TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Faculty of Computer Science

IT-Based textgeneration using NLP methods

State of the art and design of a prototype

Bachelor Thesis in
Business Informatics and Management

from

Tim Löhr

Student ID 3060802

First advisor: Prof. Dr. Alfred Holl

Second advisor: Prof. Dr. Florian Gallwitz

Hinweis: Diese Erklärung ist in alle Exemplare der Abschlussarbeit fest einzubinden. (Keine Spiralbindung)

TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Prüfungsrechtliche Erklärung der/des Studierenden

Angaben des bzw. der Studierenden:

Name: Vorname: Matrikel-Nr.:

Fakultät: Studiengang:

Semester:

Titel der Abschlussarbeit:

Ich versichere, dass ich die Arbeit selbständig verfasst, nicht anderweitig für Prüfungszwecke vorgelegt, alle benutzten Quellen und Hilfsmittel angegeben sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

Ort, Datum, Unterschrift Studierende/Studierender

Erklärung zur Veröffentlichung der vorstehend bezeichneten Abschlussarbeit

Die Entscheidung über die vollständige oder auszugsweise Veröffentlichung der Abschlussarbeit liegt grundsätzlich erst einmal allein in der Zuständigkeit der/des studentischen Verfasserin/Verfassers. Nach dem Urheberrechtsgesetz (UrhG) erwirbt die Verfasserin/der Verfasser einer Abschlussarbeit mit Anfertigung ihrer/seiner Arbeit das alleinige Urheberrecht und grundsätzlich auch die hieraus resultierenden Nutzungsrechte wie z.B. Erstveröffentlichung (§ 12 UrhG), Verbreitung (§ 17 UrhG), Vervielfältigung (§ 16 UrhG), Online-Nutzung usw., also alle Rechte, die die nicht-kommerzielle oder kommerzielle Verwertung betreffen.

Die Hochschule und deren Beschäftigte werden Abschlussarbeiten oder Teile davon nicht ohne Zustimmung der/des studentischen Verfasserin/Verfassers veröffentlichen, insbesondere nicht öffentlich zugänglich in die Bibliothek der Hochschule einstellen.

Hiermit ☐ genehmige ich, wenn und soweit keine entgegenstehenden Vereinbarungen mit Dritten getroffen worden sind,

☐ genehmige ich nicht,

dass die oben genannte Abschlussarbeit durch die Technische Hochschule Nürnberg Georg Simon Ohm, ggf. nach Ablauf einer mittels eines auf der Abschlussarbeit aufgebrachten Sperrvermerks kenntlich gemachten Sperrfrist

von Jahren (0 - 5 Jahren ab Datum der Abgabe der Arbeit),

der Öffentlichkeit zugänglich gemacht wird. Im Falle der Genehmigung erfolgt diese unwiderruflich; hierzu wird der Abschlussarbeit ein Exemplar im digitalisierten PDF-Format auf einem Datenträger beigefügt. Bestimmungen der jeweils geltenden Studien- und Prüfungsordnung über Art und Umfang der im Rahmen der Arbeit abzugebenden Exemplare und Materialien werden hierdurch nicht berührt.

Ort, Datum, Unterschrift Studierende/Studierender

Preface I

The following thesis was created during the seventh and last semester at the Georg Simon Ohm University of Applied Science. Within the last three semesters, I realized that my major interest among all IT related topics is artificial intelligence.

My personal interest started basically with a group IT-project, in which my team and I programmed an autonomously driving remote control car with a deep neural network together with a Raspberry Pi 3B+. From this first project on, I selected all my further elective courses to be related to machine learning or data science in any possible way. I wanted to increase my knowledge further, so I searched for a website that provides courses related to AI. I found *www.udacity.com*, which offers courses in cooperation with top IT companies, such as Google, Airbnb, or Microsoft. Out of curiosity, I bought the course *Natural Language Processing*. After successfully finishing it, I was encouraged to write my bachelor thesis in a *Natural Language Processing* related topic. Together with my professor *Prof. Dr. Alfred Holl*, I worked out a method-matrix for the entire structure of this paper. Without his cooperative support overseas while I was studying abroad at the City University of Hong Kong, this thesis would not have been possible for me. Even though Natural Language Processing is just a subfield of machine learning, the current state-of-the-art research is far beyond what I can research within a bachelor thesis. In this way, I decided to write my thesis about the subfield *textgeneration* within NLP. My state-of-the-art research includes all *hot topics* within NLP, and my prototype focuses only on the text generation part, to dive deeper into what NLP and especially text generation can accomplish in the year 2020.

Preface II

For my research, I encountered a lot of old and recently published papers, mostly from *https://arxiv.org/*. To read through the papers requires a lot of prior knowledge, especially in mathematics, which I learned in my abroad semester in Hong Kong. To fully understand the mathematics given in this thesis, enhanced knowledge of calculus and linear algebra is required. Even if this is not the case, I will describe the process in such a way that it can be comprehended without looking at the maths.
Machine Learning and, more specifically, NLP is not an intuitive study. I provided for the matrix notations the common terminologies originated from top researchers and tried to make the entry into this field as smooth as possible if the reader has no prior knowledge about this topic. During the five-month development process of the bachelor thesis, I gained much knowledge. I recognized that NLP is a huge topic, constantly under research. To keep up to date with the latest publications requires much effort.
To give a full state-of-the-art review about *all* NLP related disciplines is not possible within this thesis. For this reason, I focus entirely on the development of the *Neural Text Generation* (NTP), which includes more fields than the reader might imagine.

	Titel / Kapitel		Untertitel / Unterkapitel	——Woher?—— Wissensinput	Frageinput	——Wie?—— Methode	——Was?—— Zielbeschreibung
0	**<u>IT-basierte Textgenerierung mit Hilfe von NLP-Methoden</u>** **State of the Art & Entwurf eines Prototypen**			Allgemeingültig: • Fachbücher, Bücher • HongKong, TH-OHM • Online Kurse	1. Was ist der State of Art von NLP - Systemen. 2. In welcher Qualität kann ich den Textgenerierungs- Prototypen selbst programmieren und welche Güte hat dieser?	1. Darstellung des State of the Art der NLP-Systeme. 2. Studium der relevanten Aspekte des NLP und Programmierung eines IT-basierten Textgenerierungs-Prototypen.	1. State of the Art fachlich herausarbeiten. 2. Einen Prototypischen Algorithmus programmieren, der zu einem gegeben Input z.B. ein Buch immer wieder neue kreative Fortsetzungen generiert.
1	**Einleitung**	**1.1**	**Fallbeispiel eines aktuellen NLP-Systems**	• [0.1] • Wissenschaftliches Schreiben und	1. Was sind aktuelle, nützliche Einsatzgebiete von NLP-Textverarbeitungs-Systemen? 2. Was ist der Nutzen meines NLP-Prototypen im Bereich der Textverarbeitung?	1. Recherche über die aktuellen und geplanten NLP-Systeme, im Bereich der Textverarbeitung. 2. Vorstellung meines Beitrags zu NLP-Systemen mithilfe meines Prototyps.	1. Antwort auf die Frage, warum meine Bachelorarbeit sinnvoll ist und welche Motivation ich habe zur Bearbeitung 2. Erläuterung durch einen interessanten leichten Einstieg.
2	**State of the Art**	**2.1**	**Relevante Aspekte der Mathematik**	[1.1]	Welches mathematische „know-how“ ist notwendig, um NLP-Systeme für Textverarbeitung und meinen Prototypen technisch verstehen zu können?	Recherche nach den relevanten Aspekten der Mathematik für dieses Thema.	Beschreibung der anwendungsbezogenen mathematischen Modelle für diesen Themenkomplex anhand von Formeln und Erklärungen.
		2.2	**Geschichte des NLP**	• [0] • [0.1]	1. Seit wann wird an NLP-Systemen geforscht? 2. Ab welchem Punkt konnte man effektiven Nutzen aus diesen Systemen ziehen?	1. Literaturrecherche über die Geschichte des NLP (40 Jahre). 2. Literaturrecherche über die ersten Einsätze der NLP-Systeme.	1. Darstellung der Geschichte des NLP in Form einer zeitlichen Abfolge. 2. Nutzen der ersten NLP-Prototypen oder Technologien die im Einsatz waren.
		2.3	**Aktuelle Trends der Technologie**	• [0] • [0.1] • [2.2] • Fallbeispiele	1. Was sind aktuelle NLP-Systeme imstande zu leisten? 2. Wo sind die Einsatzgebiete?	1. Literaturrecherche über aktuelle Trends (+ - 5 Jahre). 2. Recherche von aktuelle Papern und Veröffentlichungen.	1. Darstellung der aktuellen Technologien. 2. Blick in die kurzfristige Zukunft anhand von aktuellen Fallbeispielen und Forschungsergebnissen.
3	**Prototyp**	**3.1**	**Zielsetzung / Anforderungen**	• [0] • [1] • [2]	1. Was soll mein Prototyp mit gegebenen Mitteln leisten können? 2. Welcher Output ist im besten Fall zu erwarten?	1. Requirements Engineering. 2. Klassifizierung und Analyse möglicher Ergebnisse, z.B. ob der Output grammatikalisch korrekt ist.	1. Erläuterung des Umfangs meines Prototyps. 2. Sammlung und Klassifizierung der Anforderungen an den Algorithmus und dessen Output.
		3.2	**Fachkonzept**	[3]	1. Wie ist mein Prototyp strukturiert? 2. Welche Algorithmen verwende ich? 3. Welche Prozesse durchlaufen die zu verarbeitenden Daten? 4. Wie werden die Daten verarbeitet?	1. Erstellen eines Fachkonzepts 2. Algorithmus modellieren 3. Prozessmodellierung 4. Datenflussmodellierung und, oder Datenmodellierung	1. Fachkonzept fertig erstellt. 2. Der Prototyp wird ohne IT Bezug anhand von verschiedenen Teilmodellen modelliert. 3. Die einzelnen Prozesse werden ohne konkreten Implememtierungs-Vorschlag modelliert. 4. Datenverarbeitung visualisiert
		3.3	**Implementierung**	[3.3]	1. Welche Technologien verwende ich für meinen Prototypen: - „Welche Python Bibliotheken und IDE?“ - „Welche HW & SW-Anforderungen gibt es?“ 2. Welche Probleme traten bei der Programmierung auf?	1. Software-Abhängigkeits-Portfolio erstellen - Vergleich geeigneter Programmiersprachen - Recherche der erforderlichen Bibliotheken - Recherche der erforderlichen Hardware, Software und Auswahl 2. Software entwickeln - Fehler reporten an Hersteller, Bib, etc.	1. Erstellung eines IT-Konzepts in Form einer Beschreibung der notwendigen technischen Mittel anhand von Teilmodellen 2. Problemstellungen erklären und das Auftreten eines Problems „reverse Engineeren“
		3.4	**Evaluation**	[3.4]	1. Wie ist der Output des Prototyps zu bewerten? 2. Wie bewertet man die Qualität des Outputs? 3. Was kann verbessert werden?	1. Soll-Ist-Vergleich der Anforderungen mit dem Output des Prototypen. 2. Vergleich mit verwandten Arbeiten. 3. Recherche über potentielle Verbesserungen des Algorithmus.	1. Evaluation und Analyse des Ergebnisses anhand von grammatikalischer Richtigkeit und Sinn. 2. Bessere Ergebnisse mit meinen vergleichen. 3. Optimierungsmöglichkeiten für meinen Prototypen evaluieren.
4	**Generierung von übertragba-rem Wissen**			[0] bis [3]	Um welche Elemente könnte mein Projekt modular Erweitert werden um ein Anderes oder Besseres Ergebnis zu erzeugen und welchen Einfluss könnte es auf die Forschung haben?	Verallgemeinerung aus den bisher erarbeiteten Ergebnissen.	Einordnung der Evaluationsergebnisse in einen gesellschaftlichen Kontext.

Abstract

– At the end , finally finished :) –

Contents

Chapter 1.

Intro

The boundary between Natural Language Processing (NLP), Text Generation (TG) and *Neural Text Generation (NTP)*, is relatively blurred and overlap in many ways. Generally speaking, all of the NTG tasks are NLP based, but not all NLP tasks are NTG based. NTG is the latest improvement of TG, because it applies the latest Deep Learning (DL) research achievements.

1.1. Difference of NLP and NTG

In recent months and years, neural networks have produced many *state-of-the-art* results in almost all possible disciplines of machine learning [Xie 17]. The roots of Neural Networks (NN) lie down almost 80 years ago in 1943 when **McCulloch-Pitts** [McCu 43] compared for the first time neuronal networks with the structure of the human brain. This first attempt to approach artificial neurons with neurons from the brain leads to the nowadays commonly used understanding of a simple neuron of an underlying neural network (shown in figure 1.1). The connected neurons create an artificial neural network, which can calculate any possible logical or arithmetical function.

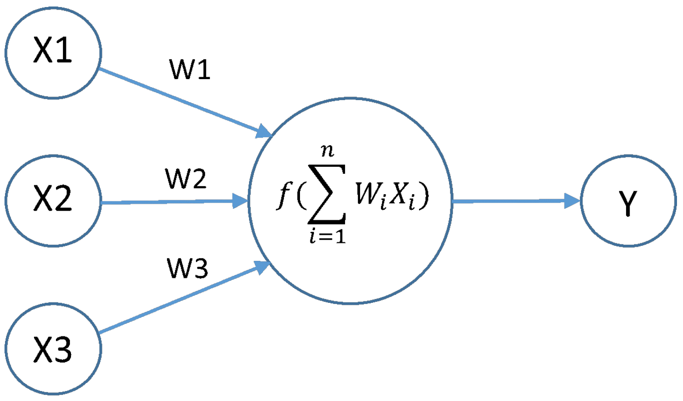

Figure 1.1.: A simple Neuron with 3 inputs and 1 output [Sing 17]

The range in which NN's (in the year 2020) apply to modern technologies is wide. Some disciplines have only been created due to the invention of neural networks because they solve existing- and new problems very effectively and efficiently. Many frequently held conferences around the globe contribute continuous evidence of the successes of neural networks. Among those various disciplines counts for example *Pattern recognition* with Convolutional Neural Networks (CNN) [Yann 98] or the famous *CIFAR-10* dataset [Kriz], where many amateurs [Löh 19] and experts attempt annually to further increase the accuracy of predicting the 10 different image classes.
The basis of this thesis's topic *textgeneration* is Natural Language Processing, *NLP* for short. This field covers many other hot research topics, such as

- Sentiment Analysis
- Machine Translation
- Speech Recognition
- Text Generation (Neural Text Generation *NTG*)
- Chatbots

Another term for text generation is *Language Modelling*, because text generators use the words of a language and grammar as input for the model. In the past five years, primarily two approaches were used for modelling NLP, namely the **rule-based** system and the **template-based** system (Figure 1.2) [Xie 17]. Today neural end-to-end systems are *state-of-the-art* [Jeka 17]. These systems offer more flexibility and scale with proportionately better results, and less data is required because of the increased complexity. A major disadvantage is that the necessary computing power has increased exponentially. However, this leads to a complex problem because it becomes more and more challenging to understand the decisions of the neural network. The neural network is still, to a large extent, a *black box.* Especially in NLP it gives surprisingly good results. The neural network models for text processing are difficult to understand, so nowadays, compromises between rule-based systems still have to be made, and hybrid systems are most commonly in use.

The neural text generation, also called *NTG*, has many other exciting application fields, which overlap partly with NLP, including

- Speech recording and conversion to text
- Conversation systems e.g. chatbots
- Text summary
- Caption generation

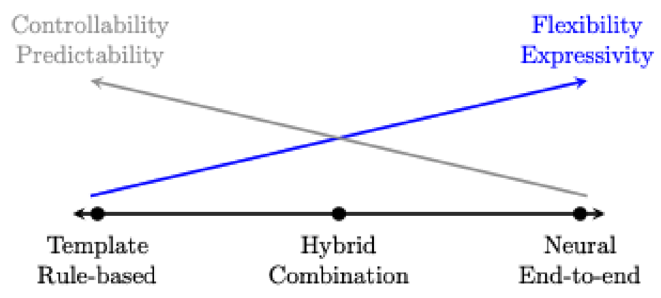

Figure 1.2.: Rule-Based vs. Neural-Text-Generations System [Xie 17]

In order to train language models properly, Deep Learning algorithms teach the model the probabilities of occurring words with respect to the preceding words. There are several approaches to achieve this goal. Language models can be trained on the level of words, whole sentences, or even whole paragraphs. The granularity in which the training takes place is called *n-grams*, where *n* represents the number of preceding words. Further explanation in subsection 2.1.2 of chapter 2.

1.2. Case study of a current NLP system

The following case introduces the underlying architecture of the famous **"Hey Siri"** command from Apple's *Siri*. There are many fascinating use cases, but the most crucial hot research topics of 2019-2020 are introduced in chapter 2.3. Siri is a prime example because it combines two of the most relevant NLP tasks, namely Speech Recognition and Text Generation. Apple released the first version of Siri in 2011 with the iPhone 4s for the public costumers. At this time, users were still forced to press the *Home Button* to give Siri a command via voice. For Siri to fulfill the user commands, it first needs to understand the human language itself by recognizing and splitting the words accordingly. Secondly, it needs to process those words further to a context and figure out what the user wants.

1.2.1. Hands-Free Access to Siri

For the release of the iPhone 6 (iOS 8) in 2015, Apple upgraded Siri to a large extent. Without the need to interact with the iPhone physically, it can now detect the primary-users voice *"Hey Siri"* to wake up automatically. Even though it might not sound like an innovation, still a lot is going on behind the scenes to create a flowless experience [Team 17].

The following figure 1.3 shows the entire process of detecting the wake-up sentence *"Hey Siri"*.

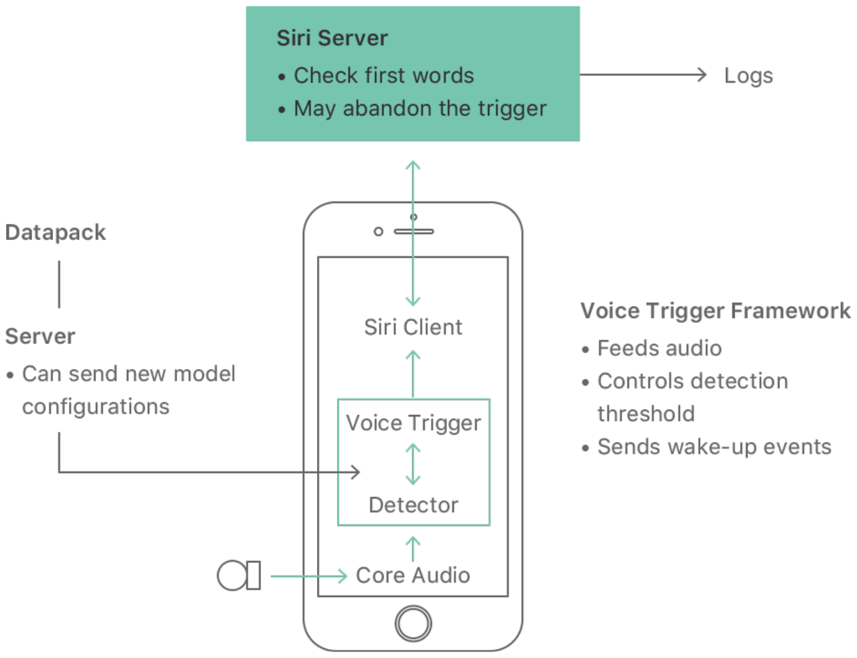

Figure 1.3.: The Hey Siri flow on iPhone (Apple 2017) [Team 17]

For computing the error, there exist two different metrics. The false-accept rate (FAR) denotes the number of false activations which occur every hour, and the false-reject-rate (FRR) counts who often Siri is not activated when it was asked. Those objectives are essential to denote because machine learning and text generation tasks usually have loss function. The training aims to reduce the loss, in the case of Siri, Deep Learning aims to reduce the two different error rates. Deep Learning was applied with a network consisting of five hidden-layers.

The recurrent state scores (figure 1.4) are achieved through the use of Recurrent Neural Networks (RNN) [Team 18a], multi-style training, and curriculum learning. Recurrent Neural Networks (RNN) will be further explained in subsection 3.2.2 of chapter 3. The spoken words are split into *phonemes* consisting of all sounds in the corresponding spoken language. The network was trained on tons of data, according to Apple [Team 17]. With this training, this *acustic model* is now able to funnel the user's voice through this network and distinguish whether the sentence was *"Hey Siri"* or not. The core In figure 1.3 is this acoustic model. The iPhone microphone always listens and converts all sounds into a stream of waveform samples. Those samples are transformed into a sequence of frames, which describe approximately 0.01 seconds in time. Those frames are fed into the Deep Neural Network (DNN) acoustic model and produce a log probability to calculate the probability of the current sound being the *"Hey Siri"* activation sentence. The network contains two different models, to further reduce the error rate by double-checking if the spoken sentence is understood properly.

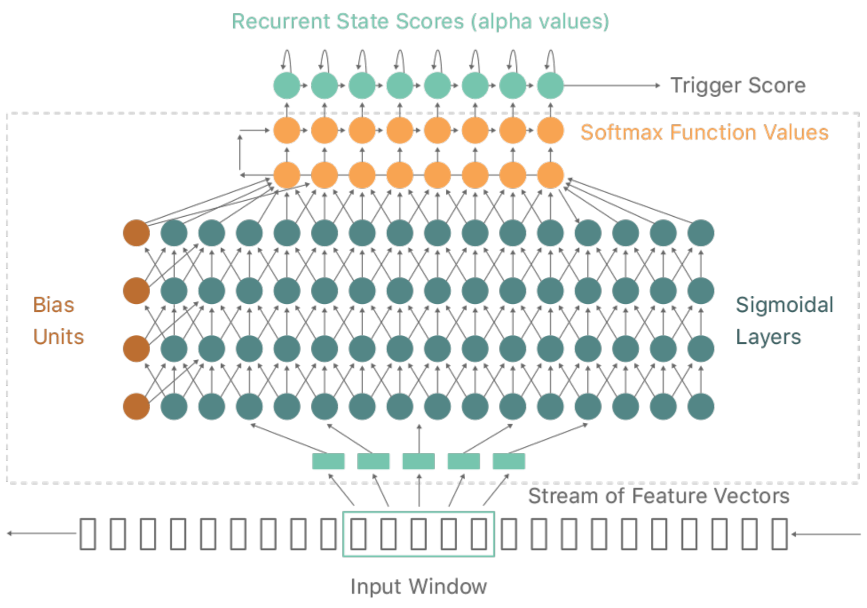

Figure 1.4.: Structure of the DNN with Recurrent State Scores (Apple 2017) [Team 17]

Many companies tend to store their high-tech technologies like speech recognition and NLP applications recently on the cloud. Apple uses cloud technologies as well as [Team 17] for Siri because the team behind Siri faced limitations regarding the battery life and the computing capacity of the CPU of the iPhone. So, the wake-up neural network is stored locally on the iPhone, whereas the model for understanding the generating the output to a question is stored on the cloud.

1.2.2. Personalized Hey Siri

Siri can recognize the words *"Hey Siri"*, but it responds to anyone, not only the primary user. To troubleshoot this error, Apple introduced the speaker recognition (SR) system.

The wake-up task is rather simple compared to figuring out the context, but still highly researched. When it comes to programming a model, the input language for this model gets split up into basically three parts as the first step[Team 18a].

- *Syntax*: Composition of the phrases
- *Semantics*: Meaning of the phrases
- *Pragmatics*: Composition and context of the phrases

Siri gets activated when it recognizes the words *"Hey Siri"*. The difference with the personalized system is that acoustic input now is further split up in the second step into the phonetic

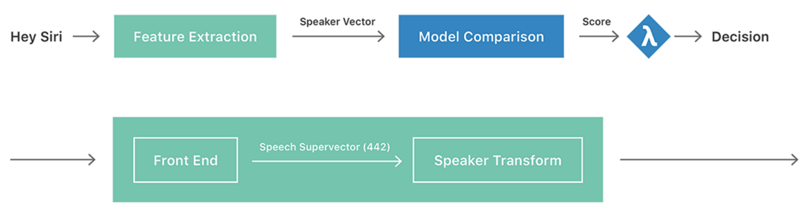

Figure 1.5.: Block diagram of Personalized "Hey Siri" (Apple 2018)[Team 18a]

content, background recording environment, and the speaker's identity (personalization), as in figure 1.5 (upper boxes) [Team 18a].

Siri is able to avoid unintended activations that sound similar but have a different meaning. Unintended activations are especially challenging, because people all over the world have different accents and dialects, depending on their origins.

When the phone is configurated, in the *enrolment stage*, the user is asked to repeat common phrases for a couple of times. This input is fed into a *statistical model* for recognizing the user's voice. In the *recognition stage*, the computer evaluates if the speech input fits the primary-users-trained model and accepts or rejects the request based on that decision.

Figure 1.5 from Apple shows the fundamental diagram for this process [Team 18a]. The *Feature Extraction* computes a fixed-length speaker vector for the input sentence *"Hey Siri"*. This vector contains information about phonetics, background noise, and the identity of the user. In the second step, the vector is transformed in such a way that the environmental background noise is reduced to a minimum with the help of the *Fourier Transform*, and the user's identity is extracted.

In the early days, Apple used different algorithms to train the speech recognition models. Apple used for example the *Linear Discriminant Analysis* (LDA) which scored very poor compared to the latest approach *Deep Neural Networks* (DNN) [Team 17]. The research paper from Apple states that the DNN achieves a speaker recognition error rate of 4.3%, whereas the LDA produces an 8.0% error rate. This performance test was conducted with an overall performance test. This shows that the personalized models score with much higher accuracy than before.

1.2.3. Regionally Specific Language Models for Speech Recognition

Geo-Language-Models

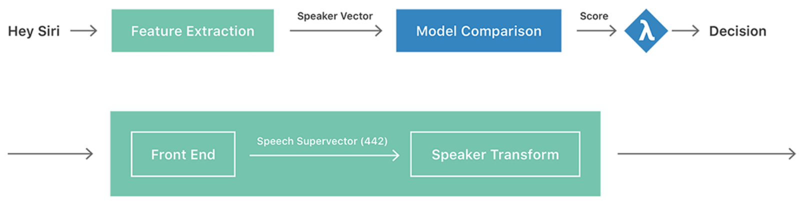

Figure 1.6.: System Overview (Apple 2018)[Team 18b]

Chapter 2.

State of the Art

2.1. Relevant aspects of mathematics

2.1.1. Mathematical Notations

2.1.2. N-gram Language Models (LM)

A language model (LM) s a a model that has assigned probabilities to a sequence of words to it. The gram indicates the amount of words the phrase contains. For example a bigram (2-gram) could be represented by *"survey passed"*, whereas a trigram (3-gram) represents the same with an additional word, like *"survey passed after"*. The N indicates the amount of words in the sequence. For showing the probability of random variable X_i taking in the value $P(X_i = \text{"}survey\text{"})$, it is easier to just write P(survey). For the notation of representing the sequences of N words, I use the notation of Dan Jurafsky and James H. Martin [Jura 19]. N is represented as $(w_1 ... w_n)$ or equal as w_1^n. In this way the expression w_1^{n-1} is the same as $(w_1, w_2, ..., w_n)$. The joint probability for every word in the sequence for the value $P(X = w_1, Y = w_2, Z = w_3, ...W = w_n - 1)$ will be denoted as $P(w_1, w_2, ..., w_n)$. This probability can be decomposed into the **chain rule of probability**:

$$\begin{aligned} P(X_1 ... X_n) &= P(X_1)P(X_2|X_1)P(X_3|X_1^2)...P(X_n|X_1^{n-1}) \\ &= \prod_{k=1}^{n} P(X_k|X_1^{k-1}) \end{aligned} \tag{2.1}$$

Applying the chain rule to the words, the formula changes into [Jura 19]

$$\begin{aligned} P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)...P(w_n|w_1^{n-1}) \\ &= \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \end{aligned} \tag{2.2}$$

With this equation one word is estimated by all prior words of the sentence. This approach is not really good, because language can be used in various new and creative ways and calculating all possible sentences for *N-grams* is inefficient. The better approach is the **bigram** model. It computes the probability based on the one prior word with the following approximation:

$$P(w_n|w_1^n - 1) \approx P(w_n|w_n - 1) \tag{2.3}$$

This assumption that the probability only depends on the prior word is known as the **Markov** [A.A. Markov] assumption. This assumption can be extended to *N-grams*. The general equation for the conditional probability of the next word in the sequence is

$$P(w_n|w_1^n - 1) \approx P(w_n|w_n - N + 1^{n-1}) \tag{2.4}$$

Given the bigram assumption for the probability of an individual word, we can compute the probability of a complete word sequence by substituting Eq. 2.3 into Eq. 2.2 [Jura 19]:

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_k - 1) \tag{2.5}$$

2.1.3. Maximum Likelihood Estimation

MLE

2.2. History of NLP and NTG

Zeitabfolge der geschichtlichen Hintergründe von NTG

2.3. Current trends in technology

Neuronales Ende-zu-Ende Grammerly DeepL Summarizing

2.3.1. Application areas of NLP systems

Image-to-Text, Weatherforecast

2.3.2. Application areas of NTG systems

Speech Recognition, Machine Translation

Chapter 3.

Prototype

```python
x = 1
if x == 1:
    # indented four spaces
    print("x is 1.")
```

Listing 3.1: This is an example of inline listing

You can also include listings from a file directly:

```python
x = 1
if x == 1:
    # indented four spaces
    print("x is 1.")
```

Listing 3.2: This is an example of included listing

3.1. Objective

Image Captioning

3.2. Technical concept

Fachkonzept

3.2.1. Structure

The different steps of Text Generation

- Importing Dependencies
- Loading the Data
- Creating Character/Word mappings
- Data Preprocessing
- Modelling
- Generating text

3.2.2. Neuronal Net

LSTM

RNN

Experimenting with different models

- A more trained model
- A deeper model
- A wider model
- A gigantic model

3.2.3. Process Modeling

Funktionen etc.

3.2.4. Data flow modelling

Diagramm

3.3. Implementation

Code for the Machine Translating

3.4. Evaluation

Print Ergebnisse

Bild

Image Caption

Chapter 4.

Generation of transferable knowledge

Modular expandability of my project. Classification in social context

Appendix A.

Supplemental Information

List of Figures

List of Tables

List of Listings

Bibliography

[Jeka 17] O. D. Jekaterina Novikova and V. Rieser. “The E2E Dataset: New Challenges For End-to-End Generation”. 2017.

[Jura 19] D. Jurafsky and J. H. Martin. “Speech and Language Processing (3rd ed. draft)”. *Stanford University*, 10 2019.

[Kriz] A. Krizhevsky, V. Nair, and G. Hinton. “CIFAR-10 (Canadian Institute for Advanced Research)”.

[Löh 19] T. Löhr and T. Bohnstedt. “Image Classification on the CIFAR10 Dataset”. 2019.

[McCu 43] W. S. McCulloch and W. Pitts. “A logical calculus of the ideas immanent in nervous activity”. *The bulletin of mathematical biophysics*, Vol. 5, No. 4, pp. 115–133, 1943.

[Sing 17] P. Singh. “Neuron explained using simple algebra”. 2017.

[Team 17] A. S. Team. “Hey Siri: An On-device DNN-powered Voice Trigger for Apple’s Personal Assistant”. 10 2017.

[Team 18a] A. S. Team. “Finding Local Destinations with Siri’s Regionally Specific Language Models for Speech Recognition”. 08 2018.

[Team 18b] A. S. Team. “Personalized Hey Siri”. 04 2018.

[Xie 17] Z. Xie. “Neural Text Generation: A Practical Guide”. 2017.

[Yann 98] L. B. Yann LeCun, Patrick Haffner and Y. Bengio. “Object Recognition with Gradient-Based Learning”. 1998.