Diktierfunktion mit Satzzeichen

Eine Jugendforscht Projekt von Antonios Schinarakis und Jonathan Gärtner

1.Einleitung

1.2 Problem

2. Lösungsideen

3. Techniken

3.1. EVA

3.2 Spracherkennungen

3.3 NN

4. Exkurs: Funktionsweise neuronaler Netze

5. Ergebnisse zu Spracherkennung

6. Verlauf des Projekts

7. Quellen

1. Einleitung:

Wir kamen auf die Idee, weil wir regelmäßig die Diktierfunktion unserer Geräte benutzen und es uns genervt hat das wir immer die Satzzeichen mitdiktieren müssen die meistens als Wörter geschrieben werden, deswegen haben wir überlegt, dass das doch einfacher gehen müsste deshalb ist unser Projekt eine Spracherkennung zu bauen, die die Satzzeichen automatisch einfügt. So etwas wurde auch noch nicht oft versucht nur Google hat eine sehr frühe Alpha.

1.2 Problem:

Unser Projekt fingen wir an, weil wir Probleme in den heutigen Diktierprogrammen entdeckt hatten, welches uns ziemlich gestört hatte. Oftmals war es so, dass die Satzzeichen nicht ergänzt wurden und so der diktierte Satz einen ganz anderen Sinn hatten. Diese Zeichen musste man dann selbst ergänzen oder mit diktieren. Wenn man diese mit diktiert hat, wurden sie meistens auch als Wörter geschrieben und nicht als die Satzzeichen selbst. Das hat dann oft Probleme erzeugt, zum Beispiel bei der Spracherkennung oder bei dem Verschicken von Nachrichten.

3. EVA und neuronale Netze:

Hier einige Erklärungen zu den Programmierstrukturen, die wir verwenden

3.1 EVA-Prinzip:

EVA – Eingabe, Verarbeitung, Ausgabe ist praktisch die grundlegendste Programmierstruktur, denn ein Programm ohne Verarbeitung ist kein Programm genauso wie ein Programm ohne Ausgabe etwas bringt oder ein Programm ohne Eingabe immer dieselbe Ausgabe erzeugt. Deswegen braucht man in jedem Programm das EVA-Prinzip.

3.2 Neuronale Netze:

Ein neuronales Netz ist ein Programm das versucht unser Gehirn zu simulieren, d.h. nicht, dass es eine KI ist, sondern das es von selbst lernt, allerdings braucht es dafür sehr große Datenmengen. Das tolle daran ist das die neuronalen Netze so Problem lösen zu denen ein normales „dumm, lineares“ Programm nicht fähig gewesen wäre. Wenn ein neuronales Netz trainiert ist, d.h. seine Aufgabe erfüllt ist es für uns wie ein EVA-Prinzip wobei die Verarbeitung, allerdings eine Blackbox ist. Es gibt verschiedene Arten von NNs und es werden immer wieder neue Entwickelt beispielsweise gibt es neuronale Netze die speziell darauf ausgelegt sind etwas in Bildern zu erkennen und andere die darauf spezialisiert sind Sprache zu erkennen und in Text umzuwandeln. Genaueres hierzu in Kapitel 4

4. Exkurs: Neuronale Netze

5. Ergebnisse zu Spracherkennungen:

Wofür sind große Internetfirmen wie z.B. Google bekannt? Dafür das alle Daten die einmal dort landen nicht mehr wegkommen. Am Anfang unseres Projektes war uns das zwar bewusst, aber erst im Laufe des Projektes wurde es uns wichtiger nicht mehr darauf angewiesen zu seien. In unserem Praktikum habe wir dann das Voxforge Modell gefunden das mit CMU Sphinx funktioniert, allerdings funktioniert es verglichen mit Google ziemlich schlecht. Deshalb wollen wir jetzt ein Modell der Uni Hamburg verwenden, welches aber nicht mit Sphinx sondern mit kaldi läuft was deutlich schwieriger zu verwenden ist.

6. Entwicklung des Projekts:

6.1 Ähnliche Projekte:

Am Anfang haben wir geschaut ob es schon ähnliche Projekte wie das unsere gibt. Nach einiger Suche haben wir festgestellt das es offiziell nur Google etwas derartiges als frühe Alpha anbietet fürs englische anbietet. Es gibt auch andere Projekte auf beispielsweise GitHub die, allerdings nicht für die Kopplung mit einer Spracherkennung gedacht sind. Außerdem bestehen diese Projekte nur aus einfachen seq2seq Training, weshalb die Genauigkeit relativ niedrig ist (je nach Komplexität des Satzes) und es gibt keine gute Möglichkeit in diesem Fall dieses Ergebnis zu Verbessern.

6.2 Warum die Punktierung bei einer Spracherkennung einfacher ist:

In diesem Fall hat man zwei Eingaben:

1. Text:

Hier wandeln wir mit einer bereits vorhanden Spracherkennung (in unserem Fall Sphinx) die Sprache in Text um STT und können den Text analysieren

2. Ton:

Wir können den Ton direkt analysieren und so beispielsweise die Pause zwischen Worten und Sätzen erkennen.

Wir wollen beides Nutzen, den Text hauptsätzlich durch machine learning und die Sprache durch mathematische Operationen. Wenn wir beides Zusammenbringen können wir die Genauigkeit hoffentlich deutlich verbessern.

6.2 Spracherkennungen:

Die bekanntesten Spracherkennungen sind wohl die von Apple und Google, aber es gibt auch viele andere z.B. Microsoft, Amazon und IBM, allerdings werden die Daten hier größtenteils an einen Server gesendet und es ist oftmals nicht klar was mit den Daten passiert (Außerdem kann man die Spracherkennung dann nur online nutzen), ganz im Gegensatz zur open source Spracherkennung CMU Sphinx welche offline funktioniert, sie funktioniert zwar nicht ganz so gut wie Googles, aber aufgrund von der Möglichkeit der offline Nutzung und des Datenschutzes haben wir uns für CMU Sphinx (in Zukunft nur noch Sphinx) entschieden, auch wenn es lange gedauert hat ein Modell zum Laufen zu bringen.

6.3 Praktikum: Um weitere Erfahrungen zu diesem Thema zu machen hat Jonathan mit einem Doktoranten der an seiner Schule eine AG macht geredet, der Vorgeschlagen hat ein Praktikum zu machen in Kooperation mit dem DFKI (deutsches Forschungszentrum für künstliche Intelligenz). Das hielten wir für eine gute Idee die leider nur Jonathan war nehmen konnte da Antoni im Urlaub war und auch nicht angemeldet war. Am letzten Tag als er aus den Ferien kam ist er trotzdem gekommen.

6.3.1: Erster Tag:

Wir hatten einen Raum für uns, will heißen die meiste Zeit Jonathan und zwei seiner Mitschüler die mit open CV gearbeitet haben und konnten mit professioneller Unterstützung unser Programm voranbringen. Außerdem habe Studenten kennengelernt und einiges über die Uni und das Studentenleben gelernt.

6.3.1 Was wir geschafft haben:

Wir haben uns mit pyaudio vertraut gemacht, einer Bibliothek um Audio aufzunehmen und zu analysieren. Des Weiteren haben wir eine Bibliothek namens nltk gefunden mit der wir natrual language processing (zu Deutsch so viel wie natürliche Sprache Verarbeitung) machen können und für uns umso so wichtiger POS (Part of Tagging) damit können wir die einzelnen Worte einer Gruppe zuordnen z.B. Verben, Nomen, Artikel etc. Außerdem haben wir es geschafft eine andere Spracherkennung zu finden.

7.Quellen:

Neuronale Netze programmieren mit Python