Diktierfunktion mit Satzzeichen

Eine Jugendforscht Projekt von Antoni Schinarakis und Jonathan Gärtner

1.Einleitung

1.2 Problem

2. Lösungsideen

3. EVA Neuronale Netze

4. Regeln

5. Verlauf

6. 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.

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. Das hat in vielen Lagen des Lebens nicht weitergeholfen.

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 neuronale 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.

!!Ausführlicher!!

4. Grammatische Regeln:

Um unserem Programm beizubringen wo die Kommas, etc. hingehören, müssen wir erstmal klare Regeln formulieren.

4.1 Satzende:

Als erstes muss unser Programm wissen wo es Satzenden hinsetzen muss, weil es dann den Text nur noch Satzweise analysieren muss.

4.1.1 Satzanalyse:

4.1.2 Sprachanalyse:

Am Ende eines Satzes macht man eine kleine Pause oder geht bei einer Frage mit der Stimme nach oben.

4.2 Nominalisierung:

4.3 Kommata:

4.3.1 Aufzählung:

Nomen (mitbestimmt und unbestimmter Artikeln) oder Adjektive die direkt nebeneinanderstehen und am Ende ein Nomen das mit einem und, oder, beziehungsweise oder sowie getrennt ist. In dem Fall sind außerdem „Nomen“ etc. ähnliche des Weiteren.

(Manchmal sind die Nomen auch mit Adjektiven beschrieben)

4.3.2 Komma vor bestimmten Worten:

Vor den Worten aber, sondern, jedoch

4.3.3 Anreden:

Nach Anreden kommt ein Komma und ein Absatz

4.3.4 Präzisierungen:

Präzisierungen werden durch Kommata vorne und hinten abgetrennt

4.3.5 Nebensätze:

Mit Kommata trennt man neben Sätze voneinander.

4.3.7 Konjunktionen die Satzteile verbinden:

Vor Konjunktionen die Sätze verbinden stehen Kommata.

4.3.8 Gliedsatz:

Manche Gliedsätze werden von Kommata unterteilt

4.3.8.1 Bei folgenden Satztypen:

Gliedsätze die Kausal-, Temporal-, Konditional-, Konzessiv-, Konsekutiv-, Final- oder Modalsätze sind werden durch Kommata getrennt.

4.3.8.2 Fragesatz:

Bei indirekten Fragesätzen werden Gliedsätzen mit Kommas abgetrennt.

4.3.8.3 Relativsätze:

Bei relativen Sätzen werden Gliedsätze mit Kommas abgetrennt.

4.3.8.4

6.Quellen:

Die meisten Grammatiken kannten wir zwar, allerdings wollten wir nochmal exakt alles nachgucken, deswegen haben wir es auf einigen Websites nachgeguckt:

<http://www.udoklinger.de/Deutsch/Grammatik/Kommaregeln.htm>