Leren van regelmatigheden in grote hoeveelheden natuurlijke taaldata

Een grammatica van een taal is een heel nuttig hulpmiddel voor taalkundigen, taalbeheersers en taaltechnologen. Maar het ontwikkelen van een grammatica is een behoorlijke klus. Hiervoor zijn experts nodig die kennis hebben van de taal die beschreven moet worden, van het taalkundig formalisme waarin de taal beschreven wordt, en van de tools die het mogelijk maken de grammatica te schrijven en te evalueren. Het is dan ook interessant om te onderzoeken of deze grammatica's helemaal of gedeeltelijk automatisch gegenereerd zouden kunnen worden op basis van heel grote hoeveelheden natuurlijke taaldata.

Menno van Zaanen
Universiteit van Tilburg

Structuur

Structuur is een belangrijk onderdeel van taal: het bepaalt bijvoorbeeld in welke volgorde we letters of morfemen tot woorden kunnen vormen of hoe we woorden kunnen samenvoegen tot frasen of zinnen. Grammatica's worden gebruikt voor het beschrijven van deze structuren en kunnen gebruikt worden om de structuur van woorden en zinnen door een automatische analyse te achterhalen. Hoe een analyse er precies uit ziet is onder meer afhankelijk van de achterliggende taalkundige theorie en het formalisme op basis waarvan de grammatica is geschreven. De resulterende beschrijving van de structuur van een sequentie kan voor veel taken gebruikt worden. De structuur van woorden bijvoorbeeld is soms nodig om te bepalen hoe een woord uitgesproken moet worden en de structuur van een zin kan helpen om de betekenis ervan te achterhalen.

Inductie

Grammatica-inductie (GI), ook wel grammatica- of grammaticale inferentie genoemd, is een alternatief voor het lastige en tijdrovende handmatige ontwikkelen van grammatica's door experts. GI probeert het ontwikkelen van een grammatica op een andere manier aan te pakken: het onderzoekt de (on)mogelijkheid van het efficiënt automatisch leren van talen en de daarbij behorende grammatica's. Het onderzoeksgebied van GI kan verdeeld worden in twee delen: formele en empirische GI.

Formele GI onderzoekt de leerbaarheid van klassen van talen (zoals contextvrije talen) op een mathematische manier. De resultaten zijn wiskundige bewijzen die laten zien of formele talen uit een specifieke klasse wel of niet efficiënt (in de betekenis van tijd complexiteit) leerbaar zijn.

Het uitgangspunt bij empirische GI is dat natuurlijke talen efficiënt leerbaar zijn (want mensen kunnen dat). Het onderzoek richt zich dan ook op het ontwikkelen van implementeerbare systemen die structuur (delen van een grammatica) zoeken in ongeannoteerde voorbeeldsequenties. De geleerde structuur zou overeen moeten komen met de structuur die taalkundigen aan de sequenties toe zouden kennen.

Geannoteerde data

Bij het ontwikkelen van empirische GI systemen willen we natuurlijk weten hoe goed ze zijn. Er zijn verschillende manieren om de evaluatie uit te voeren, maar de meest gebruikte aanpak is om de uitvoer van een systeem te vergelijken met een gouden standaard, een door taalkundigen geannoteerde dataset. Op basis hiervan kunnen maten van precisie en compleetheid berekend kunnen worden. Helaas zijn deze geannoteerde datasets beperkt in omvang en beschikbaarheid, omdat het

annoteren van taalkundige data tijdrovend is en gedaan moet worden door experts op het gebied van de gebruikte taalkundige theorie en formalisme.

De huidige generatie empirische GI-systemen (ABL (Alignment-Based Learning), ADIOS, CCM-DMV, EMILE, en U-DOP) zijn relatief 'duur' in rekenkracht. Omdat daarnaast de evaluatie plaats vindt op geannoteerde datasets van beperkte omvang, concentreren de meeste systemen zich op het leren van structuur vanuit een beperkte hoeveelheid data. Deze systemen proberen dan ook zo snel mogelijk structuur te leren. Hierbij wordt mogelijk structuur geïntroduceerd waarvan het systeem nog onzeker is. Immers, om structuur uit kleine hoeveelheden data te kunnen leren, zullen generalisaties op basis van kleine hoeveelheden bewijsmateriaal gemaakt moeten worden.
Het leren uit weinig data heeft nadelen. Natuurlijke taal volgt namelijk voor een groot gedeelte de wet van Zipf: er zijn maar een paar woorden of constructies die heel vaak voorkomen, terwijl er heel veel zijn die maar zelden voorkomen. Dit wordt ook wel de 'long tail' genoemd. Deze eigenschap heeft directe invloed op de werking van GI-systemen: er is slechts weinig data nodig om informatie te verzamelen over woorden die vaak voorkomen, maar er is heel veel data nodig om ook eigenschappen van woorden die zelden voorkomen te kunnen leren.

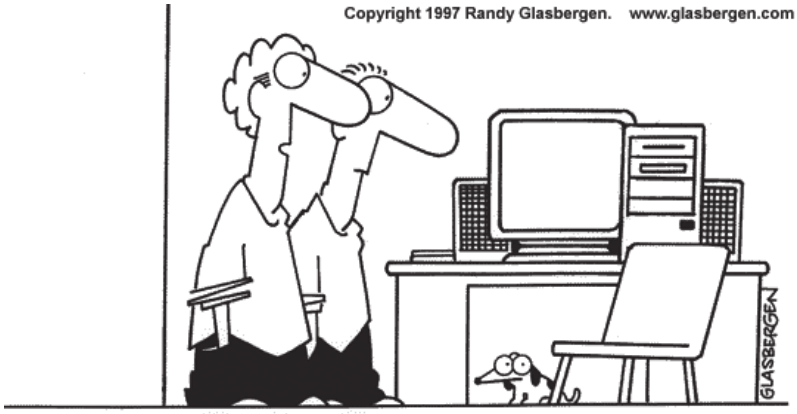
Ongeannoteerde data

Er is een simpele oplossing voor de problemen met het gebruik van beperkte hoeveelheden data: gebruik meer data. Empirische GI-systemen kunnen dan structuur leren op basis van grote hoeveelheden ongeannoteerde data, waarna met de geleerde kennis een kleine hoeveelheid data geannoteerd wordt voor evaluatie. Hierdoor is het mogelijk de systemen te evalueren zonder dat daar grote hoeveelheden handmatig geannoteerde data voor nodig zijn.

De vraag is nu of deze grote hoeveelheden data wel beschikbaar zijn. Op dit moment komen op verschillende plaatsen grote hoeveelheden teksten elektronisch beschikbaar. Denk hierbij aan het scannen en OCRen van bestaande papieren documenten, zoals kranten of boeken, maar ook sociale media genereren veel data. Op Twitter, een microblog-applicatie, worden per uur miljoenen tweets (korte tekstberichten) verstuurd. Deze zijn voor een groot deel te downloaden.
Het leren van structuur in grote hoeveelheden data levert wel een aantal praktische problemen op: de data moet ergens opgeslagen worden. Hiervoor is veel opslagruimte nodig, die bovendien efficiënt toegankelijk moet zijn en die (vanwege repliceerbaarheid van experimenten) gearchiveerd moet kunnen worden. Om de data te kunnen kopiëren of verplaatsen zijn ook snelle verbindingen tussen computers nodig.

Gedistribueerde systemen

Empirische GI-systemen bouwen intern een model van de taal op. Bij grotere hoeveelheden data zal het model complexer worden. Dit vraagt om veel grotere intern geheugens, maar ook om snellere computers. Een mogelijke richting om dit probleem op te lossen is om gedistribueerde systemen te gebruiken. Het probleem van het leren van structuur wordt dan in kleine problemen opgesplitst, dat elk afzonderlijk, op verschillende computers, aangepakt kan worden.



"This computer is very fast! It became outdated faster than any computer I've ever owned."

Het ontwikkelen van gedistribueerde systemen vereist kennis van de technische (on)mogelijkheden van de hardware, maar ook van gespecialiseerde programmeertalen of technieken om bestaande systemen gedistribueerd te maken. Niet alle onderzoekers zullen deze gespecialiseerde kennis hebben. In de (nabije) toekomst zal hier dus nog heel wat werk verzet moeten worden.