University of Lisbon
Faculty of Arts and Humanities

Preprocessing Models for Speech Technologies: The impact of the Normalizer and the Grapheme-to-Phoneme on Hybrid Systems

Masters in Linguistics

Bruna dos Santos Carriço

2022

Internship Report elaborated to obtain a Master's degree, supervised by Professor Helena Gorete Silva Moniz and co-supervised by Doctor Christopher Dane Shulby

Acknowledgments

First, I would like to thank my supervisor, Professor Helena Moniz, whose knowledge and emotional guidance were precious in the process of this thesis. Your insightful feedback encouraged me to improve my linguistic skills and raised the quality of my work.

I would like to thank Defined.ai, for giving me the chance to work and grow with them. A very special thank you to my colleagues for their wonderful collaboration. I want to specially mention my co-supervisor, Doctor Christopher Shulby. Chris, I appreciate your patient support and all the opportunities you gave me to further my career.

My gratitude also goes out to my lifelong friends who have protected and taken good care of me.

Obrigada, Mãe, pelo teu amor incondicional.

Obrigada, Rodrigo, pelas tuas gargalhadas e abraços.

Obrigada, Tia Carla, Tio Rafael, Prima Leonor. O vosso apoio nunca será esquecido.

Resumo

Um dos usos mais promissores e de crescimento mais rápido da tecnologia de linguagem natural corresponde às Tecnologias de Processamento da Fala. Esses sistemas usam tecnologia de reconhecimento automático de fala e conversão de texto em fala para fornecer uma interface de voz para aplicações de conversão. Com efeito, esta tecnologia está presente em diversas situações do nosso quotidiano, tais como assistentes virtuais em *smartphones* (como a SIRI ou Alexa), ou sistemas de interação por voz em automóveis.

As tecnologias de fala evoluíram progressivamente até ao ponto em que os sistemas podem prestar pouca atenção à sua estrutura linguística. Com efeito, o Conhecimento Linguístico pode ser extremamente importante numa arquitetura de fala, particularmente numa fase de pré-processamento de dados: combinar conhecimento linguístico em modelo de tecnologia de fala permite produzir sistemas mais confiáveis e robustos.

Neste sentido, o pré-processamento de dados é uma etapa fundamental na construção de um modelo de Inteligência Artificial (IA). Se os dados forem razoavelmente pré-processados, os resultados serão consistentes e de alta qualidade (García et al., 2016). Por exemplo, os sistemas mais modernos de reconhecimento de fala permitem modelizar entidades linguísticas em vários níveis, frases, palavras, fones e outras unidades, usando várias abordagens estatísticas (Jurafsky & Martin, 2022). Apesar de treinados sobre dados, estes sistemas são tão mais precisos quanto mais eficazes e eficientes a capturarem o conhecimento linguístico.

Perante este cenário, este trabalho descreve os métodos de pré-processamento linguístico em sistemas híbridos (de inteligência artificial combinada com conhecimento linguístico) fornecidos por uma empresa internacional de Inteligência Artificial (IA), a Defined.ai. A start-up concentra-se em fornecer dados, modelos e ferramentas de alta qualidade para IA., a partir da sua plataforma de crowdsourcing Neevo. O utilizador da plataforma tem acesso a pequenas tarefas de anotação de dados, tais como: transcrição, gravação e anotação de áudios, validação de pronúncia, tradução de frases, classificação de sentimentos num texto,

ou até extração de informação a partir de imagens e vídeos. Até ao momento, a empresa conta com mais de 500,000 utilizadores de 70 países e 50 línguas diferentes.

Através duma recolha descentralizada dos dados, a Defined.ai responde à necessidade crescente de dados de treino que sejam justos, i.e., que não reflitam e/ou amplifiquem os padrões de discriminação vigentes na nossa sociedade (e.g., de género, raça, orientação sexual). Como resultado, a Defined.ai pode ser vista como uma comunidade de especialistas em IA, que produz sistemas justos, éticos e de futuro.

Assim, o principal objetivo deste trabalho é aprimorar e avançar a qualidade dos modelos de pré-processamento, aplicando-lhes conhecimento linguístico. Assim, focamo-nos em dois modelos linguísticos introdutórios numa arquitetura de fala: Normalizador e Grafema-Fonema.

Para abordar o assunto principal deste estudo, vamos delinear duas iniciativas realizadas em colaboração com a equipa de *Machine learning* da Defined.ai. O primeiro projeto centra-se na expansão e melhoria de um modelo Normalizador pt-PT. O segundo projeto abrange a criação de modelos Grafema-Fonema (do inglês *Grapheme-to-phoneme*, G2P) para duas línguas diferentes – Sueco e Russo.

Os resultados mostram que ter uma abordagem baseada em regras para o Normalizador e G2P aumenta a sua precisão e desempenho, representado uma vantagem significativa na melhoria das ferramentas da Defined.ai e nas arquiteturas de fala. Além disso, com os resultados obtidos no primeiro projeto, melhoramos o normalizador na sua facilidade de uso, aumentando cada regra com o respetivo conhecimento linguístico. Desta forma, a nossa pesquisa demonstra o valor e a importância do conhecimento linguístico em modelos de pré-processamento.

O primeiro projeto teve como objetivo fornecer cobertura para diversas regras linguísticas: Números Reais, Símbolos, Abreviaturas, Ordinais, Medidas, Moeda, Datas e Hora. A tarefa consistia em expandir as regras com suas respetivas expressões normalizadas a partir de regras

a seguir que teriam uma leitura não marcada inequívoca própria. O objetivo principal é melhorar o normalizador tornando-o mais simples, consistente entre diferentes linguagens e de forma a cobrir entradas não ambíguas.

Para preparar um modelo G2P para dois idiomas diferentes - Sueco e Russo - quatro tarefas foram realizadas: 1. Preparar uma análise linguística de cada língua, 2. Desenvolver um inventário fonético-fonológico inicial, 3. Mapear e converter automaticamente o léxico fonético para DC-Arpabet (o alfabeto fonético que a Defined.ai construiu), 4. Rever e corrigir o léxico fonético, e 4. Avaliar o modelo Grafema-Fonema. A revisão dos léxicos fonéticos foi realizada, em consulta com a nossa equipa da Defined.ai, por linguistas nativos que verificaram se os inventários fonéticos-fonológicos seriam adequados para transcrever.

Segundo os resultados de cada modelo, nós avaliamos de acordo com 5 métricas padrão na literatura: *Word Error Rate* (WER), *Precision*, *Recall*, *F1-score* e *Accuracy*. Adaptamos a métrica WER para *Word Error Rate over normalizable tokens* (WERnorm) por forma a responder às necessidades dos nossos modelos. A métrica WER (ou taxa de erro por palavra) foi adaptada de forma a contabilizar *tokens* normalizáveis, em vez de todos os *tokens*. Deste modo, a avaliação do normalizador, avalia-se usando um conjunto de aproximadamente 1000 frases de referência, normalizadas manualmente e marcadas com a regra de normalização que deveria ser aplicada (por exemplo, números reais, símbolos, entre outros).

De acordo com os resultados, na versão 2 do normalizador, obtivemos discrepâncias estatisticamente significativas entre as regras. A regra dos ordinais apresenta a maior percentagem (94%) e as abreviaturas (43%) o menor percentual. Concluímos também um aumento significativo no desempenho de algumas das regras. Por exemplo, as abreviaturas mostram um desempenho de 23 pontos percentuais (pp.) superior. Quando comparamos as duas versões, concluímos que a versão 2 do normalizador apresenta, em média, uma taxa de erro 4 pp. menor sobre os *tokens* normalizáveis em comparação com a versão 1. Assim, o uso da regra dos ordinais (94% F1-score) e da regra dos números reais (89% F1-score) é a maior fonte de

melhoria no normalizador. Além disso, em relação à precisão, a versão 2 apresenta uma melhoria de, em média, 28 pp em relação à versão 1. No geral, os resultados revelam inequivocamente uma melhoria da performance do normalizador em todas as regras aplicadas.

De acordo com os resultados do segundo projeto, o léxico fonético sueco alcançou um WER de 10%, enquanto o léxico fonético russo um WER ligeiramente inferior (11%). Os inventários fonético-fonológicos suecos apresentam uma precisão maior (97%) do que os inventários fonético-fonológicos russos (96%). No geral, o modelo sueco G2P apresenta um melhor desempenho (98%), embora a sua diferença ser menor quando comparado ao modelo russo (96%).

Em conclusão, os resultados obtidos tiveram um impacto significativo na pipeline de fala da empresa e nas arquiteturas de fala escrita (15% é a arquitetura de fala). Além disso, a versão 2 do normalizador começou a ser usada noutros projetos do Defined.ai, principalmente em coleções de *prompts* de fala. Observamos que nossa expansão e melhoria na ferramenta abrangeu expressões que compõem uma proporção considerável de expressões normalizáveis, não limitando a utilidade da ferramenta, mas aumentando a diversidade que ela pode oferecer ao entregar *prompts*, por exemplo.

Com base no trabalho desenvolvido, podemos observar que, ao ter uma abordagem baseada em regras para o Normalizador e o G2P, conseguimos aumentar a sua precisão e desempenho, representando não só uma vantagem significativa na melhoria das ferramentas da Defined.ai, como também nas arquiteturas de fala. Além disso, a nossa abordagem também foi aplicada a outras línguas obtendo resultados muito positivos e mostrando a importância da metodologia aplicada nesta tese. Desta forma, o nosso trabalho mostra a relevância e o valor acrescentado de aplicar conhecimento linguístico a modelos de pré-processamento.

Palavras-chave: Tecnologias de Fala; Normalizador; Grafema-Fonema; Conhecimento Linguístico; modelos

Abstract

One of the most fast-growing and highly promising uses of natural language technology is in Speech Technologies. Such systems use automatic speech recognition (ASR) and text-to-speech (TTS) technology to provide a voice interface for conversational applications.

Speech technologies have progressively evolved to the point where they pay little attention to their linguistic structure. Indeed, linguistic knowledge can be extremely important in a speech pipeline, particularly in the Data Preprocessing phase: combining linguistic knowledge in a speech technology model allows producing more reliable and robust systems.

Given this background, this work describes the linguistic preprocessing methods in hybrid systems provided by an Artificial Intelligence (AI) international company, Defined.ai. The startup focuses on providing high-quality data, models, and AI tools. The main goal of this work is to enhance and advance the quality of preprocessing models by applying linguistic knowledge. Thus, we focus on two introductory linguistic models in a speech pipeline: Normalizer and Grapheme-to-Phoneme (G2P). To do so, two initiatives were conducted in collaboration with the Defined.ai Machine Learning team. The first project focuses on expanding and improving a pt-PT Normalizer model. The second project covers creating G2P models for two different languages – Swedish and Russian.

Results show that having a rule-based approach to the Normalizer and G2P increases its accuracy and performance, representing a significant advantage in improving Defined.ai tools and speech pipelines. Also, with the results obtained on the first project, we improved the normalizer in ease of use by increasing each rule with linguistic knowledge. Accordingly, our research demonstrates the added value of linguistic knowledge in preprocessing models.

Keywords: Speech Technologies; Normalizer; Grapheme-to-Phoneme; Linguistic knowledge; models

Abbreviations

AED	Attention-Based Encoder-Decoder
AI	Artificial Intelligence
AM	Acoustic Model
ASCII	American Standard Code for Information Interchange
ASR	Automatic Speech Recognizer
AST	Automatic Segmentation & Transcription
CEO	Chief Executive Officer
CNN	Convolutional Neural Networks
CTC	Connectionist Temporal Classification
DBN	Deep Belief Network
DNN	Deep Neural Networks
DST	Dialog State Tracker
E2E	End-To-End
G2P	Grapheme-To-Phoneme
GMM	Gaussian Mixture Model
GP	GlobalPhone
GUS	Genial Understander System
HH	Homographic Heterophones
HIT	Human-Intelligence Task
HMM	Hidden Markov Model
IPA	International Phonetic Alphabet
IVR	Interactive Voice Response
LAS	Listen Attend and Spell
LM	Language Model
LPM	Linguistic Processing Module
LSTM	Long Short-Term Memory
LVCSR	Large Vocabulary Continuous Speech Recognition
ML	Machine Learning
NLE	Normalizer Linguistic Expansion
NLG	Natural Language Generation
NLP	Natural Language Processing
OOV	Out-Of-Vocabulary
pp	Percentual Points
pt-BR	Portuguese From Brazil
pt-PT	Portuguese From Portugal
RM	Replacement Map
RNN	Recurrent Neural Network
RNN-T	Recurrent Neural Network Transducer
ru-RU	Russian From Russia
SMS	Short Message Service
sv-SE	Swedish From Sweden

TTS	Text-To-Speech
UT	Unit Test
VAL	Voice Portal
WER	Word Error Rate
WERnorm	Word Error Rate over normalizable tokens
X-SAMPA	Extended Speech Assessment Methods Phonetic Alphabet

Contents

List of Figures

List of Tables

List of Equations

Chapter 1
Introduction

Data preprocessing is a crucial step in building a machine learning model. If data is fairly preprocessed, the results are consistent and of high quality (García et al., 2016). For example, modern speech recognition systems model linguistic entities at multiple levels, sentences, words, phones, and other units, using various statistical approaches (Jurafsky & Martin, 2022). The parameters of these models are usually trained on data, but their accuracy attempts to capture linguistic knowledge.

This work aims at evaluating how linguistic knowledge can influence speech technologies – Automatic Speech Recognition (ASR) and Text-to-Speech (TTS) – performance, as well as describe the linguistic preprocessing methods provided by an AI international company, Defined.ai. The Linguistic Processing Module treats raw text as input and structures it to make it usable for speech technologies that require linguistic knowledge. This module includes the Normalizer and the Grapheme-to-Phoneme (G2P) pipelines. These introductory linguistic models are essential to ensure the quality of data processing and to measure its quality before any other step in the pipeline is taken. A crucial step, though, since it ensures high-quality data processing through the pipelines. Therefore, this study's primary focus is speech preprocessing, a sub-field of Natural Language Processing (NLP) that studies speech signals and the processing methods of signals. The secondary focus of this work relies on text preprocessing, an NLP task that involves cleaning text data and preparing it for model building. The primary purpose is to clean and prepare text data for NLP-specific tasks. In short, we accomplish this by applying various techniques in a data-driven approach, with the primary goal of creating better and more efficient NLP models.

This work describes three main projects that were conducted along with our work with the Machine Learning Team within Defined.ai:

1. Normalizer expansion for European Portuguese.

2. Grapheme-to-phone model creation for Swedish and Russian.

Also, building and deploying speech processing systems is becoming more challenging with the increasing need to support multiple input and output languages (Meyer, 2021a). Therefore, this study sought to:

1. Get insight into how linguistic knowledge can impact speech technologies,
2. Understand how linguistic knowledge impacts Normalizer and G2P models in different languages, and
3. Understand how to upgrade and validate the Normalizer and G2P models in different languages.

This document is structured around six different sections. Chapter 2 contains a host contextualization presenting the company and its platform, products, and customers. We also discuss the importance of crowdsourcing and describe which job types the company uses. Chapter 3 contains a literature review that describes relevant work on phonetic and phonology concepts of two languages, speech recognition systems, and linguistic processing. Chapter 4 explains the proposed methodology. Chapter 5 details the obtained results for each of our models and discusses the obtained results regarding our initial research. Finally, Chapter 7 presents our study's conclusions and remarks on future work.

Chapter 2
Host contextualization

This thesis was conducted within the scope of an internship with Defined.ai, a technology company, a leading provider of data, models, and tools for Artificial Intelligence, impacting companies in all sectors, from healthcare and retail to finance and consumer goods. It was founded in 2015 by Daniela Braga and is currently spread to Seattle, Lisbon, Porto, and Tokyo. The startup is working with more than 50 languages, focusing on the fields of Natural Language Processing (NLP), Translation, and Computer Vision. Defined.ai works with many companies, such as Voci, MasterCard, and Canon. Further details on what the company is and its accomplishments throughout the years will be explained in Section 2.1. Afterward, a description of the company's crowdsourcing platform, its products and regular customers will be presented in Section 2.2. Furthermore, Section 2.3. will focus on the Machine Learning (ML) team as well as the tasks performed during the Internship. A description of the pipeline models used in the company's products will be introduced in Section 2.4. To conclude this chapter, an explanation of some of the company's projects and research lines can be found in Section 2.5.

2.1 Defined.ai presentation

We live in a world where software is being replaced by Artificial Intelligence. In a world where software is programmed and static, Artificial Intelligence (AI) is trained and evolves dynamically. For AI to be applied, a combination of large amounts of data with fast, iterative processing and intelligent algorithms is needed. AI is based on big data to be properly trained, and high-quality structured data is expensive and hard to obtain in a reasonable timeframe. This is where Defined.ai steps in and provides solutions for distinct products based on high-quality data.

Defined.ai is a trusted AI data partner, following ISO 27001, that hosts (Timberlake, 2014) online marketplaces for buying and selling AI tools, data, and models. The company offers

qualified services that contribute to the success of complex machine learning projects in distinct domains. Aligned with these goals, AI experts are focused on practicing and promoting an ethical data ecosystem that relies on factual dimensions for collection and validation, as well as promoting fair features, algorithms, and data. As a result, Defined.ai can be described as a community of AI experts building fair, accessible, and ethical AI systems of the future, while creating a natural interaction between people and machines.

From the company's foundation in 2015 until the present time, Defined.ai has faced challenges and has obtained several accomplishments. In the early years, in 2015, the CEO Daniela Braga founded the company in Seattle, Washington, and opened a research and development center in Lisbon. Only one year later, the company presented a smart data platform and received a seed round from investors that included Amazon Alexa Fund, Sony, Portugal Ventures, and Busy Angels. In 2017, *Neevo*, a crowdsourcing platform, was launched, and the company opened its third office in Porto. That same year, an alpha version of the Software-as-a-Service (SaaS) platform was released. Right after this, in 2018, Defined.ai closed an 11.8 million dollars Series A funding, led by Evolution Equity Partners, which led to the opening of their fourth office in Tokyo, an announcement of an Amazon Alexa Skills partnership, and product integration with IBM Watson Studio. In the first half of 2019, the company doubled its team to over 100 employees, now 36 different nationalities worldwide. They were named one of CB insights' "100 most promising AI startups". Achieved the coveted ISO 27001 certification and revealed the launch of the brand new *Neevo* app. At the end of that year, Defined.ai reached a new milestone – standing in Forbes AI 50 Most Promising Artificial Intelligence Companies. Afterward, in 2020, the company raised a 50.5 million dollars Series B round of funding, a highlighted achievement since it was the highest Series B round to be submitted by a female-founded AI company in the US. They also participated in the Forbes Top 50 Best Startup Employers, and in 2021, the company evolved from DefinedCrowd to Defined.ai. It currently hosts an online marketplace where AI professionals can buy or sell AI training data, tools, and models.

2.2 Defined.ai platform, products, and customers

The Defined.ai platform allows clients to obtain and reach high-quality structured training and validated data. Understanding how to effectively collect, prepare and test data helps unlock the total value of AI. Consequently, training data can feed AI models or train machine learning algorithms on better decisions based on high-quality data.

Defined.ai project parameters can include training data to the client's needs, ranging from managing unstructured data to letting AI professionals source it. This allows the company to offer quality guarantees along with delivering them on a global scale. Therefore, the company is the choice data partner for some of the biggest companies in the world.

As the industry expands to more natural forms of interaction with everyday devices and services, the need for data to train such applications increased the recent years. To answer these needs, crowdsourcing emerged as an attractive solution for extensive data collection efforts, allowing Defined.ai to expand in this field. As was said earlier, *Neevo* is a popular platform for crowdsourcing language services. It allows users to have access to jobs in which they provide answers and judgments on several tasks: transcribing and annotating audios, identifying named entities, recording audios, validating the pronunciation, translating sentences, classifying the sentiment(s) of a text, or extract information from images or videos. So far, the company has more than 500,000 contributors from over 70 countries who speak over 50 languages and have logged positive completion of over 200 million tasks.

Defined.ai crowdsourcing is characterized by five main factors: 1. allows individuals to contribute instantly, without having to go through the recruiting process; 2. grants security; 3. offers skills and certification; 4. provides quality and ethical training sets; and 5. recruits a skilled, local workforce based on demographic and skillset criteria. Overall, the Defined.ai workforce can help create and develop AI models with high-quality training data obtained at a fast pace, securely, and ethically aligned.

The company focuses on Speech, Natural Language Processing (NLP), Computer Vision, and Translation. These areas are available in various delivery options, including off-the-shelf data and customized collection, in over 50 languages. Clients' requirements can go from the

immediate use of off-the-shelf data, customized datasets for specific use cases, or even access to a global crowd that the clients can connect to their tools. With the help of a range of four products – DefinedData, DefinedWorkflows, DefinedSolutions, and DefinedCrew – Defined.ai helps clients speed up their time to market, obtain customized training data, optimize AI models, and simplify crowdsourcing.

The clients of Defined.ai are customers from big technology companies, such as Sony or Toshiba, banks, call centers, smaller companies, or any other company needing high-quality AI data.

2.3 Crowdsourcing

The industry has expanded to more natural forms of interaction with everyday devices and services, such as communication via natural language. As a result, the need for data to train such applications increases. One of AI's biggest challenges is replicating human reasoning and language. Furthermore, the wide range of new diverse, and heterogeneous users demands robust and unbiased solutions that perform successfully regardless of their characteristics or demographics.

It is important to include human interaction when creating high-quality AI datasets to train a machine learning model to answer these needs. Crowdsourcing emerged as an attractive solution for extensive data collection efforts, allowing it to reach a diverse crowd in an expeditiously and scalable manner. The concept of crowdsourcing involves obtaining goods and services, including ideas, voting, micro-tasking, and finances, from a large, relatively open, and often rapidly evolving group of participants (Charvát & Kepka, 2021). These participants henceforth referred to as *the crowd*, can perform micro-tasks, Human-Intelligence Tasks (HITs), that are available via Defined.ai's online platform, and distributes jobs across large numbers of people in exchange for a reward. By the micro-tasking data collection and annotation jobs to *the crowd*, datasets are likely to be completed much faster and with more accuracy and diversity. Everyday tasks approached with crowdsourcing are labeling images, translating or transcribing the text, and recording speech.

2.3.1 Job types

Clients can use Defined.ai platform to construct *projects* online using templates with flexible options. A project is a collection of one or more *jobs* as part of a single workflow. An example of a *project* would be speech collection, which has multiple steps, including collecting and validating the audio. *Jobs* are a collection of HITs to be completed by the *crowd*, and each one is considered a *job type.*

NLP and Speech Technologies are two areas in which the templates can be classified. These fit into three different types of processes: 1. Generation, which was created for when the company is building a new type of data that doesn't fit into any existing class; 2. Classification, which involves attributing a class to specific data as a whole (*e. g.* this whole text is "positive"); and 3. Segmentation, which requires attributing a class to a segment of data (*e. g.* within this text, this sentence is "positive").

Job Types					
Natural Language Processing (NLP)			**Speech Technologies**		
Generation	**Classification**	**Segmentation**	**Generation**	**Classification**	**Segmentation**
Write a Back-and-forth	Named Entity Tagging Disambiguation	Document Sentiment Analysis	Broadband	Audio Sentiment Analysis	Speech Dialogue Transcription
Text Variant Elicitation	NIF Mapping	Named Entity Tagging	Orthographic Transcription	Mean Opinion Score Test	Speech Dialogue Transcription & Correction
Text Variant Correction	Semantic Annotation	Tag the Overlapping Entities	Narrowband	Text-Audio Validation	
Validate & Correct the Back-and-Forth	Text Sentiment Analysis		Rich Transcription & Annotation	Validate the Pronunciation	
	Text Variant Validation		Scripted Speech Recording		
	Validate the Back-and-Forth		Text-Audio Correction		

Table 1. Job Types

Table 1 shows 25 main job types distributed in the areas of NLP and Speech Technologies. The most common jobs are the following:

1. **Text Variant Elicitation -** opinion on a given text, based on elicitation techniques.
2. **Text Variant Correction -** confirm the text variants and align with the given text. If it does not align, correct the variants to match.
3. **Named Entity Tagging Disambiguation -** choose the correct entity tag with limited context. The list of entities that tag the text must be disambiguated.
4. **Scripted speech recording -** record audio by reading the given text.
5. **Validating the Pronunciation -** determining whether all words in phrases are spoken correctly.
6. **Speech Dialogue Transcription -** accurately transcribe audio conversation.
7. **Rich Transcription and Annotation -** transcribe and annotate any given audio using highly customizable tags.

2.4 Defined.ai pipelines

In this chapter, we present the Linguistic Processing Module used at Defined.ai. Its goal is to treat raw text as input and structure it to make it usable for speech technologies that require linguistic knowledge, specifically targeting speech recognition and speech synthesis. In this module are included the Grapheme-to-Phoneme (G2P) and the Normalizer pipelines. These are introductory linguistic models which are essential to ensure the quality of data processing and to measure its quality. A crucial step since it ensures high-quality data processing throughout the pipelines. By describing these models, we explain the process of converting written text into its spoken form and how we transform graphemes into phonetic transcription, in a stepwise perspective.

Such linguistic models are frequently used as pre-requirements for the preparation of any corpus for ASR systems. At Defined.ai these pre-requirements are one of the company's initiatives that support ASR systems and DefinedData (one of Defined.ai products). The Machine Learning (ML) team ensures the creation and assistance of in-house solutions for G2P conversion and Normalization. Consequently, the outcome of this initiative is to create a more robust data collection pipeline and to develop pre-requirements for training data.

Therefore, the G2P is a fundamental part of many pipelines, including the STT (Speech to Text), TTS (Text to Speech), and AST pipelines.

The AST (Automatic Segmentation & Transcription) pipeline supports the delivery of thousands of hours of high-quality spontaneous speech data of different languages. This is done by providing preliminary segmentation and transcriptions. Furthermore, the information generated by the AST is also used for two crowd jobs, segmentation validation, and transcription correction.

2.4.1 Normalizer pipeline

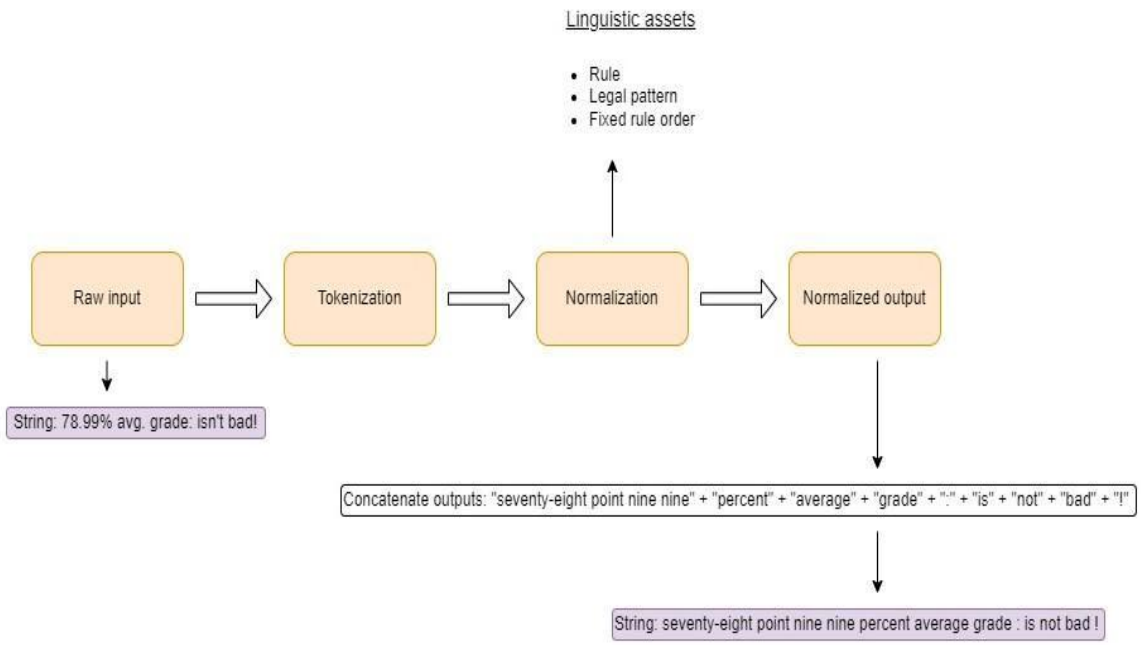


Figure 1. Normalizer Pipeline.

Figure 1 represents Defined.ai Normalizer pipeline, the process that is applied to convert written text into its spoken form that can be used by ASR systems as training data.

From left to right, we have the raw input, which is a list of sentences. The result is the normalized output, which is a list of normalized sentences, meaning in the concrete example the written form of the numbers and symbols.

Before any actual preprocessing occurs, a basic preprocessing of the input is required. This first step is text preprocessing, which is meant to minimize variability and ambiguities, which might otherwise cause downstream issues. Text cleaning turns text into a format that machine models can understand, by removing unwanted characters, abbreviations, contractions, etc. Firstly, it removes meta-tags (*e. g.*, hmtl, xml), non-printable characters, and long words or lines. Secondly, it normalizes punctuation by reconstituting words split with a hyphen at the end of a line; and unifying all quotation mark characters into single quotation mark characters. Finally, it normalizes whitespaces and Unicode[1] characters.

Before Normalization, there is a particularly crucial step that will affect the rest of the pipeline - Tokenization. The tokenizer receives raw input and separates it into tokens by considering non-printable characters and spaces (*e. g.*, line breaks, tabs, control characters) as well as logical semantic breaks, which are meant to facilitate preprocessing in the subsequent models. The tokenization is performed in two steps - Sentence parsing, and Word parsing. The following table shows an example of the tokenization process:

Input	78.99% avg. Grade: is not bad!											
Token index	0	1	2	3	4	5	6	7	8	9	10	11
Token	78	.	99	%	avg	.	grade	:	is	not	bad	!

Table 2. Tokenization process.

This tokenization process immediately turns an unstructured string into a numerical data structure suitable for machine learning.

After the Tokenization process is done, the second step is Normalization. To normalize a sentence, the normalizer attempts to apply each rule, one at a time, to each word of the sentence. When the normalizer finds a rule that can be used at a particular location, the

[1] Unicode is a character encoding system that provides a code to every character and symbol in any language.

normalizer follows the directions of that rule to produce the spoken form output and then moves on to the next word.

A rule-based normalization makes it possible to predict the output of the Normalizer since the patterns are very constant and can be accurately picked by rules. However, a new rule can be written if any current rule does not cover a particular construction.

The Normalizer now supports 12 natural languages, including English, Portuguese, German, and French. In the last year, linguists in the team have been working on other languages such as Japanese, Indi, Czech, and Korean.

Also, it provides normalized forms for numbers, dates, measurements, time, durations, and symbols. The following table illustrates the procedure to reach the final normalized output:

Token index	0	1	2	3	4	5	6	7	8	9	10	11
Token	78	.	99	%	avg	.	grade	:	is	not	bad	!
Rule #1 Abbreviations					YES	YES						
Rule #2 Real Numbers	YES	YES	YES									
Rule #3 Symbols				YES								
Normalization	seventy-eight point nine			percent	average		grade	:	is	not	bad	!
Output	seventy-eight point nine			percent	average							

Table 3. Normalization process

2.4.2 G2P pipeline

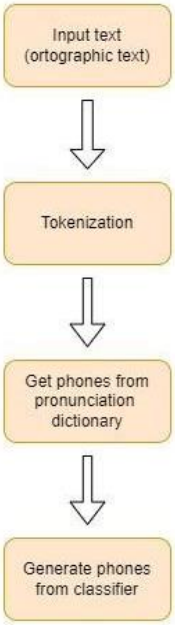


Figure 2. Grapheme-to-phoneme Pipeline.

Figure 2 shows Defined.ai G2P (Grapheme-to-Phoneme) pipeline and the steps involved in developing phonetic transcription that can be used to create and improve ASR systems.

The Defined.ai G2P predicts the phonetic transcription of any given written word. This tool takes raw, orthographic text as input and provides its phonetic transcription as the output.

From top to bottom, we have the input text. Similarly, as with the Normalizer pipeline, Tokenization is a necessary aspect before any other step. To obtain tokens, tokenization is performed on the Corpus (text data). Following that, the tokens are used to check vocabulary. We can extract the pronunciation of each word from the lexicon using this method. The next stage is generating phones from the classifier, which means that when a word is not recognized, the Classifier can predict the pronunciation of that specific word. The following figure shows how the input is processed:

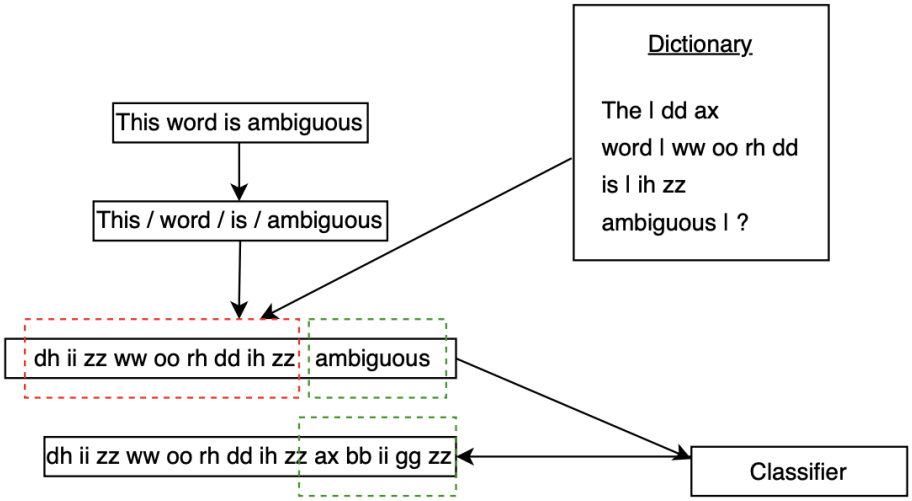


Figure 3.Pronunciation prediction process.

In the first line, we can see the orthographic text as the input, which is tokenized. After that, we will get the corresponding phonetic transcription of each word that was taken from the pronunciation lexicon. If a word is not recognized by the lexicon (*e. g.*, ambiguous) the Classifier will then predict the phonetic transcription of that given word.

The G2P now supports 18 languages, including English, Portuguese, Japanese, among others. During the internship, we have been upgrading it with two more languages, Swedish and Russian as will be described in the Section 2.5.

2.5 Focus of the internship: ML team and tasks done

During the internship, the main work was conducted within the ML team, composed of speech engineers, computational linguists, and data scientists. The ML team's mission at Defined.ai is to supply ML solutions that enable the development of scalable, available, and high-quality data products for AI. Also, they are continuously expanding the company's transdisciplinary ability in innovative AI technologies. As a result, the team is the company's point of reference for Speech, Natural Language Processing (NLP), Computer Vision, Data Science, and AI insights, benchmarks, and baselines.

The primary focus of the internship was on Speech Processing, which studies speech signals and the signal processing methods. It is, thus, a process by which speech signals are interpreted and understood.

The secondary focus of the internship was on Text Preprocessing, an NLP task that involves cleaning text data and preparing it for model creation and training. The main purpose was to clean and prepare text data for NLP-specific tasks, by applying a variety of techniques on a corpus of text with the goal of creating better and more efficient preprocessing NLP models.

Along with the internship, several tasks were performed, but for the context of this thesis I will focus on three main projects:

1. Normalizer expansion and improvement for pt-PT
2. Grapheme-to-Phoneme (G2P) models for sv-SE and ru-RU

Apart from the focus on linguistic preprocessing tasks, the internship created an opportunity to broaden our understanding of the ML industry, including the procedures involved, the linguistic challenges that keep cropping up, and how team members cooperate to overcome them.

In Section 4.1 and 4.2, the two main projects mentioned above will be described in detail considering the impact they had on the company.

2.6 Summary

This chapter addressed our host entity, Defined.ai, regarding its platform, products, and customers. In addition, we discuss the importance of using crowdsourcing as a service while explaining which job types the company includes on its platform.

Chapter 3
Literature Review

This chapter focuses on a range of three main subjects: Phonetics and Phonology, Speech Technologies, and Linguistic Processing. For this purpose, this chapter is structured as follows: Section 3.1 describes phonetic and phonology concepts of Swedish and Russian Section 3.2 reviews state of the art of speech technologies focusing on automated speech recognition, text-to-speech, and dialogue systems or, more recently, conversational agents or personal assistants; Section 3.3 introduces terminology and background on the linguistic preprocessing for Multilingual Grapheme-to-phoneme and Text normalization.

3.1 Phonetic and phonology concepts

In this chapter, we introduce the main phonetic and phonological aspects of two distinct languages – Swedish and Russian. For the context of this thesis, we focus mainly on phonetic aspects of such languages, since we aim to describe and identify speakers' real pronunciations, based on data-driven approaches, in order to implement this knowledge on the G2P model. Thus, this chapter will mostly cover phonetic transcription and phonological processes that were strictly chosen regarding their relevance to the G2P.

We begin with a key element of both speech recognition and text-to-speech systems: how words are pronounced in terms of individual speech units called phones. Following (Jurafsky & Martin, 2022) a speech recognition system needs to have a pronunciation for every word it can recognize, and a text-to-speech system needs to have a pronunciation for every word it can say. We model the pronunciation of a word as a string of symbols that stand for phones or segments. A phone is a speech sound and phones are represented with phonetic symbols that bear some resemblance to a letter in an alphabetic language, like English or Portuguese. In the context of this thesis, we use three different alphabets for describing phones: the

International Phonetic Alphabet (IPA)[2], the Extended Speech Assessment Methods Phonetic Alphabet (X-SAMPA)[3] and the DC-Arpabet (phonetic alphabet created by Defined.ai). IPA was created in the late 19th century to describe the sounds of all human languages while using a set of transcription principles and its phones. X-SAMPA (a variant of SAMPA) is a phonetic writing system created in 1995 with the intention to encode the 1993 IPA chart in ASCII. On the other hand, the DC-Arpabet was specifically designed by Defined.ai for US English, while using ASCII symbols. Throughout the years, linguists at Defined.ai have been expanding the DC-Arpabet to more languages, such as Portuguese, German, French, Hindi, among others. Table 4 shows some examples of an American English phone set using DC-Arpabet symbols for transcribing its consonants, semivowels, vowels, and diphthongs together with their IPA and X-SAMPA equivalents.

No	**Phonetic Alphabet Symbols**			**Classification**
	IPA	**DC-Arpabet**	**X-SAMPA**	
1	[b]	[bb]	[b]	consonant
2	[ð]	[dh]	[D]	consonant
3	[s]	[ss]	[s]	consonant
4	[ʃ]	[sh]	[S]	consonant
5	[ʒ]	[zh]	[Z]	consonant
6	[h]	[hh]	[h]	consonant
7	[j]	[yy]	[j]	semi-vowel
8	[w]	[ww]	[w]	semi-vowel
9	[ə]	[ax]	[@]	vowel
10	[ɚ]	[er]	[@`]	vowel
11	[ɑ]	[ah]	[A]	vowel
12	[ɛə]	[ex]	[E@]	diphthong
13	[ɔɪ]	[oi]	[Oi]	diphthong
14	[ʊə]	[ux]	[U@]	diphthong

Table 4. Examples of the American English phone set using, IPA, DC-Arpabet, and X-SAMPA symbols.

The DC-Arpabet currently contains 239 indexed unique symbols, each composed of at least 2 alphanumerical characters, plus an optional 1 to 3 alphanumerical characters. The symbols

[2] (Armstrong & Meier, 2005)
[3] (Wells, 1995)

are represented by '##' and '_', and whenever an audio includes a silence, short pause, unintelligible sounds, vocal noise or music we represent it using the respective abbreviation shown in Table 5. Building the G2P model for a language also requires a list of characters used to spell that language, that is, its graphemes.

No	Phonetic Alphabet Symbols			Classification
	IPA	DC-Arpabet	X-SAMPA	
1	##	##	##	sentence break
2		_		skipped phoneme
3		sil		silence
4		sp		short pause
5		zun		unintelligible
6		zvn		vocal noise
7		zmu		music

Table 5. Symbols and non-phonetic sounds representations.

3.1.1 Swedish

This section deals with the phonetics and phonology of Standard Swedish from a synchronistic perspective. As was said at the beginning of this chapter, we want to focus on describing the authentic pronunciation of speakers. First, we decided to choose one single variant among all existing ones in the Swedish language – Standard Swedish. This variant evolved from the Central Swedish dialects, and most Swedes speak it. Herewith we can predict how most speakers pronounce Swedish. Thus, in this section, we start by briefly describing the Swedish vowel and consonant inventory, where we discuss phenomena crucial to measuring the impact on the G2P. Following, we introduce two frequent features of this language – quantity and retroflexion – which were important in the decisions we had to make for the Swedish G2P model.

The Swedish orthographic alphabet consists of nine vowels: <a, e, i, o, u, y, å, ä, ö>. (Riad, 2014) considers nine distinct vowel phones. Each vowel occurs with long and short variants. The author finds these variants allophones of the same phoneme. The long vowels are [iː, yː, eː, ɛː, øː, ʉː, uː, oː, ɑː] and the short vowels [ɪ, ʏ, ɛ, ø, ɵ, ʊ, ɔ, a], respectively.

In Standard Swedish, several vowel qualities are distinguished in unstressed syllables. The phones /e/ and /ɛ/ neutralize in the short variant, as [ɛ]. The alternations between long and short vowels provide important cues to the phonemic system shown in Section 4.3. Thus, the author defends that the lowering of /ø/, and /ɛ/ before a retroflex motivates the height separation between /a/, which is [low], and /ø/ and /ɛ/ which are [mid]. The two main short allophones of /e/ and / ɛ/ are neutralized as [ɛ]. This results in eight short vowel allophones to match the nine long vowel allophones. However, many dialects have nine long vowels and nine short vowels (Riad, 2014). In some varieties of Standard Swedish, a similar neutralization occurs for short /ø/ and short /ʉ/, where some young speakers have neutralization as [ɵ].

Also, in many cases <e> and <ä> as in *sett* and *sätt* coincide and are both pronounced /e/. This can lead to the mistaken belief that there are only eight short vowels and that [e] and [ɛ] are allophones. Yet, in Standard Swedish, /e/ and /ɛ/ are treated as phones in Standard Swedish.

The consonant inventory contains 18 different phones. Sixteen of these occur with both a short and a long variant, and that distinction is phonological (represented in Table 6 with a raised mora - /ˑ/). In contrast to vowels, the opposition length of consonants does not carry any meaning. The consonant system has a double specification of aspiration and voicing in the obstruents. (Riad, 2014) focuses on the qualitative contrasts of the Swedish consonant system, arranged according to the place of articulation and manner of articulation:

	labial, labiodental	**dental, alveolar**	**alveolar, palatal**	**velar**	**glottal**
oral stop	p pˑ	t tˑ		k kˑ	
	b bˑ	d dˑ		g gˑ	
fricative.	f fˑ	s sˑ	ç		h
fricative/retroflex			ʂ ʂˑ		
fricative/approximant	v vˑ				
nasal stop	m mˑ	n nˑ		ŋ ŋˑ	
lateral		l lˑ			
apical trill		r rˑ			

Table 6. Standard Swedish consonants inventory mofified and extracted from (Riad, 2014): 49.

Standard Swedish has palatal and velar voiceless fricatives: /s/, /ʂ/, /ɕ/. The phones [ɧ] and [ɕ] are very similar, although the most prominent phonetic difference lies in its place of articulation. While /ɕ/ has a stable place of articulation, /ʂ/ is subject to contextual conditioned allophony in Standard Swedish, as well as a wide-ranging sociolinguistic variation. In onset position, /ʂ/ can have four different variation - [ɧ], [ɧʷ], [x], [ʂ] – while in postvocalic position the realization of /ʂ/ is mostly [ʂ] or [ʂː]. Thus, the most common prevocalic realization is [ɧ]. /ʃ/ is also used in some Swedish dictionaries (*e. g., Svensk Ordbok*[4]).

One of the most salient features of North Germanic standard varieties is the quantity system. Stressed syllables are invariably heavy, due to a prosodic condition. This condition is met in either the vowel alone, or in a combination of the vowel and the following consonant. In a stressed syllable one segment must be long, either the vowel or the consonant, but not at the same time. Vowels and consonants thus occur in long and short variants, and it is primarily in terms of quantity that these segmental distinctions are made and described.

There are qualitative differences within vowel pairs. Each long vowel has a short counterpart. The long and short consonants in a pair are naturally much more similar in quality. Most consonants have long and short pairs, but there are a few that exhibit a defective quantitative distribution. Two phones never occur directly after a stressed vowel, namely /h/ and /ɕ/, and hence lack long variants altogether. The segments /j/ and / ŋ/, on the other hand, are always long in a postvocalic coda position, provided that the syllable is stressed. The phoneme /ŋ/ never occurs word-initially, but may occur intervocalically and as onset in unstressed positions.

Syllable weight in stressed syllables is phonetically and phonologically clear. Any stressed syllable is bimoraic, where a long vowel is bimoraic, and a short vowel monomoraic. A long consonant is (mono)moraic, and a short consonant is non-moraic. If the vowel is long, then

[4] https://svenska.se

all is fine. If the vowel is short, then the following consonants must be long. In accordance with (Riad, 2014) most of the other Germanic languages lost consonant quantities early on and this has led to rather different quantitative phonologies. In this section, we shall assume that quantity is distinctive in consonants, but some consonants have lexical length, while others become grammatically lengthened or shortened by syllabification. For the vowels, quantity is predictable from prosodic context, when a syllable is stressed or when there is quantitative information (*e. g.*, lexical or positional) in the following consonant.

Another striking feature of Standard Swedish is retroflexion. The retroflexion rule creates retroflex sound when two contiguous segments (/s, t, d, n, l/ with a preceding /r/) converge into one element. The output /ʂ, ʈ, ɖ, ɳ, ɭ/ is phonologically distinct from the input segments. Thus, retroflex consonants can appear in most simple words (*e. g., framfar**t**, rampaging),* but can also occur in other articulatory patterns – word boundaries, inflections, compounds, derivations. In word boundaries, a retroflex consonant emerges if the final letter of a word is an <r> and the initial letter of the following word is <t, d, s, l, n>.

Swedish form	**Phonetic transcription**	**English translation**
vår triumf	/vo:ʈriɵmf/	our victory
hur mår du	/hʉ:rmo:ɖɵ/	how are you
under sängen	/ɵndeʂɛŋen/	under the bed
eller nej	/ɛleɳɛj/	or not
hur ledsam	/hʉ:lesam/	how sad

Table 7. Standard Swedish retroflexion rule.

As for flections, when the genitive <s> is attached to a word ending with <r>, the retroflex /ʂ/ is used (e.g, *Peters hus, /peteʂhʉ:s/, Peter's house*). When a verb ends with a final <r> the retroflex consonants /ɖ/, /ʈ/ occur (*e. g., stör-de, /stø:ɖ /; stör-t, /stø:ʈ/*). Furthermore, this rule also applies to past participles and nouns. Thus, retroflex consonants also occur in compound words (*e. g., vårdag, /vo:ɖa:g /, spring day*) and derived words (*varsam, /va:ʂam/, careful*).

3.1.2 Russian

This section deals with the phonetics of Standard Russian from a synchronistic perspective. We start by briefly outlining the Russian vowel inventory and consonant inventory. Following, we describe a frequent characteristic feature of Russian – palatalization. Finally, we describe and discuss the value of two special signs in Russian phonetic and phonological system. Therefore, in order to make a plan for training a Russian G2P model, we also need to discuss which phonetic representations are best represented of Russian graphemes, how to treat iotated vowels and if we should make a contrast between palatalized consonants and non-palatalized consonants.

The modern Russian alphabet consists of 33 letters: 20 consonants (<б, в, г, д, ж, з, к, л, м, н, п, р, с, т, ф, х, ц, ч, ш, щ>), ten vowels (<а, е, ё, и, о, у, ы, э, ю, я), a semivowel (<й>), and two modifier letters or signs (<ь, ъ>) that alter pronunciation of a preceding consonants or a following vowel.

In most analyses, the Russian vowel inventory contains five vowel phones: <i, e, a, o, u>. However, studies on Modern Russian (Yanushevskaya & Bunčić (2015), and (Timberlake (2014)) claim a sixth vowel <ɨ>. Each one of these vowels is realized as a rich set of allophones ruled by stress and phonological environment. In most cases, these vowels merge into two or four vowels when stressed: /i, u, a/ after hard consonants and /i, u/ after soft consonants.

In orthography, each vowel is represented by two letters. This happens so we can distinguish the not-iotated vowels and the iotated vowels. Vowels in Russian do not have a phonemic distinction of quantity; there are no words distinguished by, for example, a long [aː] as opposed to a short [a].

Russian has 33 different phones. Most of the consonants occur with both a palatalized and a non-palatalized version. The remaining consonants have a single version – the velars are palatalized before front vowels; palatals are either invariably palatalized (*e. g.*, j) or invariably not. The consonant inventory for Russian is given in Table 8.

		labial	dental	alveolo-palatal	palatal	retroflex	velar
stop	**voiceless**	p pʲ	t tʲ				k
	voiced	b bʲ	d dʲ				g
fricative	**voiceless**	f fʲ	s sʲ	ɕɕ		ʂ	x
	voiced	v vʲ	z zʲ			ʐ	
affricate			t͡s	t͡ɕ			
nasal		m mʲ		n nʲ			
lateral			l lʲ				
rhotic							r rʲ
glide					j		

Table 8. Russian consonants inventory modified and extracted from Timberlake (2014): 53.

Thus, ш and щ share a very similar way of pronouncing their sounds. The letter ш has the phonetic value of [ʃtʃ] (a "palatalized /ʃ/" followed by a sound similar to [tʃ], in which the stop element, represented by t, is weak) or [ʃʃ] (a long "palatalized /ʃ/"). Either of these pronunciations of ш is regarded as correct, but it is common for any speaker to use only one of them. The letter щ has a phonetic value that can switch between [tʃ] (a weak t followed by a "palatalized /ʃ/") and [tɕ] (a weak t followed by a [ɕ]). Either of these is correct but it is pronounced differently depending on the region of Russia (Timberlake, 2014).

One of the most characteristic features of Russian consonantal phonology is that most sounds have both a palatalized and a non-palatalized phonological segment. In accordance with (Bondarko, 2005), palatalized consonants are referred to as soft and non-palatalized consonants as *hard*. Palatalization is an articulation of a consonant in which the blade of the tongue moves toward the hard palate. For example, when a non-palatalized consonant is pronounced, the tip of the tongue is touches near the teeth, while the middle of the tongue lies low in the mouth. In contrast, when the palatalized consonants are pronounced, the tip of the tongue touches behind the upper teeth, and the blade and the middle of the tongue are raised towards the hard palate. Most consonant articulations in Russian have two forms, with or without palatalization. Thus, palatalization in Russian is indicated by adding a diacritic to the phone. According to IPA, we use a palatalized diacritic (/ʲ/) when referring to palatalized consonants (*e. g.*, /bʲ, dʲ, gʲ/). Palatalization is contrastive in word-final and in heterogenic medial coda positions.

Some examples illustrating the palatalization contrast extracted from (Padgett, 2001) are given in (1). Contrasts (1)a-b are prevalent in the language, while (1)c is more limited due to assimilations and neutralizations in that context.

1. a) before back vowels

mat	foul language	mat'	crumpled
rat	glad	r'at	row
vol	ox	v'ol	he led
nos	nose	n'os	he carried
suda	court of law	s'uda	here, this way

b) word-finally

mat	foul language	mat'	mother
krof	shelter	krof'	blood
ugol	corner	ugol'	(char)coal
v'es	weight.	v'es'	entire

c) before another consonant

polka	shelf	pol'ka	polka
tanka	tank		
v'etka	branch		
gorka	hill		

Russian has two modifier signs with no phonemic value. The soft sign *Ь* signals the presence of a soft consonant, and the hard sign *Ъ* signals the presence of a hard consonant. Therefore, they can be used between a consonant or a vowel (*Ь*) and between a consonant and a vowel, between two consonants, or at the end of a word after a consonant (*Ъ*)

In Russian *Ь* has much wider usage than *Ъ*; *Ь* can be used at the end of words or in between two consonants, and it indicates that the preceding consonants are soft. Neither *Ь,* or *Ъ* can be a stand-alone letter or the first letter in a word.

At the same time, there are some letters in Russian that are always hard or always soft and will sound the same way, whether there is a soft sign (Ь), or not. The consonants < ж, ш, ц > are always hard (if these consonants are followed by a soft sign, the sign cannot soften the consonant and serves a purely grammatical purpose. The consonants < ч, щ, й > are always soft. A soft sign following these consonants, once again, serves only a grammatical purpose.

3.2 Speech Technologies

3.2.1 ASR

As stated by Alasadi & Deshmukh (2018), speech recognition is an important field that is constantly being developed as an interdisciplinary subfield of computational linguistics. Speech recognition creates technology and methods that empower the acknowledgment and understanding of natural language into a computer understandable language. This is most generally known as Automatic Speech Recognition (ASR).

Accordingly, in this section, we start by presenting a brief historical overview of automatic speech recognition, from the early days until the present. Then, we introduce a traditional and a more modern approach to ASR, followed by the main challenges it faces today. Finally, we introduce an evaluation metric used for speech recognition systems.

3.2.1.1 ASR: Historical overview

Throughout the years, ASR has become more reliable and easier to handle as a result of significant advancements in computer technology and informatic techniques. Thus, the number of applications has increased significantly.

Speech recognition has its origins in early 1950s research at Bell Labs. Early systems were limited to a single speaker and only had a few dozen words in their lexicon, numbers for telecommunication purposes mostly. Modern speech recognition systems, on the other hand, have come a long way from their predecessors, which had large vocabularies in many languages and could recognize connected speech from a variety of speakers.

The first speech recognition systems were focused on numbers. In 1952, The "Audrey" system, developed by Bell Laboratories, could find a single voice speaking digits clearly. Ten years later, IBM introduced “Shoebox,” which understood and responded to 16 words in English. Other nations developed hardware to recognize sound and speech, and by the end of the 1960s, technology had advanced to the point where it could support words with four vowels and nine consonants.

In 1966, a statistical method known as the "Hidden Markov Models (HMM)" became one of the breakthroughs in the ASR field and has endured as the state of the art for some time. One of the first applications of HMMs was speech recognition, starting in the mid-1970s. HMM was the most successful approach, and hence the most used method for the classification stage of an ASR system (Hennebert et al., 1994). It is based on the modeling of the temporal sequence of spectrums in terms of Markov chains, to describe how one sound transits to another (Juang & Rabiner, 1991) The HMM calculates the probability of unknown sounds becoming words rather than using words and searching for sound patterns.

Speech recognition vocabulary grew from a few hundred to several thousand words in the 1980s. In 1997, The world's first "continuous speech recognizer" was released, in the shape of Nuance’s Dragon Naturally Speaking software. It is still in use today, capable of understanding 100 words per minute. The Voice Portal (VAL), an interactive voice recognition system that could be accessed by dialing in, was also accessible through BellSouth.

By 2001, the accuracy rate of speech recognition technology had reached approximately 80%. Google Voice Search was a step forward that resulted in substantial changes. As it is an application, millions of individuals were able to use speech recognition. It was also significant since Google could collect data from billions of searchers, which could help predict what a person is saying.

Indeed, speech recognition became highly popular, especially when Siri was introduced by Apple in 2011. Today, several speech-based commercial applications exist, ranging from

home automation systems, virtual smartphones assistants, or voice interaction systems in automobiles. This will be further discussed in section 3.2.3.

3.2.1.2 Approaches to ASR

Classical machine learning models can be classified into two different approaches: Generative[5] and Discriminative[6] approaches. For the past decades, speech recognition has mostly used a Generative approach. Two popular methods that are based on this approach are the HMMs (Hidden Markov Models) and the Gaussian mixture models (GMMs). Thus, the basic framework of a speech recognizer system consists of three major stages: capture, transducing, and decoding. The acoustic model, lexicon model, and language model are combined during the transducing step. Connecting these pieces, we obtained an ASR framework that includes the components described in Figure 7.

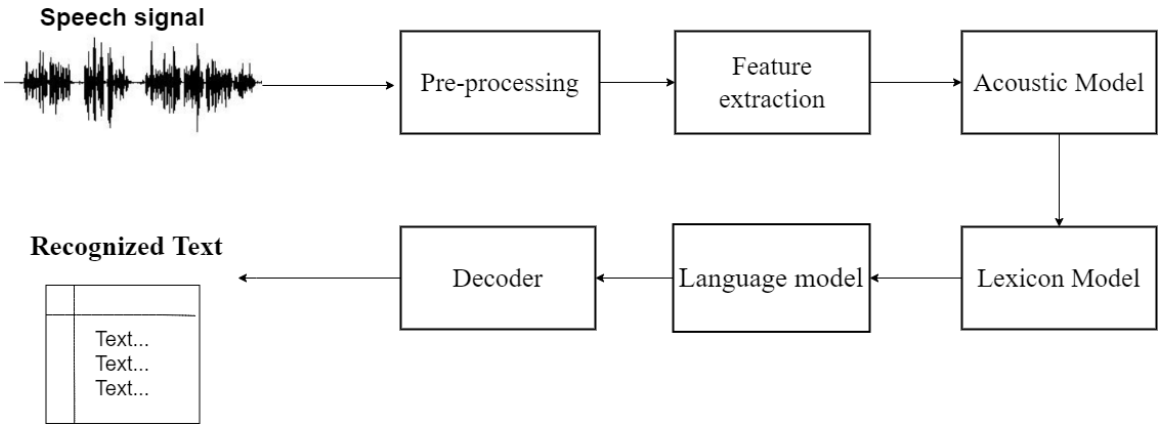


Figure 7. Basic framework of a speech recognizer system.

1. **Speech signal -** extract information on the properties of the speech signal.
2. **Pre-processing -** transforms the speech signal before any information is extracted by the feature extraction stage. In fact, the functions to be implemented by the pre-processing stage are also dependent on the approach that will be employed at the feature extraction stage.

[5] Generative approach: used to learn each language and determine which language the speech belongs to.
[6] Discriminative approach: used to determine the linguistic differences without learning any language.

3. **Feature extraction -** extracts a set of predetermined attributes from the processed speech input. In other words, it transforms the input waveform into a sequence of acoustic feature vectors. Each vector represents the information in a small-time/frames window of the signal.
4. **Acoustic Model -** used to recognize phones rom obtained acoustic features. It computes the likelihood of the observed spectral feature vectors given linguistic units (words, phones, subparts of phones).
5. **Lexicon Model -** used to identify the most probable sequence of phones. Typically, an HMM lexicon, which is a list of word pronunciations, each one represented by a string of phones.
6. **Language Model -** consists of various kinds of knowledge related to a language, such as syntax and semantics. This model is required when it is necessary to recognize the phones that make up the input speech signal and to move it to either trigram, or sentences.
7. **Decoder -** takes the information provided by the acoustic model (AM), plus an HMM lexicon of word pronunciations, and combines it with the language model (LM). The decoder outputs the most likely sequence of words.

At the present time, the so-called classical approaches gave rise to two main approaches to ASR: a traditional hybrid approach and an end-to-end Deep Learning approach. The traditional hybrid approach has dominated the field for the past ten years. Because of the extensive research and training data available, there is more understanding of how to build a robust model(Vielzeuf & Antipov, 2019).

On the other hand, in accordance with (Kurata et al., 2019) an end-to-end Deep Learning approach is a new paradigm in neural network-based speech recognition that has several advantages. Traditional “hybrid” ASR systems, which consist of an acoustic model, a language model, and a lexicon model, each of which might be sophisticated, require independent training of these components. In contrast, E2E ASR is a unified method with a much simpler training pipeline and models that perform at low audio frame rates. This reduces the amount of time spent on training and decoding. Thus, the same author revealed that across a variety of speech recognition tasks, E2E ASR systems have matched the

accuracy of hybrid ASR systems. This is due to the use of sophisticated neural network architectures.

A common end-to-end Deep Learning architecture is the encoder-decoder, which is implemented with RNNs (Recurrent Neural Network). The next figure illustrates the standard encoder-decoder architecture, also known as the attention-based encoder-decoder (AED) or listen attend and spell (LAS) (Jurafsky & Martin, 2022).

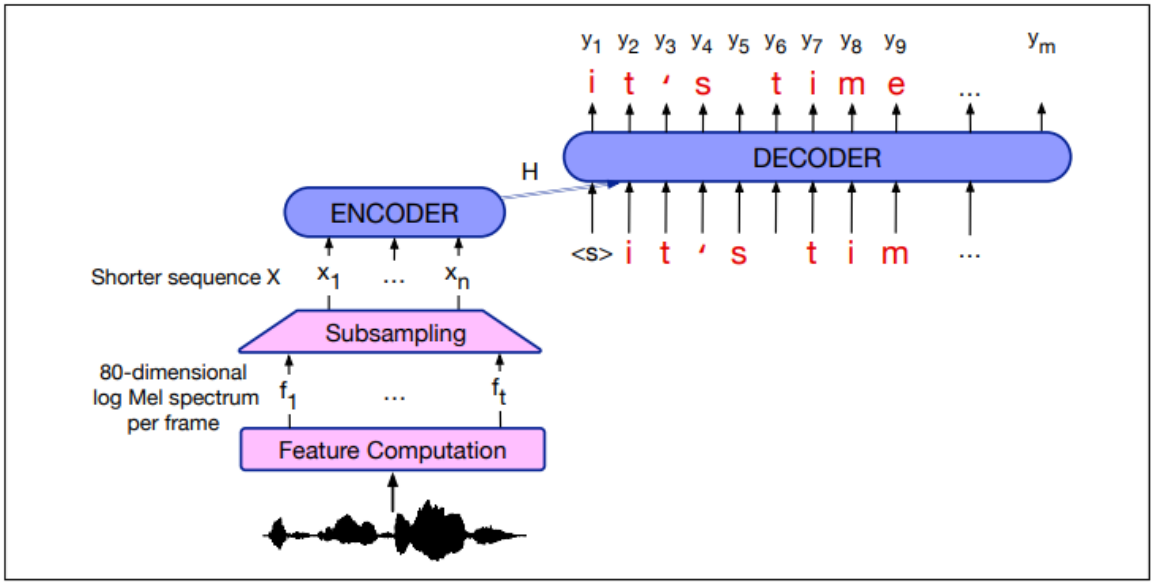


Figure 4. Encoder-Decoder Architecture extracted from (Jurafsky & Martin, 2022): 26:9.

The first component of this architecture is an encoder. Its function is to accept a single data element of the input sequence at each step, process it, collect information for that element, and transfer it forward. The second component is a decoder, it gives the entire sentence predicting an output at each step.

Several other approaches have also appeared , such as Connectionist Temporal Classification (CTC) and the Recurrent Neural Network Transducer (RNN-T). Jurafsky & Martin (2022) concluded that attention-based models offer higher accuracies, while CTC models are more adaptable to streaming, allowing them to generate graphemes online rather than waiting for the audio input to finish.

Errattahi & el Hannani (2017) concluded that the benchmarks have shown that the Deep Neural Networks (DNN) and Convolutional Neural Networks (CNN) have proven their efficiency on several Large Vocabulary Continuous Speech Recognition (LVCSR) tasks by outperforming the traditional Hidden Markov Models. LVCSR is characterized by the authors as being the most challenging task in ASR. Recently, Rajadnya (2020) developed an application for continuous speech recognition based on DNN-HMM and Deep Belief Network (DBN) algorithms. This proved that using DNN-HMM with DBN supplies better accuracy than using typical GMM-HMM systems. Deep learning is quickly becoming a standard approach for speech recognition, having effectively replaced all the classical approaches.

3.2.1.3 ASR challenges

One of the main challenges of ASR is the persistent quest for human accuracy levels. Both ASR approaches, traditional hybrid and end-to-end Deep Learning, have become significantly become more accurate, however, neither can claim human accuracy. The main characteristics in speech recognition systems that can influence recognition systems are speaker-dependent or speaker-independent models, acoustic models, vocabulary, and language models.[7] Thus, system failures are caused by issues such as a noisy environment, distinct pronunciation of one word by two different speakers, and distinct pronunciations of one word by two different speakers. Each of these issues must be resolved to get perfect recognition.

Recently, bias and gender issues in data have also deserved attention from the community. Gender and racial bias are a concept where models and algorithms do not provide optimal services to people of a specific gender or dialect. The disparity of accessible data for both genders and dialects has been proven in recent studies to be its main cause. Brasoveanu et al., (2020) found bias against women in the performance of speech recognition systems by analyzing the gender representation in different corpora. Bias was also identified against

[7](He, 2021)

dialect groups; dialect speakers have lower ASR performance than speakers of standard pronunciations (Wassink et al., 2022).

3.2.1.4 Evaluation metrics: Word Error Rate, Accuracy, Precision, Recall and F1 score

The industry standard evaluation metric for speech recognition systems is the word error rate (WER) (Jurafsky & Martin, 2022). Concerning the ASR models – Normalizer and G2P– these are performed by using WER and the following standard performance metrics - Accuracy, Precision, Recall, and F1 score (Makhoul et al., 1999).

In accordance with Jurafsky & Martin (2022), the word error rate is based on how much the word string returned by the recognizer (the hypothesized word string) differs from a reference transcription. Thus, WER is the proportion of transcription errors that the ASR system makes relative to the number of words that were said. The lower the WER, the more accurate the system.

$$Word\ Error\ Rate\ = 100\ \times \frac{Insertions + Substitutions + Deletions}{Total\ Words\ in\ the\ Correct\ Transcript}$$

Equation 1. Word Error Rate formula.

Accuracy is also metric for evaluating classification models. It answers the question: "Overall, how often is our model correct?". Thus, Accuracy can tell us immediately whether a model is being trained correctly and how it may perform generally, where 1 represents total accuracy.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

Equation 2. Accuracy metric formula.

Precision is defined as the number of true positives divided by the number of true positives plus false positives. This formula is used to understand the model's accuracy.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Equation 3. Precision metric formula.

Recall is described as the number of true positives divided by the number of true positives plus false negatives. That is, it calculates the true positives by anything that should have been predicted as positive.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Equation 4. Recall formula

F1 score, also known as F1, is a measure of a model's accuracy on a dataset. F1 score is a way of combining the precision and recall of the model, and it is defined as the harmonic mean of the model's precision and recall. A perfect model has an F1 score of 1.

$$F1 = \frac{2(Precision * Recall)}{Precision + Recall}$$

Equation 5. F1 score metric formula.

3.2.2 TTS

For centuries, the thought of machines speaking has caught the interest of researchers and creative thinkers. One of the earliest fields of speech and language processing is speech synthesis. Text-to-speech (TTS) was initially used in reading systems for the blind, where a machine would read some text from a book and convert it into speech. These early systems had a rather mechanical sound. Today sophisticated systems exist that help human computer interaction. The main uses of TTS today are in call-center automation, reading text as in SMS, weather reports, travel directions and a wide variety of other applications.

Accordingly, in this section we start by presenting a brief historical overview of TTS. Then, we introduce its main approaches, followed by the main challenges it faces today.

3.2.2.1 TTS: Historical overview

TTS has its origins in the early 1950s with the development of three paradigms of waveform synthesis: formant, articulatory, and concatenative synthesis. Formant synthesizers were originally created to simulate human speech by generating artificial spectrograms. The most well-known formant synthesizer was the Klatt formant synthesizer and its successor systems, the MITalk system and the Klattalk software. The second paradigm, articulatory synthesis, models the natural speech production process, creating a synthetic model of human articulators – lips, tongue, velum, pharynx, vocal cords, etc. – and making it speak (Palo, 2006) The third paradigm, concatenative synthesis, was first proposed at Bell Laboratories. It produces artificial speech by concatenating prerecorded units of speech such as phones, diphones, syllables, words, and sentences (Khan et al., 2016).

Afterward, a theoretical model based on diphones was proposed, although such diphone synthesis models were not implemented until the 1960s and 1970s (Lenzo & Black, 2000). Diphone synthesis creates machine speech by blending diphones, single-unit combinations of phones and the transitions from one phoneme to the next (*e. g.*, the /c/ and /ae/ sound in the word *cat*).

In the 1980s and 1990s, the unit selection synthesis, also referred to as corpus-based synthesis, was developed. Whereas diphone synthesis was using a second processing method to add length and pitch, unit selection synthesis skips that step and uses a large database. It segments each recorded utterance into individual phones, syllables, morphemes, words, phrases, and sentences that already have the duration and pitch information (Mundada et al., 2015)

While most early TTS systems used phones as input; development of text analysis components of TTS came later, drawing on NLP. At the present time, TTS uses Deep Learning techniques to generate speech.

The rise of voice computing has led to a growing range of applications for TTS devices. Today, we have independent virtual assistants such as Apple's Siri, voice guidance and navigations tools, conversational interactive voice response (IVR) systems (customer service call centers), among others. Even though there are a variety of signal processing technologies that are being used to create the synthetic voice, the best AI-created TTS voices still start with a human speaker. Therefore, recent studies are leading to TTS voices that speak with emotional expression (Rui et al., 2009), singles voices in multiple languages, and higher audio quality.

3.2.2.2 Approaches to TTS

The goal of text-to-speech (TTS) systems is to map from strings of letters to wave forms. In contrast with ASR, basic TTS systems are speaker dependent. They are trained to have a consistent voice, on much less data, but all from one single speaker. In this section we will present and briefly describe some of the most popular and innovative techniques used in TTS systems.

Earlier text-to-speech systems are classified into four categories: Concatenative systems, Diphone systems, Formant systems, and Articulatory systems. Later, with the advancement of statistical modelling, the statistical approach gave rise to Parametric synthesis (Delić et al., 2017). Modern systems use Deep Learning techniques.

The concatenative speech synthesis method was the most popular method. It involves the production of artificial human-like speech from pre-recorded units of speech by phones, diphones, syllables, words or sentences. However, developing robust concatenative systems require large corpora to collect all forms of spoken words with different combinations of emotions, prosody, stress, etc.[8] The architecture of a concatenative system consists of two main stages: Text analysis and Waveform Synthesis. The text normalization, phonetic analysis, and prosodic analysis are combined during the text analysis. Figure 5 shows an architecture for concatenative synthesis (Taylor, 2007).

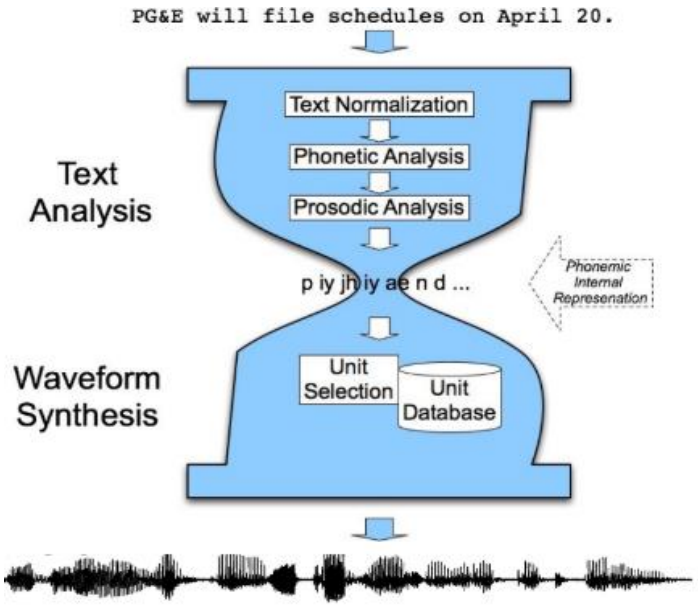


Figure 5. TTS architecture for concatenative speech synthesis.

1. **Text normalization** - breaks the input text into sentences, and deals with abbreviations, numbers, time, dates, etc.
2. **Phonetic analysis** – produces a pronunciation for each word in the normalized word strings from text analysis. It creates a large pronunciation lexicon.
3. **Prosodic analysis** – computes an abstract representation of the prosodic structure, prominence, and tune of the text.

[8] https://medium.com/@saxenauts/speech-synthesis-techniques-using-deep-neural-networks-38699e943861

In contrast, more modern TTS systems consist of two different components: an encoder-decoder model and a vocoder. The encoder-decoder model is used for spectrogram prediction, mapping from strings of letters to spectrograms, to a waveform and finally to an audio signal. The following figure illustrates the process of mapping a string to an audio signal:

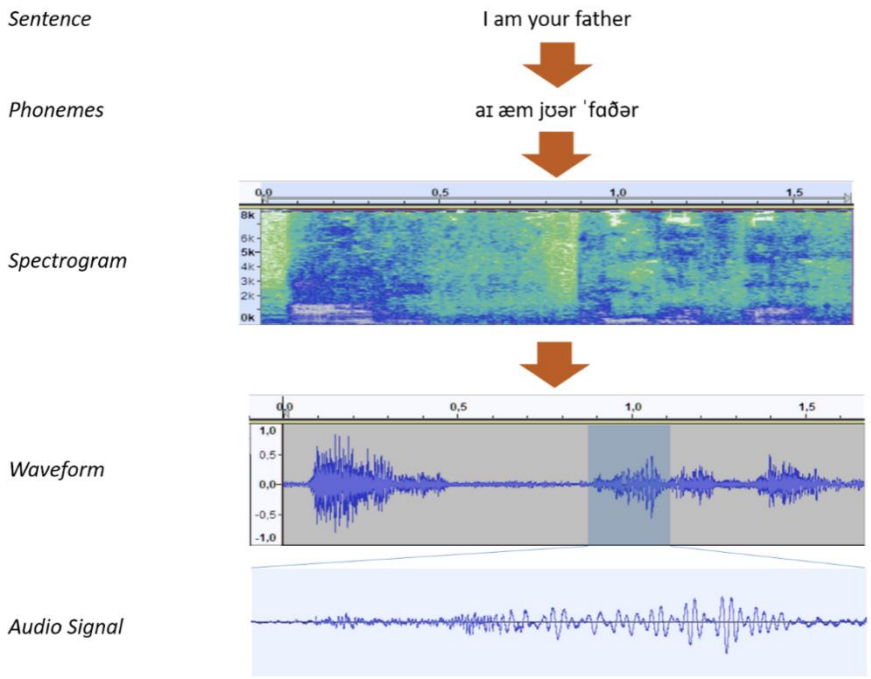


Figure 6. The stages of converting a string into an audio signal extracted from (Meyer, 2021b).

The first component of a modern TTS system uses the same architecture shown for ASR – the encoder-decoder. Hence, one of the most modern architectures used for TTS is the usage of Tacotron2[9] architecture, extended to the earlier Tacotron (Wang et al., 2017) and the Wavenet vocoder (van den Oord et al., 2016). The following figure displays the components of a machine learning architecture used to generate speech:

[9] https://pytorch.org/hub/nvidia_deeplearningexamples_tacotron2/

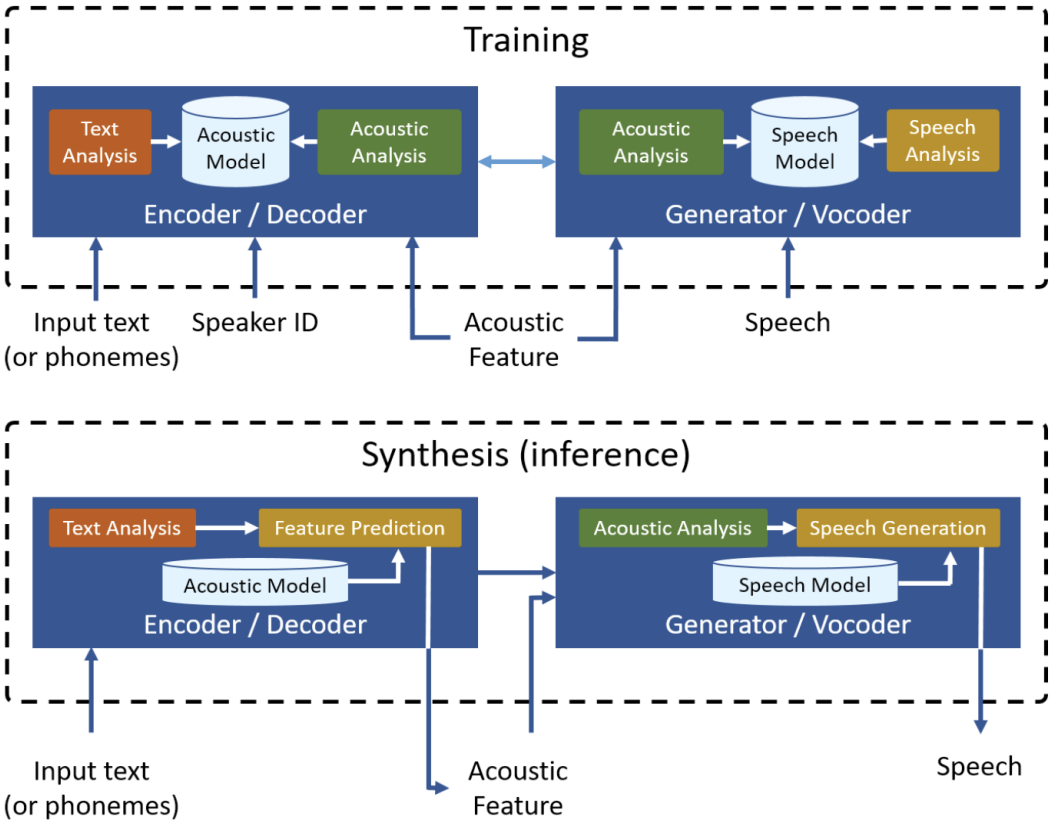


Figure 7. Machine learning architecture of speech synthesis systems extracted from (Meyer, 2021b).

3.2.2.3 TTS challenges

Text-to-speech has improved over the years; however, it still lacks the dynamic variation and adaptability of human speech. Two of the main challenges TTS systems deal with today are prosody and out-of-vocabulary words.

In recent years, end-to-end text-to-speech systems have seen remarkable success, increasing naturalness and intelligibility. Nevertheless, end-to-end models do not supply an explicit way to control and incorporate the necessary prosody – the rhythm, emphasis, melody, duration and vocal stress in speech – while synthesizing the signal. Hence, unnatural prosody is one of the most pressing issues TTS faces today.

In recent studies, TTS systems are showing improvement by achieving more dynamic and natural intonation in synthesized speech. Tyagi et al. (2020) describes a model that uses

syntactic and semantic properties of the utterance to decide prosodic features. Similarly, Latif et al. (2021) provides a control mechanism for E2E systems to manipulate prosodic features.

Given the unlimited context for text-to-speech synthesis and the current multilingual environment, TTS is often affected by the presence of out-of-vocabulary (OOV) words. OOV words are caused by a variety of factors, including the use of technical terminology, proper nouns, unusual words not covered by the vocabulary, and foreign words. This is an even greater problem for non-English TTS systems. Hence, most of the TTS systems are currently made just for English, leaving other languages with less resources.

The issue of homographic heterophones (HH) in languages such as Portuguese or English, is still a question that hasn't been yet very explored in TTS systems. The automatic classification of HH – pairs of words with the same spelling but different pronunciations – has been studied for Brazilian Portuguese (Shulby et al., 2013). The automatic disambiguation of HH word pairs belong to different grammatical classes and are distinguished by openness of a mid-vowel in stressed syllables (*e. g., [o]lho (noun), [O]lho (verb*)).

3.2.3 Conversational agents

Artificial intelligence technologies have become more widely used in recent years, particularly in apps developed for organizations to manage their user requests automatically. A conversational agent, also referred to in state of the art as dialogue systems or dialogue agents, is applied to interact with users using natural language. Indeed, conversational agents are gradually being used in a variety of fields, such as healthcare, education, marketing and customer service. The following section provides an overview of the progress of conversational agents, followed by a description of a frame-based architecture from a simple model to a complex intelligent system.

3.2.3.1 Conversational agents: Overview

A conversational agent is any dialogue system that not only conducts natural language processing but also responds automatically using human language. These agents represent the practical implementation of computational linguistics (Jurafsky & Martin, 2022) The author divides conversational agents in two types: chatbots and (task-based) dialogue agents. Chatbots mimic informal human chatting and are generally used for entertainment, while task-based dialogue systems interface to personal assistants that can perform sophisticated tasks, such as booking a flight, buying a product or answering questions about weather or sports. In the context of this thesis, we will focus on the task-based dialogue agents (or conversational agents).

In the 1950s, Alan Turing created the first chatbot software – ELIZA. It is a computer program that uses natural language processing to simulate a psychotherapist's language. This was the first time a chatbot has been able to have a human-like conversation. A downside of ELIZA is that its knowledge is limited to a range of topics. Thus, it is unable to support long dialogues and to learn context. In 1972, a more advanced chatbot was created – PARRY. Although it showed few advancements in having "emotional responses" it still showed low capabilities concerning language understanding and the ability to express emotions.

3.2.3.2 Dialogue Systems architectures

For a task-based dialogue architecture, we present a frame-based architecture with two different versions: the GUS architecture, and the dialogue-state (or belief-state) architecture. GUS (Genial Understander System) is a simpler architecture. It was first introduced in 1977 but it is still in use in most modern commercial digital assistants. On the other hand, the dialogue-state architecture is a more sophisticated and modern version of a conversational system. Jurafsky & Martin (2022) ensures most commercial systems are architectural hybrids, based on GUS architecture expanded with some dialogue-state components. However, there are a wide range of dialogue-state systems being developed in research labs.

According to Bobrow et al. (1977), GUS was constructed with the intent to engage a sympathetic and highly cooperative human in an English dialog, directed towards a specific

goal within a restricted domain of discourse. It has its limitations making this system a more-or-less realistic dialog. However, the author notes that GUS is indeed interesting because of the natural dialog phenomena, which tries to model rather than offering services or provide information that is difficult to obtain. Thus, the following figure shows a conversation with the original GUS system. The figure illustrates some of the language understanding problems faced by GUS, including understanding indirect answers to questions, resolving anaphora, understanding fragments of sentences offered as answers to questions, and interpreting the discourse, taking into account known conversational patterns.

GUS:	Hello. My name is GUS. I can help you plan a simple trip by air. Where do you want to go ?
Client:	I want to go to San Diego on May 28
GUS:	What time do you want to leave?
Client:	I must be in San Diego before 10 am
GUS:	Would you like P.S.A. flight 102 that arrives at 9.15 am ?
Client:	What is the next flight ?
GUS:	Air California flight 310 that leaves at 8.30am and arrives at 10.00am
Client:	I'll take the first one
GUS:	In what name should I make a reservation, please?
Client:	Dan Bobrow
GUS:	I have confirmed the following flight: P.S.A. flight 102 on Wednesday May 28 from San Jose to San Diego that leaves at 7.30 am and arrives at 9.15 am.
GUS:	What date do you want to return on ?
Client:	On Friday in the evening.
GUS:	Would you like the flight that leaves at 7.45 pm ?
Client:	That's fine.
GUS:	I have confirmed the following flight: P.S.A. flight 307 on Friday May 30 from San Diego to San Jose that leaves at 7.45 pm and arrives at 9.30 pm Thank you for calling. Goodbye

Figure 8. A transcription of a dialogue using the GUS system of Bobrow et al. (1977). *Image extrated from (Jurafsky & Martin, 2022:24,14).*

Frames are the basis of all current task-based dialogue systems, whether the simple GUS architecture or more sophisticated dialogue-state architectures. A frame is a type of knowledge structure that contains a collection of *slots*, each of which can take a set of possible values and represents the *types* of intentions the system can extract from human sentences. This collection of frames is commonly referred to as a domain ontology. A task-based dialogue frame' s collection of slots describes what the system needs to know, and the filler of each slot is limited to values of a specific semantic kind. If we look at Figure 8,

a dialogue in travel domain, a slot could be of type *city*, *date*, *airline*, *time,* or values such as *San Diego*, *May 28*, *Air California*, *9.15 am*, respectively.

A more advanced version of the frame-based architecture is the dialogue-state (or belief-state) architecture, which is used in modern research systems for task-based dialogue. The architecture of a typical dialogue-state system is demonstrated in Figure 10.

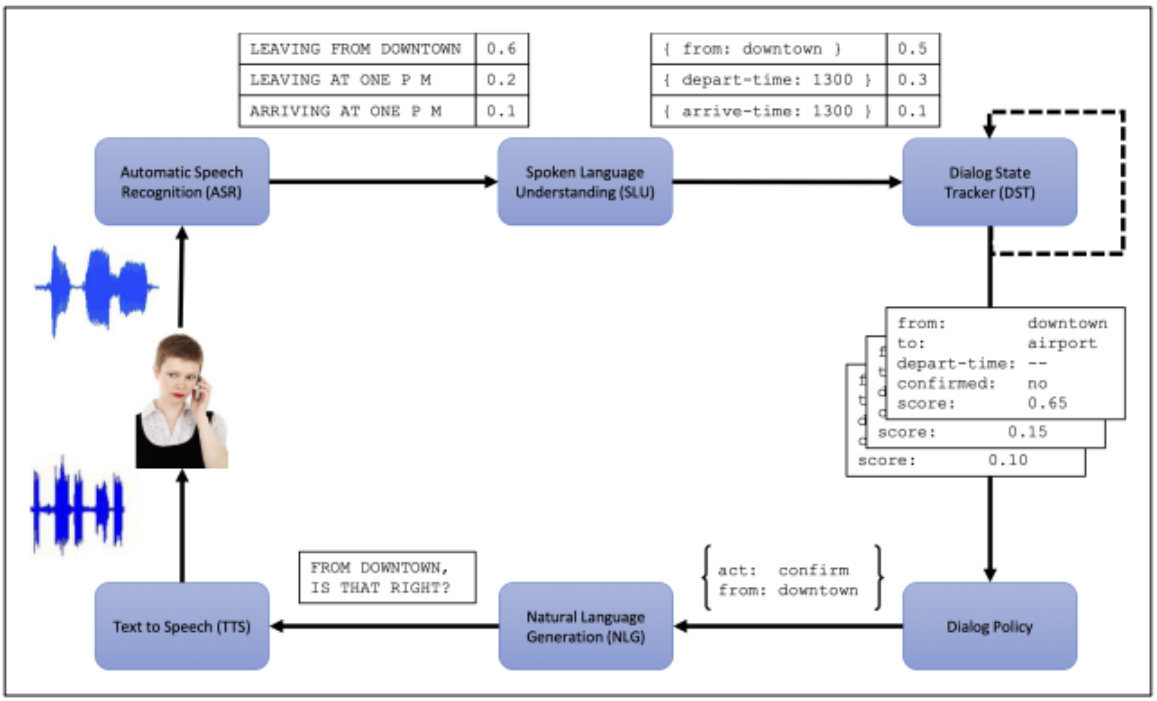


Figure 9. Dialogue-state system architecture extracted from Williams et al. (2016:5)

After describing and giving an overview of ASR and TTS, in Section 3.2.1 and 3.2.2, respectively, conversation agents combine both architectures. ASR and TTS systems make the first and final component of a dialogue-state architecture, which are combined with other major elements:

1. **Automatic Speech Recognition (ASR)** - converts audio that was previously produced by a user into words in text form. ASR and Spoken Language Understanding (SLU), make one single system. SLU, or also known as Natural

Language Understanding (NLU) maps a given utterance into a meaning representation (semantic slots).

2. **Dialog State Tracker (DST) -** updates its estimate of the dialogue state.
3. **Dialog Policy** - learns what the system should do next given the current dialogue state.
4. **Natural Language Generation (NLG) -** converts an abstract dialogue action into natural language surface utterances/responses.
5. **Text to Speech (TTS) -** converts text to an acoustic wave.

3.3 Linguistic Processing

With 7,151 languages[10] in the world and the increasing need to support multiple input and output languages, building and deploying speech processing systems is becoming more challenging. One of the most urgent concerns facing the speech and language field is unsupported languages.

Defined.ai uses a Linguistic Processing Module (LPM) to treat raw text as input and structure it to make it usable for speech technologies that require linguistic knowledge. The LPM is used in all technologies previously mentioned. Therefore, LPM includes the development of qualified pronunciation lexicons that will provide a mapping between a word's orthographic form and pronunciation. Pronunciation lexicons are mainly used to build G2P models, for the purpose of providing pronunciation of OOV words.

3.3.1 Multilingual G2P

A Grapheme-to-phoneme (G2P) model maps a written text into a string of symbols which represent the speech sounds exactly and unambiguously. This model is an essential component in the general architecture of automatic speech recognition (ASR) and text-to-

[10] https://www.ethnologue.com/guides/how-many-languages

speech (TTS) systems. Thus, a G2P model is part of the preprocessing module of the TTS system explained in Section 3.2.2.2.

The G2P conversion has been addressed by a number of frameworks, including statistical techniques based on conditional probability, such as joint n-gram models (Sar & Tan, 2019) and rule-based models that map each grapheme into its corresponding phoneme(s) (Kłosowski, 2022). A rule-based approach requires hand-crafted rules written by linguists with knowledge of the target language. Therefore, a rule-based system should provide good coverage of the correspondence between letters and sounds, especially in languages where orthography is generally phonologically based, like Portuguese, Spanish or French. However, it is unlikely that any actual human language perfectly conforms to this assumption. The most frequent irregularity is when the relationship between a grapheme and a phoneme is more than a one-to-one correspondence and it might be highly dependent on the context of the following words. Typically, in an ASR system, the G2P generates the pronunciation for out-of-vocabulary (OOV) words based on their written forms. In G2P conversion, sequences of graphemes are mapped to corresponding phones (usually, symbols of the International Phonetic Alphabet). Building a pronunciation lexicon for a new language is both time-consuming and expensive, because it requires ability in both the language and a notation system, such as IPA, applied to thousands of word-pronunciation pairs. Not so long ago, resources have only been distributed to the most heavily researched languages. GlobalPhone is one of the most extensive multilingual text and speech databases, with pronunciation dictionaries that cover 20 languages. Despite covering few languages, this multilingual database supplies a basis for research in the areas of multilingual speech recognition, rapid deployment of speech processing systems to yet unsupported languages, as well as monolingual speech recognition in a large variety of languages (Schultz & Schlippe, 2014). Indeed, research in unsupported languages is one of the main challenges the G2P faces presently. In accordance with Deri & Knight (2016), there is a clear absence of data for most of the worlds' languages, as well as many unavailable technologies permitted by G2P models. Likewise, because of the lack of multilingual pronunciation lexicons and G2P models, different methods for faster resource generation have been proposed.

Traditionally, grapheme-to-phoneme systems were monolingual and often restricted to English due to data availability. These early systems were created to use rule-based transition systems to solve the issue of intra-language inconsistencies. As described by Yu et al. (2020) multilingual G2P for languages with different writing systems, such as European and Asian languages, remains an understudied area.

An alphabetic writing system is the writing system for a language in which a symbol is given for each consonant and vowel. The phonetic representation is a conventional representation of a words' pronunciation. A one-to-one correspondence between graphemes (letters) and phones (sounds) is, to the best of our knowledge, possible only for Swedish.

3.3.1.1 G2P approaches

In earlier studies, G2P conversion for many languages used various approaches. Typically, a G2P model is created using a lexicon-based approach, a rule-based approach (knowledge-based approach) or a data-driven approach, such as statistical methods including decision trees, neural networks and Hidden Markov Models. A lexicon-based approach uses a large pronunciation lexicon, while a rule-based approach maps grapheme-to-phoneme using manual rules based on expert knowledge. In contrast, a data-driven approach learns from data.

A lexicon-based approach is commonly applied to languages whose orthography is roughly phonetically based, such as English or French (Braga, 2006). However, this technique fails when OOV words come up. While simple and effective, it has limitations, including making a pronunciation lexicon of significant size (with over 100,000 entries) by hand. More importantly, a limited lexicon will always have restricted coverage, while TTS systems are often expected to handle arbitrary words. To overcome these limitations, rule-based conversion systems were developed. A rule-based approach uses rules as the knowledge representation. It captures the knowledge of a linguist and embodies it within a computer system. Lexicons can often be used as an exception list in rule-based G2P systems. These systems provide complete coverage for languages with regular grapheme-to-phoneme mapping (Kłosowski, 2022) However, a rule-based approach is not suitable for languages

with complex grapheme-to-phoneme mapping, for example, English, where their pronunciation system is irregular. This can be explained through many heterophone (or heteronyms) examples in English. Heterophones are words that have the same spelling but a different meaning (*e. g.*, record, direct, present, accord, etc). For example, the word "present" – "I want to give you this present.", or "At the present moment I am very happy." – should not be pronounced the same way in each of the sentences. To know how to pronounce a word correctly, we first must understand its context.

In opposition to the rule-based approach, the data-driven approach to G2P requires data to learn the rules. It is based on the idea that with enough examples it is possible to predict the pronunciation of new words by analogy. ML sequence-to-sequence systems learn from examples of word and pronunciation pairs to build a learning system (Sar & Tan, 2019). Although the learning process of these systems are deterministic (including statistical and probabilistic methods), they can't be interpreted, explained, and understood well enough by humans.

3.3.2 Text normalization

Text normalization is a stage of pre-processing for a variety of speech and language processing applications. According to Zhang et al. (2019) text normalization refers to the process of verbalizing semiotic class [1] instances, for example, converting *28/08/2022* into its verbalized form *August twenty-eighth, twenty twenty-two*. Thus, text normalization is part of TTS and ASR systems, as explained, respectively, in Section 3.2.1.2 and Section 3.2.2.2. Therefore, TTS and ASR systems require comprehensive language-specific text normalization processing pipelines to handle token such as abbreviations (*abbrev*.), currency (*$300*), among many others.

When normalizing text for TTS systems, the written numeric value must be mapped to its spoken form (*e. g.*, *one thousand nine hundred ninety-nine* or *nineteen ninety-nine*). Writing long numbers out as they are spoken is not something speakers tend to do, so modifying to an accurate form is necessary. Also, when reading a written address (*e. g.*, *134 Pine Av*.), if

the numeric value is not preserved in the normalization (*thirteen hundred four Pine Avenue*), then a driver following spoken directions in a map application may be misled.

The main challenge in text normalization is the variety of semiotic classes. Sproat et al. (2001) provided the first systematic overview of the so-called Non-Standard Words (NSWs).

	EXPN	abbreviation	*adv, N.Y, mph, gov't*
alpha	LSEQ	letter sequence	*CIA, D.C, CDs*
	ASWD	read as word	*CAT*, proper names
	MSPL	misspelling	*geogaphy*
	NUM	number (cardinal)	*12, 45, 1/2, 0·6*
	NORD	number (ordinal)	*May 7, 3rd, Bill Gates III*
	NTEL	telephone (or part of)	*212 555-4523*
	NDIG	number as digits	*Room 101*
N	NIDE	identifier	*747, 386, I5, pc110, 3A*
U	NADDR	number as street address	*5000 Pennsylvania, 4523 Forbes*
M	NZIP	zip code or PO Box	*91020*
B	NTIME	a (compound) time	*3·20, 11:45*
E	NDATE	a (compound) date	*2/2/99, 14/03/87* (or US) *03/14/87*
R	NYER	year(s)	*1998, 80s, 1900s, 2003*
S	MONEY	money (US or other)	*$3·45, HK$300, Y20,000, $200K*
	BMONEY	money tr/m/billions	*$3·45 billion*
	PRCT	percentage	*75%, 3·4%*
	SPLT	mixed or "split"	*WS99, x220, 2-car* (see also SLNT and PUNC examples)
	SLNT	not spoken,	word boundary or emphasis character:
M		word boundary	*M.bath, KENT*RLTY, _really_*
I	PUNC	not spoken,	non-standard punctuation: "***" in
S		phrase boundary	*$99,9K***Whites*, "..." in *DECIDE...Year*
C	FNSP	funny spelling	*sllooooowww, sh*t*
	URL	url, pathname or email	*http://apj.co.uk, /usr/local, phj@tpt.com*
	NONE	should be ignored	ascii art, formatting junk

Figure 10. Overview of the Non-Standard Words extracted from Sproat et al., (2001):13

Several other types of NSWs have been encountered over the years when developing a general-purpose speech system. Hence, these systems aim to be robust to various domains and registers. A recent update of this taxonomy is presented by van Esch & Sproat (2017). The authors expanded semiotic classes to twelve more original categories (*e. g.*, measures, mathematical formulae, telephone numbers). This taxonomy is considered to cover most languages.

3.3.2.1 Text normalization approaches

A text normalization system can cover a wide range of approaches. The simplest method for text normalization is known in the state of the art as lexical substitution, lexicon lookup, wordlist mapping, or memorization (Bollmann, 2019). It looks up each variant in a list that maps it to its desired normalization. On the other hand, it can also use a rule-based approach, a distance-based approach, a spelling correction method, as well as neural models. One of the earliest approaches to text normalization is rule-based. Typically, rules are created manually for one specific language. This method attempts to encode regularities in spelling variants as replacement rules, which includes context information to distinguish between different usages of a character. A distance-based approach uses distance measures to find the nearest correct possible word from a lexicon to produce the output. The spelling correction method uses HMM to analyze word morphology and determine the correct spelling. On the other hand, neural models use neural networks such as encoder-decoder model with LSTM.

With the complexity of text normalization, as well as the fact that deep learning in ASR and TTS systems is innovative, there are still areas that require a careful application of linguistic knowledge. The text normalization development can be done specifically for each language and/or task, but this work is complex and time-consuming. However, there is still a lack of effort on text normalization for many less-resourced languages.

3.4 Summary

This chapter addressed relevant work on linguistic topics such as phonetic and phonology concepts regarding two different languages – Swedish and Russian. We also covered speech recognition systems, such as ASR, TTS, and Conversational agents. Finally, we discussed Defined.ai Linguistic processing module focusing on Multilingual G2P models and text normalization.

Chapter 4
Methodology

This chapter addresses our third goal: *How to upgrade and validate the Normalizer and G2P models in different languages*? In order to improve version 2 of the normalizer, we first analyze how version 1 of the normalizer performed in comparison to how we want version 2 to work.

Second, we outline how to build two new G2P models for different languages. Briefly, the following two transitional questions served as the basis for our research:

1. How to upgrade a Normalizer into one that covers most of the normalizable tokens?
2. How to create and validate new G2P models in two different languages?

For this purpose, this chapter is structured as follows: Section 4.1 describes how the Normalizer rules work, the process of expanding the rules to a better version of the normalizer (version 2 of the normalizer), and what implementations we made to upgrade it; Section 4.2 describes how we prepared a new G2P model for Swedish (see Section 4.2.1) and for Russian (see Section 4.2.2). Finally, Section 4.2.3, explains how we revised and validated the phonetic lexicons created to train the G2P models.

4.1 Normalizer Expansion

This section deals with the process of expanding and upgrading version 1 of the normalizer. Briefly, these will further answer the following two questions:

1. How well does the normalizer do its job of converting symbolic or reduced language to a spoken form?
2. What can improvements within the normalizer's scope be made to improve performance?

The Normalizer Linguistic Expansion (NLE) project aimed to provide coverage for several rules: Real numbers, Symbols, Abbreviations, Ordinals, Measurements, Currency, Dates,

and Time. The task consisted of expanding the rules Replacement Maps (RMs) with numbers, symbols, abbreviations, ordinals, measurements, currency, dates, and time, respectively, from following rules that would have an unambiguous unmarked reading of their own. The main goal was to modify the Normalizer to be simpler, more consistent between different languages, and have more coverage of unambiguous inputs. Ideally, these expanded forms match a speaker's words when reading the text. We generally take the most conservative approach, where in possible error cases, a graceful failure is the likely outcome.

Version 1 of the normalizer – the first version before the NLE project – converted currency symbols or measurement abbreviations to a spoken form only when they were adjacent to a real number. Table 9 illustrates the normalization process.

Input	**Normalized Output**
não mais **cm2** de território inacessível para a polícia eles somaram um buraco de **1577 cm2** na parede	não mais **cm2** de território inacessível para a polícia eles somaram um buraco de **mil e quinhentos e quarenta e sete centímetros quadrados** na parede

Table 9. Normalization process - input and output example

The NLE implied the decision on which symbols or abbreviations are unambiguous enough to be expanded deterministically without a real number as context. New symbols and abbreviations were added to the RMs for Portuguese and testing samples to the respective Unit Tests (UTs)[11]. The goal of this expansion is to have a deterministic reading of spoken forms (Table 10).

[11] A Unit Test (UT) is a set of one or more examples to ensure that a unit of code behaves as expected. More specifically, Unit Tests are sets of examples of inputs and expected outputs that test whether the normalizer produces the correct - or expected - output given the input.

Input	Normalized Output	Expected Output
não mais **cm2** de território inacessível para a polícia	não mais **cm2** de território inacessível para a polícia	não mais **centímetros quadrados** de território inacessível para a polícia
o valor da **₹** aumentou desde o ano passado	o valor da **₹** aumentou desde o ano passado	o valor da **rupia** aumentou desde o ano passado

Table 10. Normalization process - currency and symbols input and expected output

The Symbols rule converts a prescribed set of non-alphanumeric symbols into their spoken forms. This rule is, however, a backup rule. In most instances the symbols will be handled by other rules before they are even read by the Symbols rule (*e. g.*, the input *24$* would be handled by a separate, previous currency rule.). Though, we still need this rule to prevent miscellaneous instances where symbols do not appear in their regular context, and to avoid unambiguous symbols that appear in isolation and do not require a more restrictive rule. We were interested in adding only symbols that correspond to an actual unit of speech in an equivalent spoken sentence (Table 11).

Input	Normalized output
76 **-** 10	setenta e seis **menos** dez
Tenho 13 **%** de bateria	Tenho treze **por cento** de bateria
O número de latas **÷** caixas	o número de latas **dividido por** caixas
A temperatura está abaixo de 1 **℃**	a temperatura está abaixo de um **grau Celcius**
O **π** não é um número racional	o **pi** não é um número racional
O valor da hipotenusa é **=** 20	o valor da hipotenusa **é igual a** vinte

Table 11. Normalization process – symbols input and output

Version 1 could only access the default spoken form for any given symbol (*e. g.*, symbol can only be normalized to the singular *degree*, rather than either *degree* or *degrees*).

The Abbreviations rule is a backup rule for converting miscellaneous alphabetic or mixed alphabetic-symbolic sequences to their spoken form (Table 12). When one of the abbreviation lexicon terms is found in the input text, it will be replaced by its expanded form. The following requirements were needed for this rule:

- could consist of both abbreviations[12], acronyms[13], initialism[14] or blended acronyms[15]
- an abbreviation consists of only one word
- an abbreviation can end with a period or not.
    - in cases where the text contains a period at the end of a sentence, it is kept when the abbreviation is expanded. (*e. g.*, the phrase "A minha casa fica longe do teu ap." would be expanded to "A minha casa fica longe do teu apartamento." with the dot after the "apartamento" kept.
    - When the period after the abbreviation was not kept in the expanded text, the period was included in the abbreviation definition (*e. g., "*Av." would be expanded to "Avenida" and "Av" would be considered a separate entry.

Input	**Normalized output**
Wi-Fi	uaifai
r/c	rés do chão
O **ap.** fica longe	o **apartamento** fica longe
O **D.r** Duarte trabaha no Hospital Santa Maria	o **doutor** Duarte trabaha no Hospital Santa Maria
Ela é vegetariana, **i. e.**, ela não come carne e peixe	ela é vegetariana, **isto é**, ela não come carne e peixe
O **núm.** de erros é alto	o **número** de erros é alto

Table 12. Normalization process - abbreviations input and output.

[12] Any shortened form of a written word or phrase used in place of the whole word or phrase that does not include acronyms, initialisms, and blended acronyms (see footnotes 15, 16, 17).

[13] A kind of abbreviation in which the initial letters of other words are pronounced as a word, for example, *NASA* stands for the *National Aeronautics and Space Administration* but is pronounced as the word *nasa* following the standard rules of English phonology.

[14] A kind of abbreviation composed of the initial letters of other words but pronounced by their individual letters. For example, *CIA* stands for the *Central Intelligence Agency*, and it is pronounced as three individual letters, *C I A.*

[15] A kind of abbreviation which involves both pronouncing letters as words and spelling individual letters. For example, the standardized entrance exam for medical school is called *MCAT*, and it is pronounced as a spelled letters followed by a word, *M cat*.

Version 1 did not consider context for this rule, it could only access a single expanded form for any given entry in a lexicon. In version 2 of the normalizer, all abbreviated forms were added in order to have multiple possible expansions due to lexical or inflectional reasons.

The Real Numbers rule converts numeric values of integers and/or decimal values into their spoken forms (Table 13).

Input	**Normalized output**
5	cinco
4,44	quatro vírgula quatro quatro
0,6541	zero vírgula seis cinco quatro um
85 631,11	oitenta e cinco mil seiscentos e trinta e um vírgula um um
3612367	três milhões seiscentos e doze mil trezentos e sessenta e sete
O total de seres vivos é de **4,983,201**	o total de seres vivos é de quatro milhões novecentos e oitenta e três mil duzentos e um

Table 13. Normalization process - real numbers input and output.

Version 1 provided coverage for 6 patterns, composed of up to 7 tokens. Version 2 takes now grammatical features (*e. g.*, gender) into account for normalization. Therefore, real numbers will no longer be normalized using only a default spoken form, instead various real numbers' forms. We also implemented whole numbers starting with 0 (*e. g.*, *02*, or *002*), constituting now an acceptable pattern.

The Ordinals rule converts a sequence of a real number and an ordinal marker into their expanded spoken forms (Table 14). This rule is activated when it encounters a construction made up of two consecutive components: 1. A real number; and 2. An ordinal marker. Thus, the ordinal was normalized into its appropriate number, gender, and case spoken forms, as necessary.

Input	Normalized output
5.°	quinto
4 907.°	quarto milésimo nongentésimo sétimo
1.as	primeiras
567 890 123.º	quingentésimo sexagésimo sétimo milionésimo octigentésimo nonagésimo milésimo centésimo vigésimo terceiro
Esta é a **1 000.ª** parte	esta é a **milésima** parte
As **1.as** temporadas são as melhores	as **primeiras** temporadas são as melhores

Table 14. Normalization process - ordinals input and output.

The measurement rule converts a sequence of a real number and a measurement unit into their expanded spoken forms (Table 15). This rule is activated when it encounters a construction made up of two components, optionally separated by whitespace: 1. A real number; and 2. A measurement unit symbol. The measurement unit was normalized into its singular or plural spoken form, based on the preceding number. Also, the rule provided coverage for gender agreement between the number and the measurement unit.

Input	Normalized output
a régua tem 100 **cm**	A régua tem cem **centímetros**
23,2 **km2**	vinte e três vírgula dois **quilómetros quadrados**
2,000 **Pa**	dois mil **pascais**
O disco tem **1GB**	o disco tem um **gigabyte**
0,4 **MHz**	zero ponto quatro **megahertz**
2,8 µSv	dois ponto oito **microsieverts**

Table 15. Normalization process - measurement input and output.

The currency rule covers currency expressions in their expanded forms (Table 16). this rule is activated when it encounters a construction made up of two components, optionally separated by whitespace: 1. A real number; and 2. A currency symbol code. The real number component and the currency component could be in either order. Thus, the currency symbol was normalized into its singular and plural form, also, the rule provided coverage for gender agreement.

Input	Normalized output
178 **¥**	cento e setenta e oito **ienes**
$23,2	vinte e três **dólares** e dois **cêntimos**
2,000 **NZD**	dois mil dólares neozelandeses
CNY 8 milhões	oito milhões de iuanes chineses
0,99 **R$**	noventa e nove **centavos de real**
1 **¢**	um centavo

Table 16. Normalization process - currency input and output

In version 2 of the normalizer, the measurements and currency rules had to introduce several new abbreviations (*e. g., cm, km2, mSv, Pa*) and symbols (*e. g., rupee-sign, won-sign, ruble-sign*). Still, under the current rules, these would only be expanded to their spoken form if adjacent to an actual number.

The dates rule converts data expressions into expanded spoken forms (Table 17). Date expressions were composed in three predominant orders (1. Year-Month-Day, 2. Day-Month-Year, and 3. Year-Month-Day). In addition, we accepted reduced date expressions (*e. g.*, Day-Month and Month-Year.)

Input	**Normalized output**
No dia **13/9/2018**, comecei a trabalhar.	no dia treze de setembro de dois mil e dezoito, comecei a trabalhar.
No dia **21 de out. de 2004** comprei o meu carro.	no dia vinte e um de outubro de dois mil e quatro comprei o meu carro.
1994.11.09	nove de novembro de mil novecentos e noventa e quatro
28 Fev 2017	vinte e oito de fevereiro de dois mil e dezassete
15.03.2012	quinze de março de dois mil e doze
1º-4-2007	primeiro de abril de dois mil e sete

Table 17. Normalization process - dates input and output.

The two main types of date expressions that we covered with the dates rule were: 1. Digital format with three different date separators (*e. g., 12/12/2012, 12.12.2012,* or *12-12-2012*); and 2. Calendar date format is composed of a numbered day, a word-from month and a digital year (*e. g., 28 de novembro de 2021*). Days use one or two cardinal numbers, months are represented as full or abbreviated word form (*e. g., novembro,* or *nov.*), and years are

made up of four digits or by the last two digits from the complete year (*e. g., 2/02/1999, 2/2/99; 1999-02-02, 99-2-2*).

The time rule converts a numeric time expression into a spoken form (We cover two main types with this rule: 1. Digital clock time; and 2. Hour-minute-second time. The digit components are separated by a time separator or followed by a letter corresponding to a time unit (*e. g., 14:10, 14h10*).

Input	**Normalized output**
São **10h22**, horário local	são dez horas e vinte e dois minutos, horário local
Trabalha até as **16h**	trabalha até ás **dezasseis horas**
O terremoto começou às **10h35min22s**	o terramoto começou ás dez horas, trinta e cinco minutos e vinte e dois segundos
12:00	meio dia
O dia começa à **00h**	o dia começa à **meia noite**
São **9:00h** agora	são **nove horas** agora

Table 18. Normalization process - time input and output.

Throughout the NLE project, we expanded several normalizable tokens (symbols, abbreviations, etc.), implemented rules on the various rules, and fixed issues related to varieties confusion between pt-PT and pt-BR. We ensured that pt-PT rules and pt-BR rules were always separate in version 2. The main problem was that several PN rules and assets were shared between different regions of the same language. This had been done in pt-PT and pt-BR for several rules, but mainly due to some orthographic differences between pt-PT and pt-BR.

Indeed, we made the necessary changes to have a more refined and robust pt-PT model. A few of the adjustments we made are shown in Table 19. In the first column are pt-BR orthographic forms used in the pt-PT PN, and in the second column are the new orthographic forms in pt-PT.

Version 1 of the normalizer	Version 2 of the normalizer
kilómetros	quilómetros
bilhões	mil milhões
dezesete	dezassete
dezenove	dezanove
centavos	cêntimos

Table 19. Improvements on the version 2 of the normalizer

4.2 G2P models

To prepare a G2P model for two different languages - Swedish and Russian – the following four tasks were performed:

1. Prepare a language overview
2. Develop initial phone set
3. Map and automatically convert the initial phonetic lexicon to DC-Arpabet
4. Revision and correction of phonetic lexicon
5. Evaluate G2P model

Firstly, it is necessary to understand both languages concerning orthography, phonology, and grapheme-to-phoneme relationships. The language overview step was made regarding a review of the typology of the language; a study of the phonemic graphemes and phones; a fundamental analysis of grapheme-to-phoneme correspondences in the language and its tokenization scheme (*e. g.*, space-separated words); an overview of available phonological/phonetic corpora and lexica, and suggestions on how to proceed with data preparation. Therefore, by comprehending and considering the phonology and orthography of the specific language, we could begin developing a plan for training a G2P model.

4.2.1 Swedish G2P

Following the language overview, a phone set was created, which was used as the basis for a training lexicon. This phone set was developed based on the insights from the language overview step, which also involved a discussion with native speakers.

Defined.ai assigned two native Linguists to revise and discuss our work in the Swedish language. At this stage, the following topics were discussed regarding the Swedish phone set – 1. Choice of the most accurate/suitable DC-Arpabet phones for the equivalent Swedish graphemes; 2. The distinction between long and short vowels; 3. Application of a retroflex rule.

Firstly, choosing a suitable and accurate phone for a grapheme was our priority when developing a phone set. The following table shows Swedish graphemes <e, i, o> and their DC-Arpabet and IPA equivalents with two different phones [ex, ax, ih, ii, oo, ob], respectively. During the phone set development process, we had a few doubts about choosing the language's most representative phone. The Linguist helped us choose the most canonical way of pronunciation for the Standard Swedish graphemes

Grapheme	**Phonetic Alphabet Symbols**		**Example**
	IPA	**DC-Arpabet**	
<e>	[ə]	[ex]	leda**r**en
<e>	[ə]	[ax]	leda**r**en
<i>	[ɪ]	[ih]	enl**i**gt
<i>	[i]	[ii]	enl**i**gt
<o>	[o]	[oo]	inkomsttax**o**r
<o>	[ɒ]	[ob]	inkomsttax**o**r

Table 20. Standard Swedish vowels pronunciation.

Secondly, we initially observed that the Swedish language distinguishes 9 long vowels [aː, ɛː, æː, eː, oː, øː, œː, uː, yː] from the corresponding 9 short vowels [a, ɛ, æ, e, o, ø, œː, u, y]. In accordance with the native Linguist, vowel length is not essential to get a transcription that reflects a canonical and accurate pronunciation. So, we decided to merge these vowels and only keep the short version of the vowel. Consequently, we reduced and got a more straightforward phone set, which meets our initial necessities on an excellent quality G2P model. On the other hand, the Swedish vowel inventory has vowels that are consistently long [ɑː, iː] and vowels that are constantly short [ə, ɪ, ɔ, ɵ, ʉ]. For the always long vowels, we also adopted the equivalent short DC-Arpabet phones [ah, ii]. The short vowels have the respective DC-Arpabet phones [ex, ih, oh, uo, uc].

Thirdly, the Swedish consonant inventory includes 5 retroflex consonants [ʈ , ɖ, ʂ ɭ, ɳ]. To make a simpler phone set, we decided not to add these retroflex consonants but rather to apply a rule – r + base consonant. Retroflex consonants merge when an <r> is followed by <t, d, s, l, n>, for example in *lä**rt***, *saabo**rd**er*, *lände**rs***, *medborge**rl**iga*, *läka**rn**a*. When this pattern is found, the retroflex rule is applied.

We used a GlobalPhone (GP) lexicon for both languages as an initial phonetic lexicon. After developing the phone set, we needed to convert the GP lexicon to a form usable for G2P training. At Defined.ai, the phonetic lexicons used to train the G2P model use the DC-Arpabet and follow a simple and straightforward format. Accordingly, the final phonetic lexicon for Swedish had to respect the following requirements:

- only words (no words with numbers and/or symbols)
- all words in lowercase
- all words and their respective transcriptions separated by a vertical bar character
- all words should follow a phonetic transcription using the DC-Arpabet
- all words written using the Swedish script (diacritics, letters, additional characters)

Therefore, we, firstly, created a phone set mapping from the original phone set used in the GP lexicon to the DC-Arpabet phone set. A reusable script was then converted from the original phonetic lexicon to the G2P-ready lexicon. In that way, we would be able to convert all transcriptions automatically. The following figure shows the GlobalPhone phonetic lexicon.

{AOberg} {{M_ol WB} M_b M_ale M_r {M_j WB}}

{AObrink} {{M_ol WB} M_b M_r M_il M_ng {M_k WB}}

{AOhle+n} {{M_ol WB} M_h M_l M_el {M_n WB}}

{AOrja^ng} {{M_ol WB} M_r M_j M_ae {M_ng WB}}

Figure 11. Examples of entries in the Swedish GlobalPhone phonetic lexicon.

The Initial GlobalPhone lexicon used a highly complex format, in which phonetic transcriptions were hard to read, following a complicated original phone set. The lexicon

included several loanwords, words with numbers, symbols, acronyms, and abbreviations. Swedish words did not use the correct Swedish diacritics characters, as seen in Table 21. We created a script to automatically change the diacritics used in GP to ones used in Swedish

GlobalPhone words	**GP | Swedish diacritics**		**Swedish words**
	Before	**After**	
A0berg	Aa + 0	å	**Å**berg
fl**o**^g	Oo + ^	ö	fl**ö**g
Allm**a**^nna	Aa + ^	ä	Allm**ä**nna
arm**e**+er	Ee + +	é	arm**é**er

Table 21. GlobalPhone Swedish diacritics and Swedish diacritics.

To have a more concise lexicon, the script respected the following rules:

- Skip entries with: Abbreviations, Initialisms (that cannot be read as a word/acronym), Symbols (or words that contain symbols), words that have numbers, Non-words, Loanwords that would strain the usual orthography-pronunciation system of the language too much, Nouns that are not common in Swedish culture.
- Remove curly brackets around the word
- Convert diacritic characters to the correct form
- Extract all the GP phones from within the transcription (*e. g.* M_abl)
- Convert all the GP phones to DC-Arpabet using the mapping lexicon

4.2.2 Russian G2P

Developing a Russian G2P model shares the same steps as the Swedish model. Therefore, a phone set was created after doing the language overview, which would also be used as the basis for the training lexicon. The phone set was developed and discussed with a native speaker regarding the following topics – 1. Phonetic representations of ш and щ; 2. Soft sign, hard sign, and glottal stop values; and 3. Palatalized consonants.

As mentioned in Section 3.1.2, the Russian consonant graphemes ш and щ share similar phonetic values. Also, both ш and щ have two different ways of pronunciation depending on the speaker or the region within Russia. In agreement with a native Linguist, we decided

to give the simplest and most representative of the Standard Russian pronunciation phonetic symbols. The grapheme щ is represented by the IPA symbol [t͡ɕ] (a plain voiceless alveolo-palatal affricate), and the grapheme ш is represented by the IPA symbol [ʂ] (a voiceless retroflex fricative).

As also mentioned in Section 3.1.2, the soft sign and hard sign have no phonetic value. Thus, when present between a consonant and a vowel, it indicates that the following consonant is hard (*Ъ*), and when it occurs between a consonant and a vowel, between two consonants, or at the end of a word after a consonant, it indicates that the following consonant is soft (*Ь).* In that matter, we will not give a phonetic symbol to graphemes *Ъ* and *Ь*. We added *Ь* to hard consonants (*e. g.*, <ть, сь, рь>) to distinguish hard consonants from soft consonants.

Conversely, the soft sign often uses the Russian glottal stop (/ʔ/). We concluded that /ʔ/ is redundant. Therefore, we do not want to include it in our phone set.

Palatalization is contrastive in word-final and heterogenic medial coda positions. Likewise, we contrast non-palatalized (hard) consonants with palatalized (soft) consonants. We decided on merging soft consonants (/fʲ, gʲ, kʲ, xʲ/) with their hard version (/f, g, k, x/) because these consonants rarely occur on the GlobalPhone database we used to create the phonetic lexicon.

{Андреевич} {{M_a WB} M_n M_d M_r M_jE M_jE M_v M_i {M_tS WB}}

{Анкаре} {{M_a WB} M_n M_k M_a M_r {M_jE WB}}

{Аннабы} {{M_a WB} M_n M_n M_a M_b {M_i2 WB}}

{Анпилов} {{M_a WB} M_n M_p M_i M_l M_o {M_v WB}}

Figure 12. Examples of entries in the Russian GlobalPhone phonetic lexicon.

The next few steps were followed: 1. Map and automatically convert the initial phonetic lexicon (Figure 12) to DC-Arpabet, 2. Human lexicon revision and correction 3. Evaluate the G2P model). Steps 2 and 3 will be described in Section 4.2.3 and Section 4.3, respectively.

4.2.3 Phonetic lexica revision

Neevo is a crowdsourcing language service platform, as mentioned and described in Section 2.3 that provides users with jobs. Jobs provide answers and judgments on several hits: language tasks such as checking and correcting transcriptions. Thus, to assess and improve the results of the G2P, the lexicon used to train the model for each language was revised by native and highly proficient linguists. This revision took several steps, developed within a Text Variant Correction job on the *Neevo* platform.

Firstly, in consultation with our team at Defined.ai, Linguists checked that the DC-Arpabet phone set for their language was suitable for transcribing words and making phonemic distinctions. Linguists corrected and revised the phone set as necessary.

After the phone set revision, the lexicon was ready to be revised. Linguists worked throughout three steps:

1. The revision of each entry in the G2P lexicon does not match the transcription given by the G2P classifier.
2. Establishment, for each entry, the type of words there are – loanwords, words with a canonical pronunciation, and words with a non-canonical pronunciation. Loanwords needed to be removed.
3. If the phonetic transcription was incorrect, provide a correct transcription following the DC-Arpabet phone set for the language.

In conclusion, the Swedish and Russian Linguists accessed each entry in the lexicon through the platform and a) validated the spelling of the word and corrected it if invalid, b) validated the transcription of the word and corrected it as invalid. Below are listed the steps enclosed in the Text Variant Correction job[16]:

1. Read and validate the word:

[16] Images of each of the referred steps are made available in the Appendix section – Figure 15, Figure 16 and Figure 17.

1.1. If the word is correctly spelled, choose *thick.*

1.2. If the word is misspelled, choose *cross.*

1.3. Correct the spelling of the incorrect word.

2. Read and validate the transcription:

2.1. If the transcription correctly represents the standard pronunciation of the word above, choose *thick.*

2.2. If the transcription is incorrect, choose *the cross.*

2.3. Correct the transcription using the DC-Arpabet.

2.4. Click *next* to submit the task and move on to the next entry.

4.3 Summary

This chapter addressed our proposed methodology for the two main projects – Normalizer expansion and improvement (pt-PT) and G2P models (sv-SE and ru-RU). This chapter was a crucial step for the subsequent phonetic lexica analysis.

Chapter 5
Results and Discussion

The following chapter centers on the results of the two main projects carried out under the Normalizer and Grapheme-to-phoneme scope.

Firstly, Section 5.1 discusses how we evaluated the Normalizer and G2P models according to well-known metrics and how we adapted the metrics to our models' needs. WER was adapted to account for normalizable tokens, instead of all the tokens. Section 5.1.2 describes the evaluation set we used to evaluate the new version of the normalizer. Thirdly, in Section 5.2, we focus on the evaluation results concerning the project that aimed to create a unique and better version of the pt-PT Normalizer tool, displaying the new model's evaluation results regarding each rule and its overall performance. Thus, we briefly compare version 1 of the normalizer with version 2.

Fourthly, in Section 5.3, we focus on the evaluation results regarding creating the G2P models for Swedish (Section 5.2.1) and Russian (Section 5.2.2). We discuss the phonetic lexicon performance results, per phone performance results, and the general G2P model performance results. Additionally, we compare the overall performance of the two different G2P models.

5.1 Normalizer and G2P evaluation

5.1.1 Evaluation metrics

When evaluating the Normalizer and the G2P, we used the following metrics, mentioned in Section 3.2.1.4 – Accuracy, WER, Precision, Recall, and F1 score. Therefore, as for the Normalizer, accuracy would tell if the normalizer output exactly matched the reference output. WER calculated the edit distance over the total number of tokens in the reference tokens. Precision, Recall, and F1 score calculate the proportion of correctly predicted normalized tokens, identifying a rule that should fire to certain instances and sentences. As

for the G2P, Precision, Recall, and F1 score would outline the proportion of correctly predicted grapheme to phone correspondences, where 1 represents total accuracy.

When evaluating the Normalizer tool, a problem related to using WER as a metric for assessing the normalizer is that in any given sentence – even if we filter only for sentences containing some normalizable expression – most of the tokens are likely to be irrelevant to the normalizer. Thus, WER divides edit distance over all tokens in the reference; however, most tokens we would not expect to be normalized anyway, so applying WER to the Normalizer distorts the results. Instead, using the WERnorm (Word Error Rate over Normalizable tokens) metric, we divide by the number of normalized reference tokens.

$$WER_{norm} = \frac{\sum_{x,y \in X,Y} edit\ distance_{x,y}}{\sum_{y \in Y} \max(1, normalizable\ tokens_{x,y})}$$

Equation 6. Word Error Rate over Normalizable Tokens formula.

In Table 23, the sentence expects only the *40 sq. mi.* to be of interest. Very few of the tokens need to be normalized. Taking that into account, the WERnorm metric only divides over those tokens in the reference.

Original	The	estate	covers	40	sq	.	mi	.	of	space
Reference	The	estate	covers	forty	square	miles	of	space		
Hypothesis	The	estate	covers	four	zero	square	mile	of	space	

Table 22. Word Error Rate (WER) and Word Error Rate over Normalizable Token[17].

In Equation 6, we see WER calculating edit distance over the total number of tokens in the reference tokens. On the other hand, when using the WERnorm metric Equation 6. Word Error Rate over Normalizable Tokens formula. We calculate the edit distance over the number of normalized reference tokens, transforming the total number of 0.375 (3/8) into 1 (3/3) WER.

[17] Orange shading = Normalizable reference tokens; Green shading = Substitutions, insertions, deletions

5.1.2 Evaluation set Normalizer

After expanding and improving the normalizer for pt-PT, we evaluated its performance quantitively on a standard evaluation set for each rule. Regarding the normalizer's overall performance, we wanted to know:

1. What is the percentage difference between the canonical normalization of a sentence and the normalizer's prediction?
2. How well does each normalizer rule detect normalizable expressions of distinct types?

Therefore, we evaluated the normalizer using a set of approximately 1000 reference sentences, manually normalized and tagged with the normalization rule which should be applied (*e. g.*, real numbers, symbols, among others). Then, we ran the normalizer on these sentences and compared its output with the reference.

The evaluation set was created in PyCharm[18] and had the following distribution:

- ID number
- Original sentence
- Reference sentence
- Tokens
    - Originals tokens
    - Reference tokens
    - Number of reference tokens
    - Number of normalizable reference tokens
- Tags
    - Type (rules)
    - Extent (normalizable token)
    - Start (count of tokens)
    - End (count of token)

[18] https://www.jetbrains.com/pycharm/

Thus, we present an example taken from the test set referred to above:

ID: 1
Original: XI Seminário Nacional de Sustentabilidade do Sistema Sustentabilidade:
Reference: décimo primeiro Seminário Nacional de Sustentabilidade do Sistema Sustentabilidade:
Tokens:
original_tokens: XI, Seminário, Nacional, de, Sustentabilidade, do, Sistema, Sustentabilidade, :,
reference_token: décimo, primeiro, Seminário, Nacional, de, Sustentabilidade, do, Sistema, Sustentabilidade, :,
number_reference_tokens: 10
number_normalizable_reference_tokens: 2
Tags:
type: ordinals rule
extent: XI
start: 0
end: 2

Table 23. Test set example.

The overview of the evaluation set used to evaluate version 2 of the normalizer is shown in Table 24. It gives the number of sentences we used, the number of normalizable reference tokens, and the number of reference tokens. Therefore, out of 23 428 reference tokens given in the 1000 sentences, only 31,2 % are normalizable tokens.

	Count	**%**
Sentences	1 000	-
Normalizable Reference Tokens	7 312	31,2%
Reference Tokens	23 428	-

Table 24. Evaluation set Overview for the version 2 of the normalizer.

All 1000 sentences present various normalizable tokens and for each a rule was applied. However, there is a lack of ordinals, for example, in the sentences which gives a lower (1,4%) number of normalizable ordinals when comparing with real numbers (49,9%). Consequently, this could have an impact on the further results that we will present in the next chapter.

Rules	Count	%
Abbreviations rule	474	24,7%
Ordinals rule	26	1,4%
Real Numbers rule	957	49,9%
Currency rule	215	11,3%
Symbols rule	147	7,7%
Measurements rule	50	2,6%
Dates rule	21	11,1%
Time rule	25	1,3%

Table 25. Rules applied to the normalizable tokens and their weight.

5.2 Normalizer results

In this section, we compare how each rule performed on each version of the normalizer (focusing on version 2) while discussing rules' behavior regarding the number of normalizable reference tokens and its final evaluation. Next, we present results on normalizers' performance regarding WER, WERnorm, and Accuracy metrics. Briefly, we wanted to know: How well did version 2 of the normalizer perform compared to version 1?

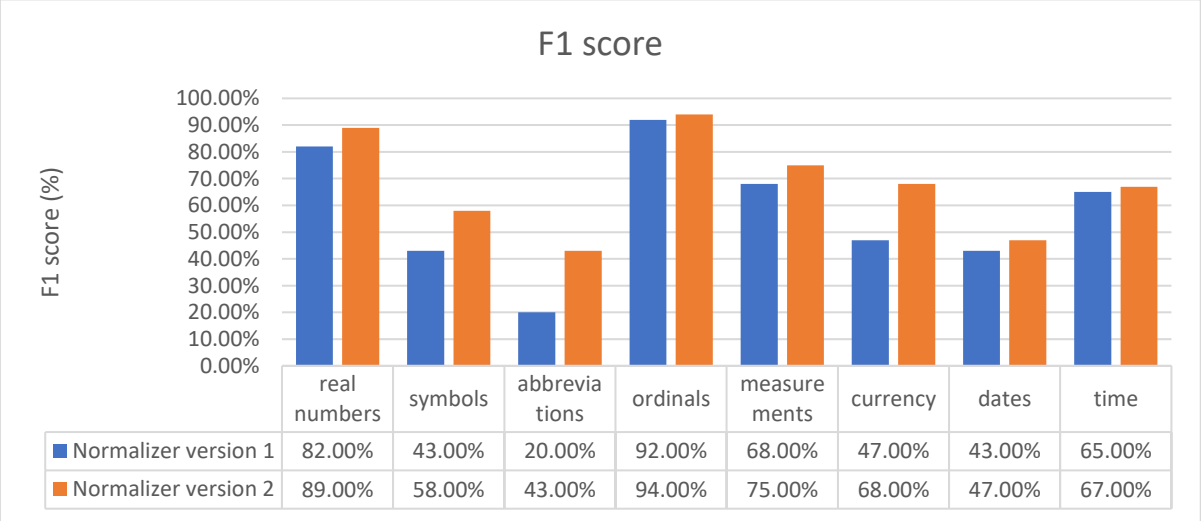
In order to compare normalizers' version 1 results with normalizers' version 2, we executed the evaluation using an evaluation set mentioned in Section 4.3.2. In the following tables, we present both versions of the normalizers' performance using the three metrics –F1-score (Table 26), Precision (Table 27), and Recall (Table 28) – for all rules applied in the evaluation set. Therefore, all tables present both versions of normalizers' performance on the following rules: real numbers, abbreviation, ordinals, measurements, currency, dates, and time.

Table 26 represents both versions of the normalizers' performance regarding the F1-score metric. As for version 1, we find statistically significant differences between the rules. The ordinals rule exhibits the highest percentage (92%), on the one hand, and abbreviations the lowest percentage (20%), on the other hand. Real numbers (82%), measurements (68%), currency (65%), time (65%), and dates and symbols (43%) exhibit a considerable variation.

As for version 2, we notice statistically significant discrepancies among the rules. The ordinals rule exhibits the highest percentage (94%), and abbreviations (43%) are the lowest

percentage. Real numbers (89%), measurements (75%), currency (68%) time (67%), symbols (58%), and dates (47%) exhibit noteworthy variations. Considering the evaluation set, in detail in Section 4.3.2, in all 1000 sentences, the number of normalizable abbreviations is higher (24,7%) when compared to the number of normalizable ordinals (1,4%), which could explain the better performance of the ordinals rule and the worst performance of the abbreviations rule. Continuously, measurements and time also may have satisfactory performance (75%, and 67%, respectively) if we consider the small number of normalizable measurements (2,6%) and time (1,3%) on the evaluation set. However, the same behavior does not happen with the real numbers rule and currency rule, which presents a higher performance overall (89%, and 68%, respectively), even though the number of normalizable real numbers (49,9%) and currency (11,3%) is high. We conclude that when presenting a satisfactory performance, even if the number of normalizable tokens is high, the real numbers and currency rule perform well in every normalizable real number and normalizable currency it finds.

Thus, if we consider the recall metric (Table 28), the low F1 score for abbreviations and time can be explained by abbreviations' low recall (44%) and times' low recall (58%). A low recall indicates that when tokens are not correctly identified, the F1-score decreases.

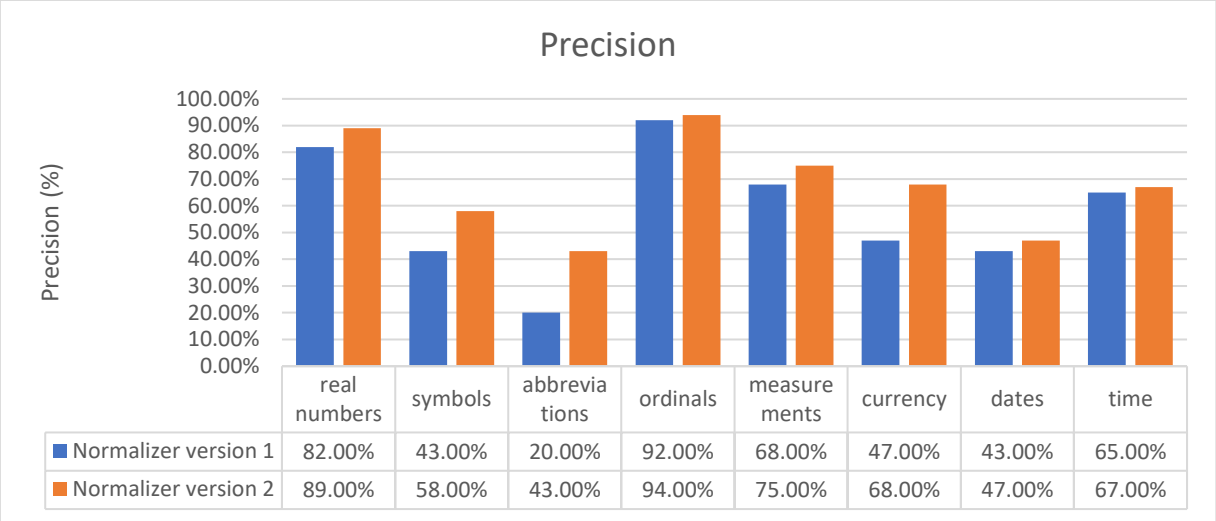


	real numbers	symbols	abbreviations	ordinals	measurements	currency	dates	time
Normalizer version 1	82.00%	43.00%	20.00%	92.00%	68.00%	47.00%	43.00%	65.00%
Normalizer version 2	89.00%	58.00%	43.00%	94.00%	75.00%	68.00%	47.00%	67.00%

Table 26. Normalizers' performance (version 1 and 2) regarding F1-score metric.

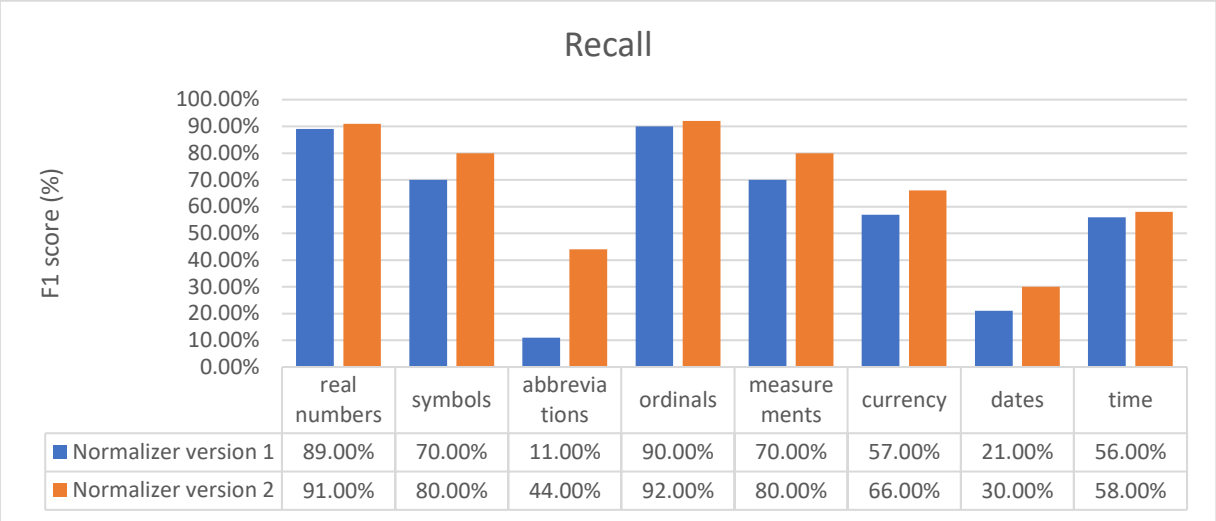
After analyzing the F1-score results on each rule, we conclude a significant performance increase in some of the rules. For example, abbreviations show a 23-percentage point (pp.) higher performance in comparison to symbols (2 pp.) and ordinals (2 pp.). Currency has the second higher performance with a 21 pp. difference, following symbols (15 pp.), real number and measurements (7 pp.), dates (4 pp.), and ordinals (2 pp.), respectively.

Indeed, we added more abbreviations, currency symbols, and symbols to the RMs than new real numbers, new data formats, or new ordinals. These last rules had a lower improvement because they were not subject to more additions of new tokens. Instead, we were more interested in upgrading the rule itself.



	real numbers	symbols	abbreviations	ordinals	measurements	currency	dates	time
Normalizer version 1	82.00%	43.00%	20.00%	92.00%	68.00%	47.00%	43.00%	65.00%
Normalizer version 2	89.00%	58.00%	43.00%	94.00%	75.00%	68.00%	47.00%	67.00%

Table 27. Normalizers' performance (version 1 and 2) regarding Precision metric.



	real numbers	symbols	abbreviations	ordinals	measurements	currency	dates	time
Normalizer version 1	89.00%	70.00%	11.00%	90.00%	70.00%	57.00%	21.00%	56.00%
Normalizer version 2	91.00%	80.00%	44.00%	92.00%	80.00%	66.00%	30.00%	58.00%

Table 28. Normalizers' performance (version 1 and 2) regarding Recall metric.

Next, Table 29 evaluates the normalizer regarding WER as a performance metric. Since this metric will consider all reference tokens (68,8%) rather than just the normalizable tokens (31,2%) in the evaluation set, as detailed in Section 4.3.1, it should guarantee a WER of around 35%. Section 4.3.1 explains in detail why we considered a formulation of the WER metric formula. Therefore, the following performance metric we use is Word Error Rate over Normalizable tokens (WERnorm). Considering that this formula will ponder only the normalizable tokens, it should guarantee a better WERnorm of around 11%. Given the possible misleading information the WER metric gives us and the 24 pp. difference between WER and WERnorm, we will only compare the results considering WERnorm.

Normalizer pt-PT	**WER**	**WERnorm**	**Accuracy**
Normalizer version 1	35,29%	10,58%	74,09%
Normalizer version 2	40,13%	12,47%	46,96%

Table 29. Normalizers version 1 and version 2 performance regarding WER, WERnorm, and Accuracy metrics.

Therefore, the normalizer version 2 shows, on average, a 4 pp. lower error rate over normalizable tokens compared to version 1. We also concluded the use of the ordinals rule (94% F1-score) and real numbers rule (89% F1-score) is the largest source of improvement on normalizer WER.

Furthermore, regarding accuracy, version 2 shows an improvement of, on average, 28 pp in compared to version 1. Overall, this clearly shows a significant improvement in the normalizer in all the rules.

5.3 G2P results

In this section, we present Swedish and Russian G2P results regarding its phonetic lexicon and the final G2P model. As for the phonetic lexicon we describe and analyze the final phone set, which includes graphemes, phonetic symbols, phone classification, and examples. Next, we give an overview of the final phonetic lexicon, including the number of total entries, correct and incorrect words, and correct and incorrect transcriptions. As for the final G2P model, we describe and discuss phone performance results regarding F1-score, Precision, Recall, and Accuracy metrics. Finally, we compare G2P models (Swedish and Russian) overall performance regarding F1-score and Accuracy metrics.

5.3.1 Swedish

To create a new G2P model for Swedish we first need a phonetic lexicon. The first task to make a phonetic lexicon is defining a phone set that covers all language sounds. Hence, for Standard Swedish, we defined 36 phones, though the number of phones is fewer than this. The G2P model developed was used with speech recognition applications; therefore, acoustic variability is to be considered while transcribing orthographic words. The phone set was designed to cover the entire acoustic space of Standard Swedish[19]. The final phone set includes Swedish graphemes, its equivalent phonetic alphabet symbols (IPA, and DC-Arpabet symbols), phone classification, and examples (orthographic words and phonetic transcription). For example, grapheme <a> has two equivalent phonetic symbols: a low front unrounded vowel [a]/[aa] (IPA and DC-Arpabet, respectively), and a low back unrounded vowel [ɑː]/[ah]. For phone [a], the orthographic word *c**a**lén* should have the following

[19] Table of the referred Swedish phone set is available in the Appendix section –Table 33.

phonetic transcription: [kk **aa** ll ee nn]. On the other side, for phone [ah], the orthographic word ***a**vsnitt* should have the following phonetic transcription: [**ah** vv ss nn ih tt].

After creating and validating the Swedish phone set, we prepared a phonetic lexicon (detailed in Section 4.2.1.). Thus, the phonetic lexicon trails the following structure: In alphabetic order, firstly it presents a Swedish word, followed by its phonetic transcription using the DC-Arpabet phones. A straight bar separates the orthographic word from the phonetic transcription word. The output looks like the following:

åsbrink | ob ss bb rr ii ng kk
åström | ob ss tt rr eu mm
åtta | ob tt ah

Figure 13. Examples from the Swedish phonetic lexicon.

After the phonetic lexicon was revised and corrected by a native linguist (see Section 4.2.3.), the final output was given (Table 29):

Swedish Phonetic Lexicon Overview

	#	%
Lexicon entries	25248	100
Corrected word	25067	99
Incorrect word	181	1
Correct transcription	22339	88
Incorrect transcription	2909	11
WER		10

Table 30. Normalizers version 1 and version 2 performance regarding WER, WERnorm, and Accuracy metrics.

When analyzing the lexicon's revision (Table 30), we find statistically reliable results. 99% of the orthographic words are correct, leaving only 1% of incorrect words. The most common errors regarding orthographic words are in 1. Double letters (*e. g.*, *u**tt**lardet* – *u**t**lardet*); 2. Diacritics (*e. g*., *genéve* – *genève*). Thus, due to errors in the transcribed audio used to create the initial lexicon, we can observe 1. Segment position alteration - Metathesis (*e. g*., *vädning* – *vending*), and 2. Segment suppression in the middle of the word - Syncope (*e. g.*, *löv**s**tedt* – *löv_tedt*). Consequently, errors in the orthographic words will be followed in the phonetic transcriptions as well.

Regarding phonetic transcriptions, we find 88% of transcriptions are correct and 11% incorrect. The most common errors of incorrect transcription are in the following vowels:

1. Mapping of graphemes <ä, o> - these graphemes have different equivalent phones; however, their occurrence does not always depend on phonological rules (e. g., [**eh** gg oo] – [**ae** gg ug]). /o/ was mostly transcribed as [oo], and /ä/ as [eh] instead of [ae].
2. Vowel reduction in /i/ - weakening of a vowel in an unstressed position (e. g., [hh **ih** ll **ih** ng shx eu] – [hh **ii** ll **ii** ng ss eu]). The generated transcription reduced the vowel /i/ whereas the correct transcription should be [ih] instead of [ii].

Regarding the phonetic lexicon performance, we achieved a WER of 10%. We consider this lexicon to be of very good quality and ready to use for the G2P model.

Regarding per phone evaluation, we tested each phone in terms of Precision, recall, and F1-score (Table 31). We conclude that the best performed phones are consonants [bb, dd, ff, hh, kk, ll, mm, nn, rr, ss, tt, vv]. Phones with a perfect F1 score are the following two: [ff, rr]. Therefore, consonants are performing better than vowels. The phone [ug] has the worst performance with an F1-score of 81%, following [ii] (83%), [uu] (85%), [ah] (86%), and [oo] (87%). The average Precision, Recall, and F1-score present a good result of 96% per phone. Thus, in total, all phones present 97% of accuracy.

Phones	Precision	Recall	F1-score
aa	95%	96%	96%
ae	93%	93%	93%
ah	87%	84%	86%
bb	98%	99%	99%
cc	87%	88%	88%
dd	99%	99%	99%
ee	89%	88%	89%
eh	97%	97%	97%
eu	90%	89%	90%
ff	100%	98%	100%
gg	97%	98%	98%
hh	98%	100%	99%
ih	95%	96%	96%
ii	83%	82%	83%
iu	92%	90%	91%

iy	88%	88%	88%
kk	99%	99%	99%
ll	99%	98%	99%
mm	100%	99%	99%
ng	97%	96%	97%
nn	99%	99%	99%
oe	87%	89%	88%
oh	89%	90%	89%
oo	87%	86%	87%
pp	98%	97%	98%
rr	100%	100%	100%
shx	94%	94%	94%
ss	98%	99%	99%
tt	99%	98%	99%
tt_ss	95%	95%	94%
uc	93%	92%	95%
ug	80%	82%	81%
uo	95%	96%	95%
uu	87%	84%	85%
vv	99%	99%	99%
yy	91%	90%	91%
Average	96%	99%	96%
Accuracy			97%

Table 31. Per phone results on the Swedish G2P.

5.3.2 Russian

To create a new Russian G2P model we preceded with the same steps as on the Swedish G2P model creation. For the phonetic lexicon, we defined a phone set that covers all language sounds. Hence, for Standard Swedish, we defined 43 phones, though the number of phones is fewer than this. The phone set was designed to cover the entire acoustic space of Standard Russian [20].

The final phone set includes Russian graphemes, its equivalent phonetic alphabet symbols (IPA, and DC-Arpabet symbols), phone classification, and examples (orthographic words

[20] Table of the referred Russian phone set is available in the Appendix section –Table 34.

and phonetic transcription). For example, according to palatalization in Russian[21], non-palatalized <б> has the equivalent phonetic symbol: voiced bilabial plosive [b]/[bb] (IPA and DC-Arpabet, respectively). Palatalized <бь> has the equivalent phone [bʲ]/[bbj] classified as palatalized voiced bilabial plosive. On the other side, iotated vowels <е, ё, ю, я> can have the equivalent phones [ja, je, jo, ju] (IPA), [yya, yye, yyo, yyu] (DC-Arpabet).

After creating and validating the Russian phone set, we prepared the phonetic lexicon (detailed in Section 4.2.2.). The phonetic lexicon trails the same structure as the Swedish phonetic lexicon (see Section 5.3.1). The output looks like the following:

авторов | aa vv tt oo rr oo vv
авторы | aa vv tt oo rr ie
агитка | aa gg ii tt kk aa

Figure 14. Examples from the Russian phonetic lexicon.

After the phonetic lexicon was revised and corrected by a native Linguist the final output was given (Table 32):

Russian Phonetic Lexicon Overview

	#	%
Entries	27340	100
Corrected words	26286	96
Incorrect words	1054	4
Correct transcriptions	23989	88
Incorrect transcriptions	3351	12
WER		11

Table 32. Russian phonetic lexicon overview.

Table 33 contains the results of our analysis of the lexicon revision, which are statistically meaningful. There are just 4% incorrect words and 96% of correct words. We observed the following errors in orthographic Russian words: 1. Segment suppression in the middle of the word- Syncope (e. g., а_нализа – а**з**нализа), and 3. Diacritics suppression (e. g., взл**е**ты –

[21] Palatalization is explained in Section 3.1.2.

взл**ё**ты). Consequently, errors in the orthographic words will be followed in the phonetic transcriptions as well.

Regarding phonetic transcriptions, we find 88% of transcriptions are correct and 12% incorrect. The most common errors of incorrect transcriptions are due to:

1. Sonorization –sound change where a voiceless consonant becomes voiced (*e. g.,* [**ss** oo zz dd aa nn ii yye] – [**zz** oo zz dd aa nn ii yye]).
2. Distinction between vowels and iotated vowels – vowels before <*ж, ш,* н> *are not iotated* (*e. g.,* [aa vv aa nn ss ts **yye** nn uu] – [aa vv aa nn ss ts **ee** nn uu])
3. Mapping of vowel <и> – its regular equivalent phone is [ii], however it can also be [ie] (*e. g.,* [mm oo ss kk vv **ii**] – [mm oo ss kk vv **ie**]).

We got a reasonable WER of 11% for the phonetic lexicon performance. This lexicon is of very good quality and ideal for the G2P model.

Regarding per phone evaluation, we tested each phone regarding Precision, Recall, and F1-score (Table 33). We conclude that the best performed phones are mainly consonants [gg, zx zx, ll, mm, nn, rrj, ts, txj]. Phones with a perfect F1 score are predominantly palatalized consonants [zj, llj, mmj, nnj]. Indedd, consonants are performing better than vowels. The phone [ii] has the worst performance with an F1-score of 83%, following [ie] (85%), iotated vowels [yya, yye, yyo, yyu] (86%), and consonant [zz] (87%). The average Precision, Recall, and F1-score present a good result of 95% per phone. Thus, in total, all phones present 96% of accuracy.

Phones	**Precision**	**Recall**	**F1-score**
aa	97%	97%	97%
bb	91%	89%	93%
bbj	96%	97%	97%
vv	90%	89%	92%
vvj	89%	88%	89%
gg	99%	99%	99%
dd	91%	90%	91%
ddj	98%	96%	98%
ee	85%	86%	86%
zj	100%	100%	100%

zx zx	99%	98%	99%
zz	87%	87%	87%
zzj	97%	96%	97%
ii	83%	82%	83%
yy	96%	95%	96%
kk	88%	88%	90%
ll	99%	99%	99%
llj	100%	100%	100%
mm	100%	99%	99%
mmj	97%	96%	100%
nn	99%	99%	99%
nnj	100%	100%	100%
oo	97%	96%	97%
pp	88%	87%	88%
ppj	96%	97%	96%
rr	98%	98%	98%
rrj	99%	99%	99%
ss	98%	97%	98%
ssj	95%	96%	96%
tt	96%	95%	96%
ttj	97%	97%	97%
uu	95%	96%	96%
ff	95%	96%	95%
xx	96%	95%	96%
ts	99%	99%	99%
tx	97%	96%	97%
sj	98%	98%	98%
ie	85%	84%	85%
yya	85%	84%	86%
yye	85%	84%	86%
yyo	85%	84%	86%
yyu	85%	84%	86%
txj	99%	99%	99%
Average	95%	94%	95%
Accuracy			96%

Table 33. Per phone results on the Russian G2P.

After testing phones' performance, we evaluated the Swedish and Russian G2P models regarding the F1-score metric. The Swedish G2P model shows a better performance (98%), than the Russian model (96%). This indicates that Swedish orthography is more phonetically based than Russian. In our opinion, the Swedish language also has a more regular grapheme-to-phoneme mapping. Russian presents a complex and wide phone set (43 phones), whereas Swedish phone set is much simpler (36 phones).

5.4 Summary

This chapter focused on the steps towards expanded and improved Normalizer and G2P models for two languages. We started by showing how we evaluated the Normalizer, pointing out very promising results on version 2 of the tool. Regarding accuracy, version 2 improved by 28 pp., and regarding WERnorm, it improved by 4 pp. when compared with version 1. Following, we show the G2P model results. Regarding the phonetic lexicons, Swedish offers a WER of 10%, whereas Russian presents a minor difference (11%). Regarding the general G2P model performance, the Swedish model has an average of 98% F1-score, and the Russian model presents a lower difference (22 pp.) with 96%.

Chapter 6
Conclusions and Future work

In our present work, we focused on giving a general overview of the importance that linguistic knowledge has in preprocessing models used in Speech Technologies, focusing on the Normalizer and Grapheme-to-Phoneme models. Considering the work we did on the Normalizer, together with the ML team, we were able to:

1. Test version 1 and version 2 of the pt-PT Normalizer while using an extensive evaluation set.
2. Apply statistics and qualitative evaluations of the normalizer performance which was crucial to our results and improvements.

Normalizer version 2 presents, on average, a 10% WERnorm. Compared to version 1, a 2 pp. lower error rate. Regarding accuracy, it has an improvement of 28 pp. higher accuracy (74%) than version 1. Overall, ordinals and real numbers rule performed best.

Considering our work on the G2P models, together with Defined.ai and the help of external native linguists, we were able to obtain:

1. Initial phonetic lexicon for Swedish and Russian to use as a basis of our final phonetic lexica.
2. Accurate revision and correction of our phone sets and phonetic lexicons.
3. Statistics and qualitative evaluations of G2P models' performance and quality regarding linguistic knowledge we implemented.

The Swedish phonetic lexicon achieved a 10% WER, whereas the Russian phonetic lexicon a minor slightly lower WER (11%). Swedish phones present a higher accuracy (97%) than Russian phones (96%). Overall, the Swedish G2P model presents a better performance (98%), even though its difference is minor when compared to the Russian G2P model (96%).

With the results obtained, we managed to expand the pt-PT Normalizer tool with linguistic assets covering more text that require normalization. Hence, we improved the normalizer in ease of use by increasing each rule on the normalizer. On the other side, we also provided coverage for two more languages (Swedish and Russian) adding two new models, and phonetic lexicons to the G2P tool, which supports now 14 languages. Hence, we succeeded in preparing text data for scripted speech tasks. Consequently, these will have a significant impact on the company's speech framework and scripted speech pipelines (15% is speech pipeline). Also, version 2 of the normalizer has begun to be used in other Defined.ai projects, mainly in speech prompt collections. We observed that our expansion and improvement on the tool covered expressions that make up a sizable proportion of normalizable expressions, not limiting the utility of the tool but increasing the diversity it can give when delivering prompts, for example.

Based on the work developed, we can observe that having a rule-based approach to the Normalizer and G2P increases its accuracy and performance, representing a significant advantage in improving Defined.ai tools and speech pipelines. Moreover, our approach was also applied to other languages gaining very positive results and showing the relevance of the methodology applied in this thesis. Therefore, our work shows the relevance and meaningfulness of having linguistic knowledge on preprocessing tools.

Also, the work that we developed on the normalizer contributed to discussions on specific vs language-generic rules. In this sense, we see future possibilities of expanding the coverage of the current normalizer by implementing the real numbers and ordinals rule (best-performed rules) to a module that can be used for various languages. Languages in which real numbers and ordinals are represented similarly (e. g., magnitude/decimal separators, number base). The same could be done with other rules, although this still needs thorough research.

Bibliography

Alasadi, A., & Deshmukh, R. (2018). *Automatic Speech Recognition Techniques: A Review*. https://www.researchgate.net/publication/325296232

Armstrong, E., & Meier, P. (2005). *IPA Chart*. https://www.ipachart.com/

Bobrow, D. G., Kaplan, R. M., Kay, M., Norman, D. A., Thompson, H., Winograd, T., & Norman, D. A. (1977). GUS, A Frame-Driven Dia|og System. *Artificial Intelligence*, *8*(0977), 155–173.

Bollmann, M. (2019). *A Large-Scale Comparison of Historical Text Normalization Systems*. https://github.com/clarinsi/csmtiser

Bondarko, L. v. (2005). Phonetic and phonological aspects of the opposition of "soft" and "hard" consonants in the modern Russian language. *Speech Communication*, *47*(1–2), 7–14. https://doi.org/10.1016/j.specom.2005.03.012

Braga, D. (2006). *Grapheme-to-Phone Transcription Algorithm for Text-to-Speech Systems in European Portuguese*.

Charvát, K., & Kepka, M. (2021). Crowdsourced Data. *Big Data in Bioeconomy*, 63–67. https://doi.org/10.1007/978-3-030-71069-9_5

Delić, T., Suzic, S., Ostrogonac, S., & Etinski, S. (2017). *Multi-style Statistical Parametric TTS Development of Dialogue Systems for Serbian and Other South Slavic Languages View project*. https://www.researchgate.net/publication/321331101

Deri, A., & Knight, K. (2016). *Grapheme-to-Phoneme Models for (Almost) Any Language*. 399–408. https://en.wikipedia.org/wiki/Category:

Errattahi, R., & el Hannani, A. (2017). Recent advances in LVCSR: A benchmark comparison of performances. *International Journal of Electrical and Computer Engineering*, *7*(6), 3358–3368. https://doi.org/10.11591/ijece.v7i6.pp3358-3368

García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., & Herrera, F. (2016). Big data preprocessing: methods and prospects. *Big Data Analytics 2016 1:1*, *1*(1), 1–22. https://doi.org/10.1186/S41044-016-0014-0

He, K. (2021). *Automatic Speech Recognition: Breaking Down Components of Speech | Towards Data Science*. https://towardsdatascience.com/automatic-speech-recognition-breaking-down-components-of-speech-85d065061517

Hennebert, J., Hasler, M., & Dedieu, H. (1994). *Neural Networks In Speech Recognition*. https://www.researchgate.net/publication/2249623

Juang, B. H., & Rabiner, L. R. (1991). Hidden Markov Models for Speech Recognition. *Technometrics*, *33*(3), 251. https://doi.org/10.2307/1268779

Jurafsky, D., & Martin, J. H. (2022). *Speech and Language Processing*. https://web.stanford.edu/~jurafsky/slp3/

Khan, R. A., Scholar, R., Jhunjhunu, J. J. T. U., & Chitode, I. J. S. (2016). Concatenative Speech Synthesis: A Review. *International Journal of Computer Applications*, *136*(3), 975–8887.

Kłosowski, P. (2022). A Rule-Based Grapheme-to-Phoneme Conversion System. *Applied Sciences (Switzerland)*, *12*(5). https://doi.org/10.3390/app12052758

Kurata, G., Audhkhasi, K., & Kingsbury, B. (2019). *IBM Research advances in end-to-end speech recognition at INTERSPEECH 2019 | IBM Research Blog*. https://www.ibm.com/blogs/research/2019/10/end-to-end-speech-recognition/

Latif, S., Kim, I., Calapodescu, I., & Besacier, L. (2021). *Controlling Prosody in End-to-End TTS: A Case Study on Contrastive Focus Generation*. 544–551. https://europe.naverlabs.

Lenzo, K. A., & Black, A. W. (2000). *Diphone Collection and Synthesis*. http://festvox.org/pointyclicky

Makhoul, J., Kubala, F., Schwartz, R., & Weischedel, R. (1999). *Performance Measures for information extraction*.

Meyer, P. (2021a). *State Of The Art of Speech Synthesis*. https://towardsdatascience.com/state-of-the-art-of-speech-synthesis-at-the-end-of-may-2021-6ace4fd512f2

Meyer, P. (2021b). *State Of The Art of Speech Synthesis | Towards Data Science*. https://towardsdatascience.com/state-of-the-art-of-speech-synthesis-at-the-end-of-may-2021-6ace4fd512f2

Mundada, M., Kayte, S., & Kayte, C. (2015). *A Review of Unit Selection Speech Synthesis* (Vol. 5, Issue 10). https://www.researchgate.net/publication/283354632

Padgett, J. (2001). *Contrast Dispersion and Russian Palatalization*. Academic Press.

Palo, P. (2006). *A Review of Articulatory Speech Synthesis*.

Rajadnya, Prof. K. (2020). Speech Recognition using Deep Neural Network Neural (DNN) and Deep Belief Network (DBN). *International Journal for Research in Applied Science and Engineering Technology*, *8*(5), 1543–1548. https://doi.org/10.22214/IJRASET.2020.5359

Riad, T. (2014). The Phonology of Swedish. *The Phonology of Swedish*. https://doi.org/10.1093/ACPROF:OSO/9780199543571.001.0001

Rui, A., Rebordao, F., Al, M., Shaikh, M., Hirose, K., & Minematsu, N. (2009). *How to Improve TTS Systems for Emotional Expressivity*. http://www.w3.org/2001/XMLSchema-instance

Sar, V., & Tan, T.-P. (2019). ScienceDirect Applying Linguistic G2P Knowledge on a Statistical Grapheme-to-phoneme Conversion in Khmer. *Procedia Computer Science*, *161*, 415–423. https://doi.org/10.1016/j.procs.2019.11.140

Schultz, T., & Schlippe, T. (2014). *GlobalPhone: Pronunciation Dictionaries in 20 Languages*.

Shulby, C., Mendonça, G., & Marquiafável, V. (2013). *Automatic Disambiguation of Homographic Heterophone Pairs Containing Open and Closed Mid Vowels*. 126–137. http://www.nilc.icmc.usp.br/lacioweb/

Sproat, R., Black, A. W., Chen, S., Kumar, S., Ostendorf, M., & Richards, C. (2001). Normalization of non-standard words. *Computer Speech and Language*, *15*, 287–333. https://doi.org/10.1006/csla.2001.0169

Taylor, P. (2007). *Text-to-Speech Synthesis*.

Timberlake, A. (2014). *A Reference Grammar of Russian*.

Tyagi, S., Nicolis, M., Rohnke, J., Drugman, T., & Lorenzo-Trueba, J. (2020). *Dynamic Prosody Generation for Speech Synthesis Using Linguistics-Driven Acoustic Embedding Selection*. https://www.amazon.science/blog/

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., & Kavukcuoglu, K. (2016). *Wavenet: A generative model for raw audio*.

van Esch, D., & Sproat, R. (2017). *An Expanded Taxonomy of Semiotic Classes for Text Normalization*. https://doi.org/10.21437/Interspeech.2017-402

Vielzeuf, V., & Antipov, G. (2019). *Are E2E ASR models ready for an industrial usage?*

Wang, Y., Skerry-Ryan, R. J., Stanton, D., Wu, Y., Weiss, R. J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q., Agiomyrgiannakis, Y., Clark, R., & Saurous, R. A. (2017). *Tacotron: Towards End-to-End Speech Synthesis*. https://google.github.io/tacotron

Wells, J. C. (1995). *X-SAMPA.* https://www.wikifox.org/pt/wiki/Extended_Speech_Assessment_Methods_Phonetic_Alphabet

Yanushevskaya, I., & Bunčić, D. (2015). Russian. *Journal of the International Phonetic Association*, *45*(2), 221–228. https://doi.org/10.1017/S0025100314000395

Yu, M., Duy Nguyen, H., Sokolov, A., Lepird, J., Sathyendra, K. M., Choudhary, S., Mouchtaris, A., & Kunzmann, S. (2020). *Multilingual Grapheme-to-Phoneme conversion with byte representation*.

Zhang, H., Sproat, R., Ng, A. H., Stahlberg, F., Peng, X., Gorman, K., & Roark, B. (2019). Neural Models of Text Normalization for Speech Applications under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license. *Computational Linguistics*, *45*(2). https://doi.org/10.1162/COLI

Appendix

Phonetic lexicon revision

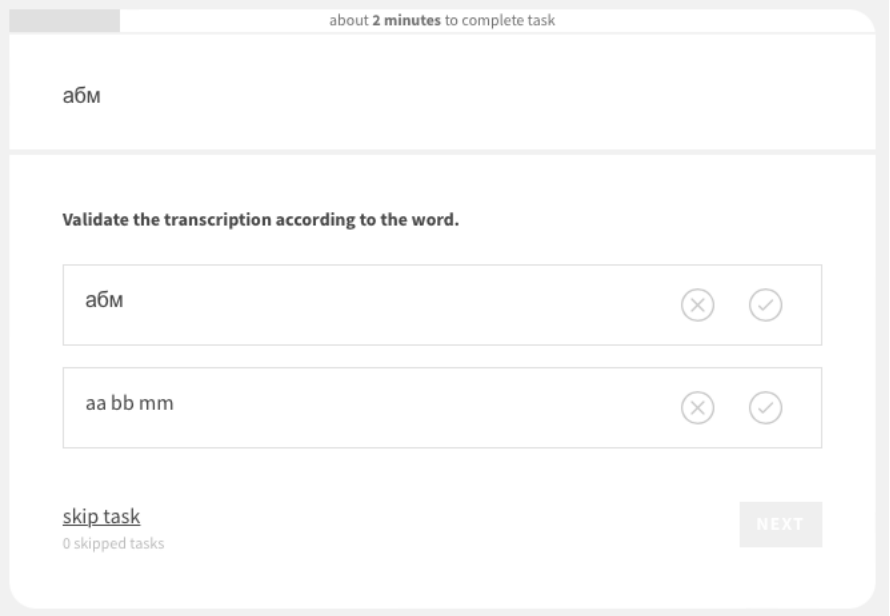


Figure 15. Text Variant Correction job in Neevo platform – Validation step for word.

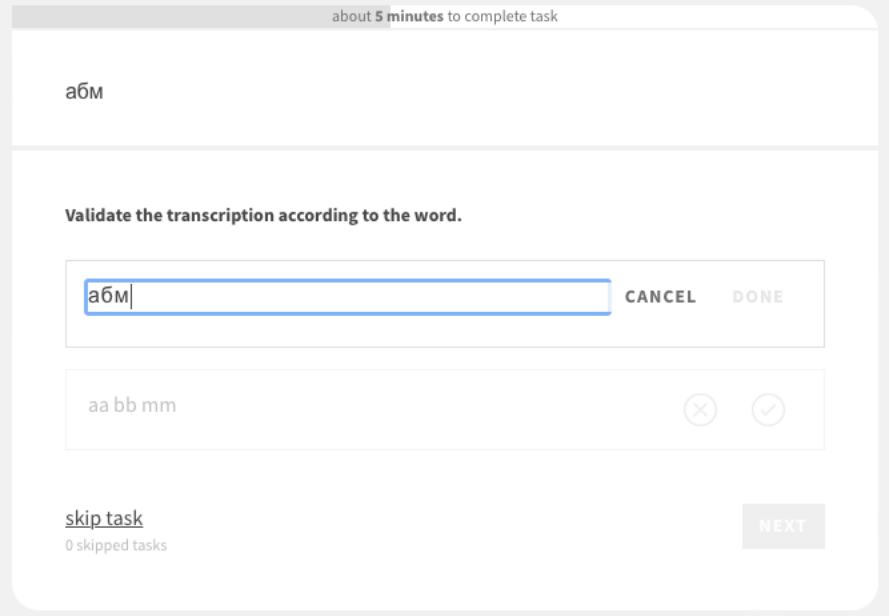


Figure 16. Text Variant Correction job in Neevo platform – Correction step for word.

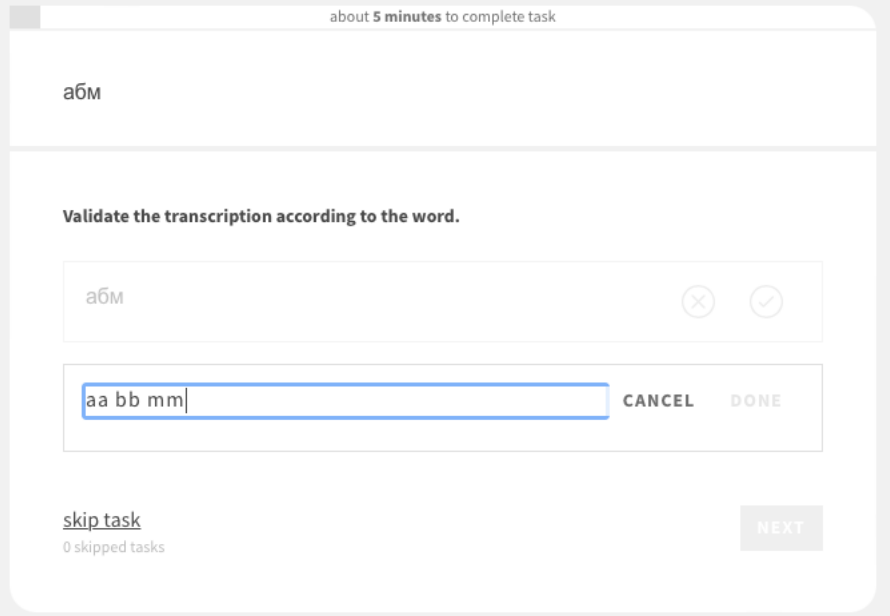


Figure 17. Text Variant Correction job in Neevo platform – Correction step for transcription.

No	Graphemes	Phonetic Alphabet Symbols		Classification	Examples	
		IPA	DC-Arpabet		Ortographic Words	Phonetic Transcription
1	<a>	[a]	[aa]	low front unrounded vowel	c**a**lén	kk **aa** ll ee nn
2	<a>	[ɑː]	[ah]	low back unrounded vowel	**a**vsnitt	**ah** vv ss nn ih tt
3	<e, ä>	[ɛ]	[eh]	low-mid front unrounded vowel	bank**e**r	bb aa ng kk **eh** rr
4	<ä>	[æː]	[ae]	near-low front unrounded vowel	bilv**ä**gar	ll vv **ae** gg aa rr
5	<e>	[eː]	[ee]	high-mid front unrounded vowel	bl**e**kt	bb ll **ee** kk tt
6	<i>	[ɪ]	[ih]	near-high front unrounded vowel	c**i**vila	ss **ih** vv ii ll aa
7	<i>	[iː]	[ii]	high front unrounded vowel	delt**i**d	dd ee ll tt **ii** dd
8	<o, ö, å>	[ɔ]	[oh]	low-mid back rounded vowel	eg**o**n	ee gg **oh** nn
9	<ä, o, å>	[oː]	[oo]	high-mid back rounded vowel	ekb**å**ge	bb **oo** gg eh
10	<ö, o>	[œ]	[oe]	low-mid front rounded vowel	f**ö**ljet	ff **oe** ll yy eh tt
11	<ö>	[øː]	[eu]	high-mid front rounded vowel	hörsammat	hh **eu** rr ss aa mm mm aa tt
12	<o, u>	[ʊ]	[ug]	near-high near-back rounded vowel	j**o**hnsson	yy **ug** nn ss oh nn
13	<u, o>	[uː]	[uu]	high back rounded vowel	k**ro**nkurs	kk **uu** nn kk uo rr ss
14	<y>	[ʏ]	[iu]	near-high front rounded vowel	kl**y**fta	kk ll **iu** ff tt aa
15	<u>	[yː]	[iy]	high front rounded vowel	k**y**lrum	cc **iy** ll rr uo mm
16	<u>	[ʊ]	[uo]	high-mid central rounded vowel	l**u**dvig	ll **uo** dd vv ih gg
17	<u>	[ʉ]	[uc]	high central rounded vowel	min**u**ten	mm ih nn **uc** tt eh nn
18	<i>	[j]	[yy]	voiced palatal approximant (semivowel)	og**i**ltigt	oo gg **yy** ih ll tt ih kk tt
19	<p>	[p]	[pp]	voiceless bilabial plosive	ö**pp**en	oe **pp pp** eh nn

20	<b>	[b]	[bb]	voiced bilabial plosive	**b**efogad	**bb** ee ff uu gg aa dd
21	<t>	[t]	[tt]	voiceless alveolar plosive	**t**ände	**tt** eh nn dd eh
22	<d>	[d]	[dd]	voiced alveolar plosive	**d**eltog	**dd** ee ll tt uu gg
23	<f>	[f]	[ff]	voiceless labiodental fricative	**f**amilj	**ff** aa mm ih ll yy
24	<v>	[v]	[vv]	voiced labiodental fricative	**v**ärld	**vv** ae rr dd
25	<s>	[s]	[ss]	voiceless alveolar sibilant	**s**vår	**ss** vv oo rr
26	<rs>	[ʂ]	[rr ss]	voiceless retroflex fricative	bä**rs**ebäck	bb aa **rr ss** eh bb eh kk
27	<k>	[ç]	[cc]	voiceless palatal fricative	be**k**änner	bb eh **cc** eh nn eh rr
28	<h>	[h]	[hh]	voiceless glottal fricative	**h**otell	**hh** ug tt eh ll
29	<l>	[l]	[ll]	voiced alveolar lateral approximant	**l**indrig	**ll** ii nn dd rr ih gg
30	<m>	[m]	[mm]	voiced bilabial nasal	**m**ånga	**mm** oh ng aa
31	<n>	[n]	[nn]	voiced alveolar nasal	**n**ära	**nn** ae rr aa
32	<ng>	[ŋ]	[ng]	voiced velar nasal	ri**ng**nér	rr ii **ng** nn ee rr
33	<sch>	[ɧ]	[shx]	voiceless postalveolar fricative	ägar**sch**ism	aa gg aa rr **shx** ih ss mm
34	<r>	[r]	[rr]	voiced alveolar trill	**r**ök	**rr** eu kk
35	<k>	[k]	[kk]	voiceless velar plosive	sve**k**	ss vv ee **kk**
36	<g>	[g]	[gg]	voiced velar plosive	**g**rönt	**gg** rr oe nn tt

Table 34. Standard Swedish phone set.

No	Graphemes	Phonetic Alphabet Symbols		Classification	Examples	
		IPA	DC-Arpabet		Ortographic Words	Phonetic Transcription
1	<а>	[a]	[aa]	low front unrounded vowel	б**а**нку	bb **aa** nn kk uu
2	<б>	[b]	[bb]	voiced bilabial plosive	газ**б**анка	gg aa zz **bb** aa nn kk aa
3	<бь>	[bʲ]	[bbj]	palatalized voiced bilabial plosive	тур**бь**ерн	tt uu rr **bbj** yye rr nn
4	<в>	[v]	[vv]	voiced labiodental fricative	тутако**в**а	tt uu tt aa kk oo **vv** aa
5	<вь>	[vʲ]	[vvj]	palatalized voiced labiodental fricative	ше**вь**ев	sj yye **vvj** yye vv
6	<г>	[g]	[gg]	voiced velar plosive	шпи**г**ель	sj pp ii **gg** yye llj
7	<д>	[d]	[dd]	voiced alveolar plosive	э**д**гар	ee **dd** gg aa rr
8	<дь>	[dʲ]	[ddj]	palatalized voiced alveolar plosive	бу**дь**те	bb uu **ddj** tt yye
9	<е, э>	[e]	[ee]	high-mid front unrounded vowel	встрез**э**	vv ss tt rr yye zz **ee**
10	<ж>	[ʐ]	[zj]	voiced retroflex fricative	вра**ж**най	vv rr aa **zj** nn aa yy
11	<жь>	[ʐʑ]	[zx zx]	voiced alveolo-palatal sibilant fricative (long variant)	побере**жь**ю	pp oo bb yye rr yye **zx zx** yyu
12	<з>	[z]	[zz]		бе**з**аль	bb yye **zz** aa llj
13	<зь>	[zʲ]	[zzj]	palatalized voiced alveolar sibilant fricative	во**зь**мется	vv oo **zzj** mm yyo ts yya
14	<и>	[i]	[ii]	high front unrounded vowel	вечерн**и**х	vv yye txj yye rr nn **ii** xx
15	<й>	[j]	[yy]	voiced palatal approximant (semivowel)	взаимно**й**	vv zz aa ii mm nn oo **yy**
16	<к>	[k]	[kk]	voiceless velar plosive	грузови**к**ов	gg rr uu zz oo vv ii **kk** oo vv

17	<л>	[l]	[ll]	voiced alveolar lateral approximant	грянул	gg rr yya nn uu **ll**
18	<ль>	[lʲ]	[llj]	palatalized voiced alveolar lateral approximant	губите**ль**но	gg uu bb ii tt yye **llj** nn oo
19	<м>	[m]	[mm]	voiced bilabial nasal	дела**м**и	dd yye ll aa **mm** ii
20	<мь>	[mʲ]	[mmj]	palatalized voiced bilabial nasal	вицепре**мь**ер	vv ii ts yye pp rr yye **mmj** yye rr
21	<н>	[n]	[nn]	voiced alveolar trill	име**н**но	ii mm yye **nn** nn oo
22	<нь>	[nʲ]	[nnj]	palatalized voiced alveolar nasal	де**нь**гам	dd yye **nnj** gg aa mm
23	<о>	[o]	[oo]	high-mid back rounded vowel	мин**о**вать	mm ii nn **oo** vv aa ttj
24	<п>	[p]	[pp]	voiceless bilabial plosive	митро**п**олит	mm ii tt rr oo **pp** oo ll ii tt
25	<пь>	[pʲ]	[ppj]	palatalized voiceless bilabial plosive	пьремьером	ppj rr yye mmj yye rr oo mm
26	<р>	[r]	[rr]	voiced alveolar trill	**р**езал	**pp** rr yye zz aa ll
27	<рь>	[rʲ]	[rrj]	palatalized voiced alveolar trill	**рь**одителям	**rrj** aa dd ii tt yye ll yya mm
28	<с>	[s]	[ss]	voiceless alveolar sibilant fricative	**с**ветят	**ss** vv yye tt yya tt
29	<сь>	[sʲ]	[ssj]	palatalized voiceless alveolar sibilant fricative	связало**сь**	ss vv yya zz aa ll oo **ssj**
30	<т>	[t]	[tt]	voiceless alveolar plosive	сме**т**у	ss mm yye **tt** uu
31	<ть>	[tʲ]	[ttj]	palatalized voiceless alveolar plosive	смотре**ть**	ss mm oo tt rr yye **ttj**
32	<у>	[u]	[uu]	high back rounded vowel	сообщ**у**	ss oo oo bb cx tx **uu**
33	<ф>	[f]	[ff]	voiceless labiodental fricative	специ**ф**ика	ss pp yye ts ii **ff** ii kk aa
34	<х>	[x]	[xx]	voiceless velar fricative	сроко**х**	ss rr oo kk aa **xx**

35	<ц>	[t͡s]	[ts]	voiceless alveolar affricate	демоппози**ц**ии	dd yye mm oo pp pp oo zz ii **ts** ii ii
36	<щ>	[t͡ɕ]	[tx]	voiceless alveolo-palatal affricate	теку**щ**их	tt yye kk uu **tx** ii xx
37	<ш>	[ʂ]	[sj]	voiceless retroflex fricative	ти**ш**ина	tt ii **sj** ii nn aa
38	<ы>	[ɨ]	[ie]	high central unrounded vowel	т**ы**сяче	tt **ie** ss yya txj yye
39	<е, ё, ю, я>	[ja]	[yya]	voiced palatal approximant to low back unrounded diphthong	уважени**я**	uu vv aa zj ee nn ii **yya**
40	<е, ё, ю, я>	[je]	[yye]	voiced palatal approximant to high-mid front unrounded diphthong	исте**ц**	ii ss tt **yye** ts
41	<е, ё, ю, я>	[jo]	[yyo]	voiced palatal approximant to high-mid back rounded dipthong	л**е**гкого	ll **yyo** xx kk aa vv aa
42	<е, ё, ю, я>	[ju]	[yyu]	voiced palatal approximant to high back rounded dipthong	лучшу**ю**	ll uu txj sj uu **yyu**
43	<ч>	[tʂ]	[txj]		нын**ч**е	nn ii nn **txj** yye

Table 35. Standard Russian phone set.