Självskrivande mjukvara

Annie Tallund
Lunds Tekniska Högskola
Email: an0284ta-s@student.lu.se

Inger Gundersen
Lunds Tekniska Högskola
Email: in3530gu-s@student.lu.se

***Sammanfattning*—I vårt Whitepaper beskriver vi inledningsvis två sorters självskrivande mjukvara: Lsjbot (som producerar Wikipedia-artiklar) och Wordsmith (som sammanfattar statistik av olika slag). Den förstnämnda är ett opensource-projekt från tidigt 2010-tal som år 2014 hade producerat 2 700 000 artiklar på sidan. Lsjbot baseras på öppna databaser och utgår från färdiga klassificeringar och bilder för att beskriva bland annat geografiska platser och insektsarter. Till sin hjälp har den färdiga textsegment som utgör själva språkskelettet. Wordsmith fungerar på ett liknande vis men datan tillhandahålles av kunden.**

De två botarna är enligt våra slutsatser två tidiga exempel på artificiell intelligens inom språkteknologi. Utöver dagens funktionalitet ser vi hur algoritmerna skulle kunna utökas med hjälp av maskininlärning. I och med det borde framtiden kunna erbjuda långt mer avancerade botar som baserat på bilder, filmer och kanske data med lösare klassificering kan beskriva fenomen i vårt samhälle. Tekniken medför dock en del risker. Till exempel svårigheter att avgöra vad som är skrivet av en riktig person och inte.

I. Inledning

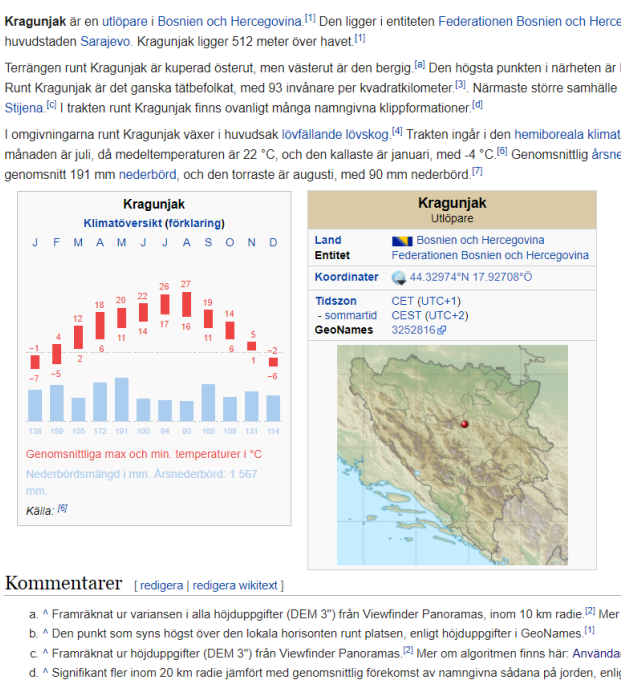
2010-talets människor är vana vid lättillgänglig information om alla tänkbara ämnen. Vår nya livsstil är däremot inte helt oproblematisk. Många olika sorters statistik (exempelvis aktier och sportresultat) förnyas ständigt. Viss fakta kan ha en målgrupp av oansenlig storlek. Det är ett nästintill oändligt projekt för oss människor att försöka skriva ned allt. Dessutom är viss kunskap så specifik och temporär att nyttan (antalet läsare) med den inte överskuggar ansträngningen som krävs för att sammafatta den. Här kommer självskrivande mjukvara in. Det är en relativt ny gren inom artificiell intelligens. Att låta datorer sammanfatta texter med utgångspunkt i data gör det bekvämt för oss att på ett effektivt sätt ta del av information. Genom att undersöka hur självskrivande mjukvara arbetar får vi en inblick i framtidens informationsflöden.

II. Bakgrund och relaterad forskning

A. Lsjbot

I vårt projekt har vi i första hand tittat på en bot som opererar på den nätbaserade encyklopedin Wikipedia. Den är utvecklad av Sverker Johansson och kan författa 10 000 artiklar om dagen. Lsjbot står för 8.5 % [2] av allt innehåll på Wikipedia, samt majoriteten av texterna på de svenska och filippinska versionerna. Ett exempel på en artikel som är skriven av Lsjbot illustreras i figur 1.

Koden och författandet av texterna bygger på färdiga textbitar där data från olika källor stoppas in på lämpliga ställen (vilket illustreras i figur1). Artdatabanken är en av flera större databaser som Lsjbot använder [5]. Varje art kategoriseras efter ett antal faktorer. Sortering kan göras baserat på bland annat geografisk utbredning, artklassificering, utrotningshot eller risker, såsom kraftig ökning eller minskning i antalet individer. Vid sökning visas egenskaperna Vetenskapligt namn, Svenskt namn, (toxologisk) Typ, Organismgrupp, Kategori (nivå av livskraftighet) samt vilken Landskapstyp den förekommer i. Det är det senare uppräknade som Sverker, i fallet med arter, baserat sin nedladdning på. Exempel på textsegment återfinnes i figur 2, vilket för övrigt är Sverkers eget dokument som används till att generera text i artiklar. Eftersom andra källor är uppbyggda på andra vis krävs det anpassningar i koden för varje typ av ämnen [5]. Lsjbot har skrivit artiklar om geografiska platser och då använt sig av exempelvis NASA-källor som innehåller väderdata.



Figur 1. Exempel på Wikipedia-artikel framställd av självskrivande mjukvara

Boten har också funktionalitet som, i fallet med sjöar, öar och berg, kan analysera bilder och beskriva ett objekt utifrån dess utseende. Delar av datan utgörs då av en bild och höjduppgifter som tillhör denna. Då ett bergs storlek ska avgöras utgår Lsjbot från den pixel där koordinater för platsen är. Rekursiva algorit-

mer tillämpas sedan tills dess att ett visst villkor inte längre är uppfyllt för pixeln som undersöks. Villkoret ges exempelvis för sjöar av att höjden ska vara lika med ursprungspixelns höjd, det vill säga sjöns yta som har samma höjd överallt. [5].

Sverker har själv kommenterat att ett krav för att en databas ska kunna användas för självskrivande mjukvara är att användarvillkoren måste tillåta att databasen laddas ned av vem som helst. Detta återfinns exempelvis i databasen GeoNames, där datan uppges vara fri för alla att ladda ned så länge GeoNames erkänns i samband med texten. Det blir enkelt att källhänvisa till dem på Wikipedia då detta är ett centralt inslag i konceptet (i och med att sidan inte betraktas som någon förstahandskälla).

```
117     The area is sparsely populated  Trakten är glest befolkad
lumulupyo
118     The area is densely populated   Trakten är tätbefolkad  Ma
119     The area is very densely populated      Trakten är mycket
ang lumulupyo
120     {{GeoGroup}}    {{GeoLänk}}     {{GeoGroup}}    {{GeoGroup
121     to the north    norrut  sa amihanan
122     to the south    söderut sa habagatan
123     to the west     västerut        sa kasadpan
124     to the east     österut sa sidlakan
125     to the northeast        åt nordost      sa amihang-sidlaka
126     to the southeast        åt sydost       sa habagatang-sidl
127     to the northwest        åt nordväst     sa amihang-kasadpa
128     to the southwest        åt sydväst      sa habagatang-kasa
129     The area of #1 is #2 square kilometers. Arean är #2 kvadra
```

Figur 2. Textexempel som artiklar baseras på (engelska, svenska, filippinska)

B. Wordsmith

En annan typ av självskrivande mjukvara som vi har kollat på är API'et Wordsmith, vilket är skapat av Automated Insights. Flera stora företag som bland annat Microsoft, Yahoo och PwC använder sig av detta applikationsgränssnitt som kan generera mer än 1,5 miljarder artiklar på ett år. [1] Wordsmith är ett såkallat *natural language generation*-verktyg som översätter data till sammanhängande text. En styrka som Wordsmith har är att den kan anpassas efter kundens behov. Den kan till exempel skriva ekonomiska artiklar baserat på tabeller och siffror för ett företag, medan den sammanfattar matcher och skriver rapporter för spel så som fantasy football för ett annat företag.

För att Wordsmith ska kunna generera text måste den ha tillgång till data som är strukturerad på ett visst sätt. Exempler på detta är databaser eller filer på CSV-format (Comma-separated values). Datan matas in till gränssnittet som använder färdiga mallar för att strukturera informationen och generera meningar. Enligt Automated Insights är gränssnittet enkelt att lära sig och användarna har stor frihet när det gäller att modifiera mallerna efter sina egna behov. De hävdar därför att gränssnittet är kraftfullt nog till att varje text som genereras är helt unik [1].

C. Maskininlärning

Avslutningsvis vill vi beröra området maskininlärning, för att kunna diskutera hur dagens teknik kan utvecklas i framtiden. Det finns ett antal olika sorters maskininlärning. De alla har gemensamt att koden på ett eller annat sätt är självlärande, det vill säga att den kan börja dra egna slutsatser kring data. Det kan ske genom *Random Forest*, en metod som med hjälp av logiska träd där varje nog är en fråga om ett specifikt fall, exempelvis "Är inkomsten högre än 70 000 kr?". På så vis når vi fram till olika fall genom komplex utsortering. I den här metoden är blackboxing, det vill säga att vi inte riktigt vet hur en algoritm kommit fram till något, ett återkommande problem [4]. Nästa alternativ är *K-Means Clustering*, som enbart genom att matas med data katergoriserar denna utifrån saker som är gemensamma. Två andra metoder är att försöka efterlikna den mänskliga hjärnans metoder för att hantera information, *neural networks*, samt *Logistic Regression*, där man jobbar mot binära värden på vår data. Den sistnämnda brukar klassificeras mot traditionell statistik [4].

III. Frågeställningar

- Hur sker nedladdningen av data och hur genereras text baserat på detta?
- Hur kommer teknologin inom området att se ut i framtiden?
- Vilka risker finns det med självskrivande mjukvara?

IV. metodbeskrivning

För att undersöka detta ämne och besvara våra frågeställningar behövde vi samla in information från ett antal källor inom området. Vi kollade på olika mjukvaror och deras uppbyggnad för att kunna jämföra deras beteende och implementation. Vi fick även analysera olika öppna databaser för att observera hur dessa presenterar sin data. Detta gjordes för att bättre förstå hur inhämtningen och sorteringen av data utförs av mjukvaran.

Dessutom fick vi göra en litteraturanalys för att ta fram de olika åsikterna kring självskrivande mjukvara som finns i den akademiska miljön. Detta för att få en mer nyanserat bild på såväl fördelarna som nackdelarna med användningen av denna typen av mjukvara. Vi kontaktade även Sverker Johansson, skaparen av LSJBot, eftersom det var den boten vi valde att lägga mest fokus på. Han gav oss information om hur nedladdningen av data sker, källkoden till boten samt en ordlista vilken texterna baseras på.

V. Resultat och diskussion

A. Tillgång till data

Sverker Johansson har själv sagt att just insektsarter är lämpligt för en självskrivande bot [5]. Det är sannolikt att han har den tydliga katergoriseringen i åtanke. Vi ser stora fördelar med CSV och strukturer eftersom det är ett format som mjukvara är van att hantera. Vid nedladdning från internet måste vi ha sorterad data, eftersom innehållet annars kan bli meningslöst för ett program som inte vet hur den ska tolka det. Det är möjligt att vi med hjälp av maskininlärning kan få våra program att bli bättre på detta, i och med att

man har kommit långt med sådan teknik inom till exempel grafik. Ett program som matats med ett stort antal bilder på bilar har med neurala nätverk lärt sig en bils kännetecken rent utseendemässigt. På samma vis skulle en självskrivande mjukvara kunna "scanna" internet och göra en uppskattning av en faktauppgifts innehåll, karaktär och pålitlighet.

Organisationen som tillhandahåller en databas måste alltså tillåta att datan laddas ned på det vis som en mjukvara opererar på. Rent juridiskt blir det alltså svårt att tillhandahålla en bot som på begäran kan skriva en artikel om vad som helst. Det kan ses som en begränsning, eftersom informationen som en självskrivande mjukvara har tillgång till i så fall är reducerad till öppna databaser. Detta blir särskilt i termer av Wikipedia där innehållet ju styrs utan användarna, och således bör det skrivas artiklar som är av intresse för läsarbasen.

Å andra sidan kan just det här ses som en styrka med självskrivande mjukvara. I och med att den endast kan basera sitt innehåll på uppstyrda databaser som är öppna för alla, ökar validiteten på automatiskt genererade artiklar. Ett alternativ hade varit att boten själv fick i uppgift att läsa in en text och tolka den, varpå det föreligger stora risker för blackbox-problematik i algoritmerna och tolkning av subjektiva åsikter som fakta. Självklart finns det alltid en risk för att inte ens databaserna är trovärdiga, men det bör med anledning av detta också förekomma en begränsning i mjukvaran gällande vilken information som laddas ned. Tack vare detta undviker vi att en artificiell intelligens helt enkelt läser in "hela internet" och försöker dra slutsatser utifrån detta.

B. Framtida utvecklingar och möjligheter

Ännu en begränsning är att Lsjbot använder sig av färdiga textsegment. Med andra ord är det inte artificiell intelligens i den traditionella bemärkelsen, eftersom koden inte är självlärande. Istället fungerar den mer som en mall med kapaciteten att producera stora mängder material av samma art. Jämfört med exempelvis en chatbot som baseras på maskininlärning torde även utvecklingen för självskrivande mjukvara kunna gå åt det hållet. En möjlighet är alltså att kombinera det neurala nätverkets förståelse för språk med nedladdning av data, vilket kombineras till en mer beskrivande text. Det skulle följdaktligen resultera i att texten upplevdes som "mänskligare".

Även Wordsmith använder sig ut av färdiga mallar när den genererar text. Den stora skillnaden mellan Lsjbot och Wordsmith är dock att Wordsmiths mallar är dynamiska. Eftersom Automated Insight har skapat sin mjukvara för att kunna sälja den till företag behöver de större utrymme för att ändra i mallarna för att på så sätt kunna anpassa verktyget till sina kunder. Lsjbot som bara ska användas för att skriva en typ av text behöver inte denna extra funktionen. Eftersom Automated Insights skyltar med att alla texter som deras mjukvara skapar är unika och att detta beror just på att man kan modifiera de färdiga mallarna och även skapa egna, så är det kanske något inom detta område som kan förbättras för att skapa mer levande texter. Ju mer man som människa kan styra över hur texten byggs upp desto större mänskligt intryck borde texten naturligtvis ge.

C. Risker

Den mest återkommande kritiken angående mjukvaran är att texterna som genereras saknar det mänskliga perspektivet och att texterna därför är tråkiga att läsa [3]. Botarna saknar till exempel funktionalitet för att föra resonemang. Texterna blir av denna anledning också väldigt korta, ofta inte mer än fyra meningar. Detta är defintivt en stor utmaning som måste lösas om man vill använda botar till att skriva något annat än rena fakta-artiklar. Däremot kan man diskutera huruvida någon går in på encyklopedier så som Wikipedia för att läsa artiklar för nöjes skull. Om man endast söker fakta täcker de bot-genererade artiklarna det behovet.

En risk med självskriven mjukvara är att det kan bli svårt att avgöra om en text är skriven av en bot eller en människa. I nuläget är man medveten om man läser en text som är genererat av självskrivande mjukvara, både för att det oftast står i artikeln, men också genom att språket inte är tillräckligt nyanserat. Med bättre teknologi kan det däremot bli svårare att urskilja vilka texter som faktiskt är människo-skrivna, och vilka som inte är det. Detta kan bli problematisk då mjukvaran kan användas för att påverka människor på ett negativt sätt. Det är därför viktigt att vi i takt som teknologien förbättras är medvetna om vilka risker som finns. I framtiden kan det följaktligen behövas riktlinjer för hur självskrivande mjukvara får användas och hur den bör designas.

Dagens teknologi utgör däremot ingen större risk som den är nu, just därför att språket inte är avancerat nog. Dessutom är all text som genereras ut av de typerna av mjukvara som vi har kollat på i denna undersökning fullständigt baserat på fakta från källor. Detta gör det lätt för de som läser texterna att verifiera innehållet, men framförallt gör det det enkelt att hitta andra artiklar inom samma område.

VI. Slutsatser

Mjukvaran som vi har kollat på i denna undersökning använder sig huvudsakligen av färdiga mallar där insamlat data från olika öppna databaser på internet eller kundtillhandahållen data stoppas in för att generera text. Texterna saknar i många aspekter en mänsklig faktor, då teknologin inte är tillräcklig avancerad än. I nuläget grundas all text enbart på fakta, och därför kan texterna upplevas som tråkiga då mjukvaran inte har kapacitet för att uttrycka mänskliga känslor eller föra resonemang än. Däremot ser vi stora möjligheter för utveckling inom detta område. Mer integration med principen inom artificiell intelligens kan göra att det i framtiden kan vara möjligt för mjukvara att generera texter som liknar det en människa kan skapa, och att dessa texter inte behöver vara

enbart baserade på data på en viss form.

Mer avancerad teknologi kan också föra med sig risker. Ju mer mänskliga texter mjukvaran kan producera, desto svårare blir det att urskilja vad som är skrivet av botar och vad som är skrivet av människor. Att detta kan användas för att påverka människor på ett negativt sätt är därför något som vi måste vara medvetna om i fortsättningen. Därför kommer det möjligtvis behövas reguleringar för hur självskrivande mjukvara ska få användas i framtiden.

VII. FORTSATT ARBETE

Vi hade velat gå in i koden och titta på hur algoritmerna är uppbyggda samt testa dessa emot olika sorters data för att få större insikter i hur en självskrivande mjukvara faktiskt fungerar. Vi hade dessutom velat titta på maskininlärningbaserade algoritmer för att under möjligheterna att kombinera dessa två tekniker enligt det resonemang som förs i diskussionen. En ytterligare utveckling av undersökningen, som har en tätare koppling till datorkommunikation, hade varit att titta på vilka paket som skickas och tas emot när data laddas ned från internet.

REFERENSER

[1] Automated Insights, Managed Services, webbartikel, 2017. Hämtad 2017-12-06 [https://automatedinsights.com/managed-services]

[2] E Emmerentze, For This Author, 10,000 Wikipedia Articles Is a Good Day's Work, artikel i Wall Street Journal (online), 2014. Hämtad 2017-11-25

[3] E Emmerentze, The Wikipedia hero and his bot, artikel i Wall Street Journal (online), 2014. Hämtad 2017-12-02 [https://www.wsj.com/articles/for-this-author-10-000-wikipedia-articles-is-a-good-days-work-1405305001]

[4] N. Castle, An Introduction to Machine Learning, webbartikel, 2017. Hämtad 2017-12-08 [https://www.datascience.com/blog/introduction-to-machine-learning-algorithms]

[5] Privat kommunikation med Sverker Johansson, email, 2017.