. At this point, I will give a more detailed list of what characters are in my datasets but not included in PHOIBLE.

[3] `https://phoible.github.io/conventions/`

5 Experiments: Automatic Grapheme-to-Phoneme Conversion

In this chapter, I present the experiments I conducted to obtain models for phonetic transcription. As a result of my first practical part in chapter 4 I presented two datasets that I will now use to perform these experiments. The NWS corpus is much smaller which is why I will use it as a test dataset only. This means that I will train any model on the WikiPron dataset. As the NWS corpus is quite well known among linguists or at least among phoneticians, it will hopefully give interesting insights when testing the models on these short texts. Table 3 shows all languages that I am using in my experiments. Note that WikiPron continuously adds data to their repository. This means that there might be new languages that are in the 100 language corpus and that I did not use. Additionally, there were languages which had a WikiPron dictionary that was smaller than 1,000 pairs. This is simply not enough to train a decent model which is why I excluded these. If the transcription type of the NWS story is unknown, I only added it to the broad type of the WikiPron dictionary, such that there are no duplicates in the table.

The model that I am using for my G2P experiments is explained in section ??. Setting it up was not very easy as there were issues with the tensorflow version and some other dependencies. Also, they do have a pre-trained model, but this uses a completely different transcription convention than IPA. So we cannot use this pre-trained model. But it is of course well possible to train new models for the languages I am using. Generally, broad and narrow transcriptions are treated as separate languages and thus trained separately as well. The same is true for dialects if there is any information available. American and British English are trained in different models. In order to compare my results to already existing models, I will use the results of the SIGMORPHON tasks in 2020 and 2020. They present results for quite a few languages. Also, they use the same data type. They cleaned some of the data for some languages and made these datasets available. Whenever a filtered version is available I use that one as the basis for my experiments.

5.1 Training settings

As I have shortly mentioned before, some WikiPron word lists have less than one thousand word pairs. I marked them as low resource and only used the high resource languages to train my models. There are different settings which I will use to train the models and analyse their performance. The settings are chosen in a way such that they are built on each other. The first experiments are set up very simply and without much effort. Then I continuously add more complexity. Below I added a short description of each setting as a n overview. I will add more details on the individual steps in separate sections.

There are a few properties that are always the same for all models:

- The CMU model splits the data automatically if no specified otherwise. The automatic split is 85% training data, 5% development data and 10% test data. I did not change this split.
- The hyperparameters are left as default if not mentioned otherwise. More on those can be found on their GitHub[1].

Setting 1: Baseline Short This is the most basic setting. I will train a model for each language to get a baseline result. The model is trained with the least amount of effort. The model is trained for the minimum number of steps which is 10,000 steps in this case. First, I trained it on those languages where I have results from the SIGMORPHON 2021 challenge. The results are compared in table **??**.

Setting 2: Baseline Long This setting is similar to setting 1 except that the model is trained as long as possible for each language. All models have been trained for 200,000 steps and the default settings. This setting is supposed to show if training for a considerably longer time changes the results a lot or not.

Setting 3: Baseline clean short I will train another model that is the same like the Baseline Large, but I will use the cleaned WikiPron data. What I will clean is described below (section 5.2).

Setting 4: Baseline clean long The same as setting 3 except that the models were trained for as long as possible like in setting 2.

[1] `https://github.com/cmusphinx/g2p-seq2seq`

Setting 5: Feature input version 1 short The final experiments will be with phonetic features as input. I use the cleaned WikiPron data from setting 3 and add the features to it. In section 5.3, I explain how I encode the features. Before training this set of models, I did some experiments with other features than the final ones I used here. All of this is explained in section 5.3 as well.

Setting 6: Feature input version 1 long The same as setting 5, but again the models for trained for 200,000 steps like in setting 2.

5.2 Preprocessing

In section 2.4 I talked about the incompleteness and difficulties of transcribing using the IPA. How exactly a sound is mapped to an IPA symbol also depends on whoever transcribes a particular text. The WikiPron data has been put together by many different people. There are conventions on how to add transcriptions by Wiktionary, but there still might be inconsistencies. Other than that, it is always possible that something is correct, but neural models just cannot handle it well. That said I will carefully examine and clean the datasets. There is preprocessing that is done for both datasets, further down I will describe some preprocessing that had to be done for each individual dataset.

The first analysis is a character based comparison. This means that I used the python segments library[2] to split up all the phonemes into unicode characters. Then I compared those characters to the PHOIBLE character set. This way I can identify characters that are a bit suspicious in the context of IPA as the PHOIBLE dataset is very extensive. While it is possible that a correctly used phoneme is not in PHOIBLE, it still gives a good overview of potential ambiguities in the transcription or even points out mistakes. This character based cleaning is done for both datasets. The more detailed list of what needs to be cleaned is found in table 6.

There are two more general things that I removed from the datasets. The first is **tones**. The reason for that is that it is not possible to infer tones from graphemes. Although they are used to distinguish meaning, they are not written down, but usually just known. This means that there are grapheme sequences that look exactly the same and their transcription is the same as well *except* for the tones. There are different ways to represent tones and I excluded all of them. The second additional preprocessing step I took is to remove all **punctuation symbols**. This accounts

[2]https://pypi.org/project/segments/

mostly for the NWS short stories. For a character-based modelling, punctuation does not add any valuable information as this only becomes relevant, and can only be inferred, on a word, respectively, sentence basis.

Phon.	Unicode name	Repl.	Explanation
ˈ	MODIFIER LETTER VERTICAL LINE	NULL	These are all IPA suprasegmentals except the long and half long marker and the extra short (ː ˑ ˘). The reason why these were excluded is that they don't carry any meaning on the character level. The vertical lines, for example, mark intonation groups which only matter in a larger sentence or text context. There are a few rare occurrences of COMBINING VERTICAL LINE ABOVE which is probably meant to be MODIFIER LETTER VERTICAL LINE as they look similar. It is excluded as well.
ˌ	MODIFIER LETTER LOW VERTICAL LINE	NULL	
|	VERTICAL LINE	NULL	
‖	DOUBLE VERTICAL LINE	NULL	
.	FULL STOP	NULL	
‿	UNDERTIE	NULL	
◌͡◌	COMBINING DOUBLE INVERTED BREVE	NULL	Both tie bars below and above are excluded in PHOIBLE[3] which is why I am excluding it as well. Put plainly, those do not add any additional information that cannot be derived otherwise.
◌͜◌	COMBINING DOUBLE BREVE BELOW	NULL	
ɜ˞	LATIN SMALL LETTER REVERSED OPEN E WITH HOOK	ə˞	Both base letters are very similar vowels. It is just more common to use the latter than the former.
g	LATIN SMALL LETTER G	ɡ	The IPA 'g' has a different code point and is a different character than the typical keyboard small Latin 'g'. This is just an IPA decision. For some fonts the two characters do not look different, for some they do.
∼	SWUNG DASH	NULL	All of characters make out less than 1% of their respective dataset, most of the time it is less than 0.1%. A close examination of the dataset and the Wiktionary transcription conventions for the respective language did not show any reason why to keep the phoneme. Note that the 'v' for the tilde is only there for correct representation.
,	COMMA	NULL	
ṽ	TILDE	NULL	
ᵊ	MODIFIER LETTER SMALL SCHWA	NULL	
ˣ	MODIFIER LETTER SMALL X	ʔ	The ˣ only occurred in the broad Finnish transcription and is used to denote possible gemination. In the narrow transcriptions there is a glottal stop instead. The occurrence of glottal stops and gemination follows the same rules. Therefore, for consistency, the gemination ˣ is mapped to a LATIN LETTER GLOTTAL STOP.
( )	(SUPERSCRIPT) [LEFT | RIGHT] PARENTHESIS	NULL	Parentheses are used to denote optionality for phonemes or tones. WikiPron actually discards those but keeps the content[4]. I will do the same for all parenthesis found.

Table 7: The table shows what phonemes where changed or excluded and what the reason is for this preprocessing. All characters that were excluded are replaced by a NULL value.

NWS corpus

Apart from cleaning the stories as described in table5.2, I had to transform them into dictionary format in order to be able to use them as testing data. I order to do that it is necessary to tokenize both orthographic and phonetic texts and then align them. As the number of tokens (when split naively at blanks) is not always the same for orthographic and phonetic text, it was necessary to align some of them manually. While this is not a problem when talking about languages that I know, it is a bit tricky for languages completely unknown to me. Luckily there are tools online that provide a rough pronunciation of a word in a given language or even a phonetic transcription (although rarely in IPA)[5].

Chinese and Thai required special tokenization. To tokenize Chinese and Thai I used *polyglot*[6], a Python library.

WikiPron

As the CMU model does not expect the graphemes to contain any whitespace, I replaced any whitespace in the input with underscores. This was necessary only for Vietnamese. In a first run, I did not do this. When running the evaluation on the Vietnamese model, it did not work but showed an error message. As the model creates a vocabulary file, I had a look at it which revealed the following:

- The model does not actually split the input file at tab characters, but splits it at spaces and then uses the first item of the resulting list as input grapheme and the rest as a list containing the output phonemes. Consequently, if the input grapheme contains a white space, everything following it will be in the phoneme part. Part of the input will be treated as a phoneme character in the training which results in a huge vocabulary and wrong predictions.

The alternative to removing the white space would have been to split the grapheme at white space and align the phonemes. However, this is firstly much more work as it is not clear where to split the phonemes and it might be that the word on its own is pronounced differently.

In Japanese there is the superscript letter $^{\beta}$. This actually means compression, which

[5]For most languages I could use Google translator (`https://translate.google.com/?hl=de&sl=ko&tl=de&op=translate`), for languages like Hebrew, I could ask someone who knows the language to help me out.

[6]`https://polyglot.readthedocs.io/en/latest/`

is a special type of rounding[7]. This is one of these things where the difficulties of transcribing texts becomes clear. This special case is not reflected in PHOIBLE nor in the official IPA but this does not mean it is wrong. As it is quite common in the Japanese data, I decided to keep it.

A last thing I did to clean the datasets was to exclude duplicate graphemes with different pronunciations. Although ambiguities and multiple possible pronunciations for one word are linguistically speaking very common, it is not possible for a neural model to distinguish such cases without any context. As G2P modelling happens on character basis and not on word basis there is no context available in our case that could account for at least some ambiguities. Also, in the WikiPron data that was preprocessed for the SIGMORPHON task, duplicates were excluded as well. It makes sense to clean all datasets in the same way.

5.3 Feature encoding

In order to incorporate the phonetic features into the dataset, we decided to add what we refer to as 'flags' to the phonemes. Generally, the idea is to encode the phonetic features that PHOIBLE provides and add them to the WikiPron dataset. There are still a few phonemes that are not in the PHOIBLE set that are in my data. These will jsut not b encoded in the following experiments. An exception is the length marker. There are a few long vowels that are not like that in the PHOIBLE data. As there is the 'long' feature, it is easily possible to use use the base character feature vector and mark the 'long' as '+'. This is the only thing I changed for the following experiments. All other features vectors are just used as is. I encoded the features in a jupyter notebook which is in my GitHub repository[8].

I experimented with several things before I trained a first set of models. I will list the steps of my experiments in the following. For all feature experiments I used the cleaned WikiPron data.

Version 1

The first thing I tried was to add two flags to every phoneme. In order to do this, I encoded the features as two strings. This means that I split the features into two

[7] https://en.wikipedia.org/wiki/Roundedness

[8] https://github.com/theDebbister/masterThesis/blob/main/data/MA_Encode_features.ipynb

sets and then used a different set of random capital letters to encode all of these uniquely. After each phoneme I will add two flags that describe that phoneme. The combination of the two flags will be unique for each phoneme. The individual flags can overlap between the phonemes, if one half of the features is the same for that phoneme. The example below shows a possible encoding of a phonetic text:

(5.1) p AB CD k^h AA BC

(5.2) k^h AA BC ʊ BF CC

A problem with this approach is that the sequences get really long if all phonemes are represented as three strings. The CMU model cuts the sequence off at 30 characters. This means that only phoneme sequences that no longer than 10 characters will be represented in full. This is way to short to account for many words. It is possible to increase that limit. However, 30 characters is already quite long and there is no point in setting this limit too high as those models still experience difficulties in processing sequences that are too long. Not only the sequence length is influenced but also the vocabulary size. Each flag will be added to the vocabulary.

It is therefore not surprising that the model did not perform well. In fact, the WER was about double the WER of the long model trained on the uncleaned data. Given this results, I decided not to train any more models and look for another way of how to encode the features.

Version 2

For the second attempt at encoding phonetic features I tried a different approach. Instead of adding two flags, I added only one flag after each phoneme. Also, this time the idea was not to add a unique flag to each phoneme but to encode only some features for all phonemes. This means that for some phonemes the same flag was added. As the phonemes were not replaced by the flags, but the phonemes were kept in the text, there is no information lost. The intuition behind it is to encode high-level patterns in phoneme sequences. For example, as vowels typically have some overlapping features, very similar vowels will be encoded using the same flag. In order to choose appropriate features for each language, I calculated the pearson correlation between the features. I only used those features whose correlation with all other features is not above a certain threshold. The reason for this is that all languages only use a relatively small subset of phonemes. This means that many features are not important for a language as those features are not used to distinguish meaning. In order to recognize high-level patterns, only the most distinctive features

are important. Other than that I encoded the features in the same way as for version one. First results for this version reached similar scores as the baselines ones. I trained a full set of models for this version.

While encoding these features, I noticed that some phonemes are not listed as individual phonemes in PHOIBLE but are listed as allophones of one phoneme that is in there. This means that some broad transcriptions contain allophones which, ideally, should not be the case. Also, it means that I did not have features for these phonemes which means that those were not encoded.

5.4 Results

For the evaluation of my models, I first used the test sets that were produced automatically by the model. In addition, I tested each model on the respective NWS story if available in that language.

ISO396-3	BS WER	BS WER NWS	SIG WER	Transcription type
eng (us)	54.40	87.60	37.43	broad
fra	7.20	47.20	5.11	broad
ell	9.80	83.20	18.67	broad
kat	0.30	65.20	0.00	broad
hin	5.60	87.10	5.11	broad
jpn	6.60	-	4.89	narrow
kor	28.70	100.00	16.20	narrow
vie	7.50	100.00	0.89	narrow

Table 8: Baseline CMUSphinx results compared with SIGMORPHON 2020 and 2021 results. For each language the best score is reported no matter what year or what model. The table shows the results for setting 1. CMUSphinx provides a WER implementation which has been used to evaluate the models.

Setting 1: Baseline Short

Setting 2: Baseline Long

Setting 3: Baseline clean

Iso 639-3	Type WikiPron	WER BS	PER BS	WER BS-clean	PER BS-clean	WER F2	PER F2
cmn	broad	17.6	3.9			18.1	4.4
deu	broad	37.1	4.8			38.6	5.9
deu	narrow	52.2	7.1			55.9	9.4
ell	broad	7.1	0.6			9.1	0.8
eng us	broad	50.7	9.5			51.3	10.4
eng us	narrow	84.6	31.4			84.9	32.3
eng uk	broad	45.5	8.6			47.6	9.7
eng uk	narrow	90.3	30.2			93.7	34.2
eus	broad	21.2	2.7			21.7	3.2
fin	broad	2.8	0.2			8.9	3.1
fin	narrow	3.2	0.3			9.0	3.5
fra	broad	5.3	0.7			5.9	0.8
hin	narrow	7.7	1.2			8.4	1.3
hin	broad	4.4	0.7			6.3	1.0
ind	broad	37.9	5.3			39.3	6.1
ind	narrow	43.1	5.4			43.1	5.6
jpn	narrow	6.5	0.6			6.6	0.7
kat	broad	0.0	0.0			1.0	0.8
kor	narrow	23.4	4.1			25.8	4.6
mya	broad	35.1	6.5			88.0	17.4
rus	narrow	1.9	0.2			5.0	1.5
spa ca	broad	1.1	0.1			2.2	0.7
spa ca	narrow	2.3	0.3			2.8	0.6
spa la	broad	1.4	0.1			1.9	0.6
spa la	narrow	2.6	0.3			2.7	0.4
tgl	broad	28.4	4.6			33.9	5.0
tgl	narrow	45.5	6.4			48.6	6.9
tha	broad	12.5	2.6			12.1	3.1
tur	broad	50.6	7.8			49.1	7.2
tur	narrow	55.1	7.6			54.8	8.0
vie	narrow	1.5	0.8			2.6	1.5
zul	broad	65.9	10.7			91.4	12.0

Table 9: This table shows the results for setting 2 (BS), setting 4 (BS-clean) and setting 6 (F2)

ISO396-3	BS WER	BS WER NWS	SIG21 WER	Transcription type
eng (us)	50.70		37.43	broad
fra	5.30		5.11	broad
ell	7.10		18.67	broad
kat	0.00		0.00	broad
hin	4.40		5.11	broad
jpn	6.50		4.89	narrow
kor	23.40		16.20	narrow
vie	7.10		0.89	narrow

Table 10: Baseline CMUSphinx results compared with SIGMORPHON 2020 and 2021 results. For each language the best score is reported no matter what year or what model. The table shows the results for setting 2. CMUSphinx provides a WER implementation which has been used to evaluate the models.

Setting 4: Feature input

6 Conclusion

6.1 What now?

Non surprisingly, apart from the many exciting things I *could* do, there are many others that would have gone beyond the scope of this thesis. I would like to list a few entry points on where further research could start.

- **Data preprocessing**: ? cleaned broad transcriptions for Bulgarian and replaced allophones by their standard phoneme. This could further improve model quality by having consistent broad transcriptions.
-

References

100-language-sample. WALS Online - Languages. In M. S. Dryer and M. Haspelmath, editors, *The World Atlas of Language Structures Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig, 2013. URL https://wals.info/languoid/samples/100.

S. Ai. A brief overview of Imitation Learning - SmartLab AI - Medium. *Medium*, 19.9.2019. URL https://smartlabai.medium.com/a-brief-overview-of-imitation-learning-8a8a75c44a9c.

J. Alammar. The Illustrated Transformer, 03.01.2022. URL https://jalammar.github.io/illustrated-transformer/.

L. F. Ashby, T. M. Bartley, S. Clematide, L. Del Signore, C. Gibson, K. Gorman, Y. Lee-Sikka, P. Makarov, A. Malanoski, S. Miller, O. Ortiz, R. Raff, A. Sengupta, B. Seo, Y. Spektor, and W. Yan. Results of the Second SIGMORPHON Shared Task on Multilingual Grapheme-to-Phoneme Conversion. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Stroudsburg, PA, USA, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.sigmorphon-1.13.

L. Baird, N. Evans, and S. J. Greenhill. Blowing in the wind: Using 'north wind and the sun' texts to sample phoneme inventories. *Journal of the International Phonetic Association*, page 1–42, 2021. doi: 10.1017/S002510032000033X.

M. Bisani and H. Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451, 2008. ISSN 0167-6393. doi: https://doi.org/10.1016/j.specom.2008.01.002. URL https://www.sciencedirect.com/science/article/pii/S0167639308000046.

E. Chodroff. Corpus Phonetics Tutorial, 2019. URL https://eleanorchodroff.com/tutorial/index.html#.

S. Clematide and P. Makarov. CLUZH at SIGMORPHON 2021 Shared Task on Multilingual Grapheme-to-Phoneme Conversion: Variations on a Baseline. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.sigmorphon-1.17.

B. Comrie, M. S. Dryer, D. Gil, and M. Haspelmath. Introduction. In M. S. Dryer and M. Haspelmath, editors, *The World Atlas of Language Structures Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig, 2013. URL `https://wals.info/chapter/s1`.

CrashCourse, 2021a. URL `https://www.youtube.com/watch?v=vyea8Ph9BOM`.

CrashCourse, 2021b. URL `https://www.youtube.com/watch?v=-sUUWyo4RZQ&list=PL8dPuuaLjXtP5mp25nStsuDzk2blncJDW&index=18`.

V. Gautam, W. Li, Z. Mahmood, F. Mailhot, S. Nadig, R. Wang, and N. Zhang. Avengers, ensemble! benefits of ensembling in grapheme-to-phoneme prediction. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 141–147, 01 2021. doi: 10.18653/v1/2021.sigmorphon-1.16.

GitHub. cmusphinx/g2p-seq2seq: G2P with Tensorflow, 03.02.2022. URL `https://github.com/cmusphinx/g2p-seq2seq`.

GitHub. SimonGreenhill/jipa, 04.02.2022. URL `https://github.com/SimonGreenhill/jipa`.

K. Gorman, L. F. Ashby, A. Goyzueta, A. McCarthy, S. Wu, and D. You. The SIGMORPHON 2020 shared task on multilingual grapheme-to-phoneme conversion. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 40–50, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.sigmorphon-1.2. URL `https://aclanthology.org/2020.sigmorphon-1.2`.

X. Gutierrez-Vasques, C. Bentz, O. Sozinova, and T. Samardzic. From characters to words: the turning point of BPE merges. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3454–3468, Online, Apr. 2021. Association for Computational Linguistics. URL `https://aclanthology.org/2021.eacl-main.302`.

M. Hammond. Data augmentation for low-resource grapheme-to-phoneme mapping. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 126–130, Online, Aug. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.sigmorphon-1.14. URL https://aclanthology.org/2021.sigmorphon-1.14.

H. H. Hock and B. D. Joseph. *Language History, Language Change, and Language Relationship*. De Gruyter, 2019. ISBN 9783110613285. doi: 10.1515/9783110613285.

U. Inc. UAX #15: Unicode Normalization Forms, 27.08.2021. URL https://unicode.org/reports/tr15/.

N. Jaitly, D. Sussillo, Q. V. Le, O. Vinyals, I. Sutskever, and S. Bengio. A neural transducer, 2016.

S. Kostadinov. Understanding Encoder-Decoder Sequence to Sequence Model. *Towards Data Science*, 02.05.2019. URL https://towardsdatascience.com/understanding-encoder-decoder-sequence-to-sequence-model-679e04af4346.

S. Kostadinov. How Recurrent Neural Networks work - Towards Data Science. *Towards Data Science*, 12.02.2017. URL https://towardsdatascience.com/learn-how-recurrent-neural-networks-work-84e975feaaf7.

M. Kracht. *Introduction to linguistics*. Los Angeles, 2007. URL https://linguistics.ucla.edu/people/kracht/courses/ling20-fall07/ling-intro.pdf.

J. L. Lee, L. F. Ashby, M. E. Garza, Y. Lee-Sikka, S. Miller, A. Wong, A. D. McCarthy, and K. Gorman. Massively Multilingual Pronunciation Modeling with WikiPron. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4223–4228, Marseille, France, 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.521.

K. Leung. Evaluate OCR Output Quality with Character Error Rate (CER) and Word Error Rate (WER). *Towards Data Science*, 24.6.2021. URL https://towardsdatascience.com/evaluating-ocr-output-quality-with-character-error-rate-cer-and-word-error-rate-5aec.

M. Y. Liberman. Corpus Phonetics. *Annual Review of Linguistics*, 5(1):91–107, 2019. ISSN 2333-9683. doi: 10.1146/annurev-linguistics-011516-033830.

R. Y.-H. Lo and G. Nicolai. Linguistic knowledge in multilingual grapheme-to-phoneme conversion. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 131–140, Online, Aug. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.sigmorphon-1.15. URL https://aclanthology.org/2021.sigmorphon-1.15.

P. Makarov and S. Clematide. CLUZH at SIGMORPHON 2020 shared task on multilingual grapheme-to-phoneme conversion. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 171–176, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.sigmorphon-1.19. URL https://aclanthology.org/2020.sigmorphon-1.19.

T. McEnery and A. Hardie. *Corpus Linguistics: Method, theory and practice*. Cambridge textbooks in linguistics. Cambridge University Press, Cambridge, 2011. ISBN 9780511981395. doi: 10.1017/CBO9780511981395.

S. Moran and M. Cysouw. *The Unicode Cookbook for Linguists: Managing writing systems using orthography profiles*. 06 2018. ISBN 978-3-96110-090-3. doi: 10.5281/zenodo.1296780.

S. Moran and D. McCloy, editors. *PHOIBLE 2.0*. Max Planck Institute for the Science of Human History, Jena, 2019. URL https://phoible.org/.

D. R. Mortensen, S. Dalmia, and P. Littell. Epitran: Precision G2P for many languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://aclanthology.org/L18-1429.

C. Olah. Understanding LSTM Networks – colah's blog, 29.01.2022. URL http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

B. Peters and A. F. T. Martins. One-size-fits-all multilingual models. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 63–69, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.sigmorphon-1.4. URL https://aclanthology.org/2020.sigmorphon-1.4.

C. U. Press. The principles of the international phonetic association (1949). *Journal of the International Phonetic Association*, 40(3):299–358, 2010. doi: 10.1017/S0025100311000089.

K. Rao, F. Peng, H. Sak, and F. Beaufays. Grapheme-to-phoneme conversion using long short-term memory recurrent neural networks. *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4225–4229, 2015.

Sigmorphon. SIGMORPHON - Special Interest Group on Computational Morphology and Phonology, 2021. URL `https://sigmorphon.github.io/`.

SPUR project. Non-randomness in Morphological Diversity, 2021. URL `https://www.spur.uzh.ch/en/departments/research/textgroup/MorphDiv.html`.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014. URL `http://arxiv.org/abs/1409.3215`.

S.-A. Toma and D. Munteanu. Rule-based automatic phonetic transcription for the romanian language. *Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, Computation World*, 0:682–686, 11 2009. doi: 10.1109/ComputationWorld.2009.59.

X. Yu, N. T. Vu, and J. Kuhn. Ensemble self-training for low-resource languages: Grapheme-to-phoneme conversion and morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 70–78, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.sigmorphon-1.5. URL `https://aclanthology.org/2020.sigmorphon-1.5`.

A Tables

Table 11: The table shows a list of the 100 languages in the corpu
mation on the language families.

Iso639-3	Name WALS	
39-3abkWALSAbkhazWALSNorthwest Caucasian6639-3	–WALS	
6639-3	–WALS	–
6639-3	–WALS	
6639-3	–WALS	–
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	–WAL
6639-3	–WALS	
6639-3	–WALS	–
6639-3	–WALS	–WALS39-3
6639-3	–WALS	
6639-3	–WALS	–WALS
6639-3	–WALS	–WALS
6639-3	–WALS	
6639-3	–WALS	–WA
6639-3	–WALS	–WALS
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	–WALS
6639-3	–WALS	
6639-3	–WALS	

Iso639-3	Name WALS	
6639-3	–WALS	–V
6639-3	–WALS	–WALS
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	–WALS39
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	–
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	–WALS39-
6639-3	–WALS	–W
6639-3	–WALS	–WALS39-3dniWALSD
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	–WALS
6639-3	–WALS	
6639-3	–WALS	–
6639-3	–WALS	
6639-3	–WALS	–WAL
6639-3	–WALS	
6639-3	–WALS	–WALS39
6639-3	–WALS	–WAL
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	

Iso639-3	Name WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	–WALS39-3m
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	–V
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	–V
6639-3	–WALS	–V
6639-3	–WALS	–WA
6639-3	–WALS	
6639-3	–WALS	–W
6639-3	–WALS	
6639-3	–WALS	–
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	
6639-3	–WALS	

Iso639-3	**Name WALS**
6639-3	–WALS

Iso639-3	Name WALS	
39-3abkWALSAbkhazWALSNorthwest Caucasian6639-3	–WALS	–WALS39-3amp
6639-3	–WALS	–WALS39-3aeyWALSAm
6639-3	–WALS	–WALS39-3apuW
6639-3	–WALS	–WALS39-3bmiWALSB
6639-3	–WALS	–WALS39-3bsnWA
6639-3	–WALS	–WALS39-3gryWA
6639-3	–WALS	–WALS39-3eus
6639-3	–WALS	–WALS39-3apeWALSArape
6639-3	–WALS	–WALS39-3bskWALSI
6639-3	–WALS	–WALS39-3ramWALSC
6639-3	–WALS	–WALS39-3tzmWALSBerber (Mi
6639-3	–WALS	–WALS39-3chaWALSC
6639-3	–WALS	–WALS39-3cktWALSChukchiV
6639-3	–WALS	–WALS39-3zocWALSZoque (C
6639-3	–WALS	–WALS39-3
6639-3	–WALS	–WALS39-3haeWALSOron
6639-3	–WALS	–WALS39-3arzWALSArabic
6639-3	–WALS	–WALS39-3fijWA
6639-3	–WALS	–WALS39-3
6639-3	–WALS	–WALS39-3gniWAL
6639-3	–WALS	–WALS39-3k
6639-3	–WALS	–WALS39-3hauW
6639-3	–WALS	–WALS39-3hebWALSHebrew
6639-3	–WALS	–WALS39-3hinWAL
6639-3	–WALS	–WALS39-3hixWA
6639-3	–WALS	–WALS39-3hnjWALSHm
6639-3	–WALS	–WALS39-3qviWALSQuechua
6639-3	–WALS	–WALS39-3im
6639-3	–WALS	–WALS39-3indWALSInd
6639-3	–WALS	–WALS39-3kalWALSGreenland
6639-3	–WALS	–WALS39-3
6639-3	–WALS	–WALS39-3gydW
6639-3	–WALS	–WALS39-3kioWAL
6639-3	–WALS	–WALS39-3ckuWA
6639-3	–WALS	–WALS39-3ko
6639-3	–WALS	–WALS39-3sesWALSKoy
6639-3	–WALS	–WALS39-3k
6639-3	–WALS	–WALS39-3ku
6639-3	–WALS	–WALS39-3lk
6639-3	–WALS	–WALS39-3lajWALSL
6639-3	–WALS	–WALS39-3lvkWALSLavukaleveW
6639-3	–WALS	–WALS39-3lezWALSLezgia
6639-3	–WALS	–WALS39-3dniWALSDani (Lower Grand Val