

Bagsiden ved at træne algoritmer til machine learning: Når datasættene bliver meget store, kræver det enorm beregningskraft og dermed et stort elforbrug. Denne problemstilling er en væsentlig drivkraft for udviklingen af mere smarte algoritmer. Foto: Shutterstock



FREMTIDENS KUNSTIGE INTELLIGENS

kræver smarte algoritmer

Om forfatteren
Af Kristian Sjøgren,
videnskabsjournalist.
ksjoegren@gmail.com

Om forskeren



Kasper Green Larsen er ph.d. og nu professor i datalogi ved Aarhus Universitet. Kasper's forskning er indenfor teoretisk datalogi med særligt fokus på algoritmer, datastrukturer og machine learning. larsen@cs.au.dk

Aktuelt er Kasper Green Larsen leder af et Sapere Aude Research Leader Grant fra Danmarks Frie Forskningsråd. Projektet er baseret på at anvende tekniker udviklet indenfor datastrukturer til at forbedre machine learning og kryptografiske værktøjer.

Udviklingen af computere er sket i et tæt samspil mellem udvikling af hardware og software. Fremtidens udfordring bliver i høj grad at forbedre de algoritmer, man bruger til machine learning, hvis ikke udgifter til computernes strømforbrug skal stikke helt af.

Der var engang, hvor computere, der efter nutidens målestok vil blive betegnet som udpræget primitive, fyldte hele gymnastiksale og alligevel kun kunne foretage forholdsvis simple beregninger. I dag render de fleste af os rundt med mange gange mere computerkraft i lommen, end der som eksempel skulle bruges til at sende det første rumfartøj til Månen. I en iPhone 13 er der 75.000 gange mere regnekraft end i Apollo 13.

Den digitale revolution er i høj grad båret af teknologiske landvindinger, der har gjort computerchips, processorer og lagringskapacitet henholdsvis mindre, hurtigere og større. Via vores telefoner har vi på

få sekunder adgang til milliarder af billeder, musiknumre og informationer. Vores biler kan næsten køre selv, og alt fra kirurger til lagermedarbejdere er assisteret af intelligente robotter, som letter deres arbejde. Det er data, data og atter data.

Alt sammen kan dog ikke alene krediteres hardwaredelen af den teknologiske udvikling. Den anden halvdel er alt det, som foregår inde i hardwaren. Algoritmer og kunstig intelligens jonglerer i det skjulte med enorme mængder data, for at du så let som ingen-ting kan finde fra København til Eiffeltårnet, finde billeder af din ekskæreste på internettet eller finde frem til kunstneren og titlen

bag et musiknummer, som du hørte i radioen.

En af de forskere, som er med til at udvikle morgendagens smarte algoritmer, er professor Kasper Green Larsen fra Institut for Datalogi ved Aarhus Universitet. Sammen med kollegaen Kaj Grønbæk holdt han for nylig i forbindelse med foredragsserien Offentlige Foredrag i Naturvidenskab et oplæg om, hvordan vi er kommet fra kuglerammer og kampen om at lave beregninger til at sende den første mand til månen til smartphones og Googlesøgninger efter nøgenfotos af kendisser.

»Hardware og de bagvedliggende algoritmer er fælles om den van-



vittige udvikling, som har fundet sted de seneste årtier. Selvom de fleste måske mest tænker på hardwaren, for eksempel størrelsen på en smartphone, som det centrale i den teknologiske udvikling, er de bagvedliggende algoritmer endnu vigtigere end selve hastigheden på de processorer, der skal køre algoritmerne. Man kan godt lave hurtige beregninger med en hurtig algoritme på en langsom processor som den, der sad i computeren ved månelandingen, men man kan ikke lave hurtige beregninger på en hurtig computer med en langsom algoritme,« siger Kasper Green Larsen.

Algoritmer er ikke bare algoritmer

En algoritme er meget simpelt en opskrift på at løse et problem. En algoritme kan være opskriften på noget så simpelt som at lægge to store tal sammen. Så plusser man ét ciffer med et andet ciffer, har noget i mente, lægger det til lidt senere og så fremdeles. Til sidst kommer man frem til resultatet. Måden, hvorpå man kommer frem til resultatet, er en algoritme – altså opskriften. Det samme gælder, hvis man vil gange to tal med hinanden. Der skal man bruge en anden algoritme.

På samme måde findes der algoritmer, som kan filtrere i store mængder data. For eksempel kan man forestille sig en algoritme, som er i stand til at finde en person i telefonbogen. Lad os sige, at den person hedder Peter Hansen.

To forskellige søgealgoritmer



Bog med 1.000 sider

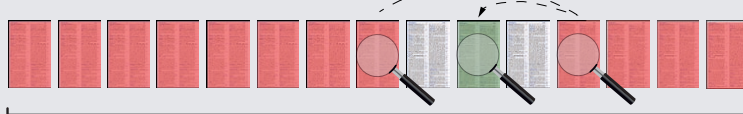
Figuren viser forskellen på effektiviteten af to forskellige søgealgoritmer, lineær og binær søgning, hvis man vil lede efter et bestemt ord i en bog på 1000 sider, hvor ordene står i alfabetisk rækkefølge.

Lineær søgning

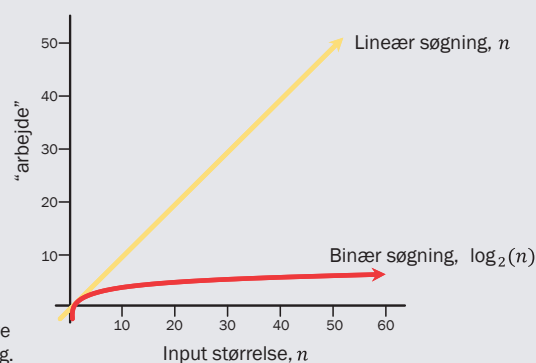


Vi udelukker 1 side ad gangen → 1.000 opslag
Antal opslag hvis n sider → n

Binær søgning



Vi udelukker halvdelen af siderne hver gang.
1.000, 500, ..., 4, 2, 1 sider tilbage → 10 opslag
Antal opslag hvis n sider → maksimalt $\log_2(n)$



Grafen viser, at der er meget stor forskel på n og $\log_2(n)$, når n bliver stor – binær søgning er med andre ord meget mere effektiv end lineær søgning.

Programmerer kan så lave en algoritme, som kigger hele telefonbogen igennem, indtil den har fundet et navn, som matcher alle bogstaverne i P-E-T-E-R H-A-N-S-E-N. Simpelt!

Det med at søge efter Peter Hansen i ordbogen illustrerer også meget godt, at algoritmer kan være mere eller mindre effektive og dermed også langsomme eller hurtige. Én algoritme kan som eksempel begynde fra A og ellers bare fortsætte gennem telefonbogen, indtil den støder på Peter Hansen.

En anden mulighed er at lave en algoritme, der starter med at slå op i midten af telefonbogen. Her kan algoritmen ud fra den alfabetiske rækkefølge så afgøre, om Peter

Hansen er længere fremme i telefonbogen eller længere bagud. På den måde kan algoritmen med det samme udelukke, at Peter Hansen står på halvdelen af siderne i telefonbogen, og at den derfor ikke behøver at lede efter ham der. Derefter gør den det samme igen og igen, indtil den til sidst har fundet frem til den person, som den gerne vil finde. Det går også meget hurtigere end at skulle hele vejen gennem telefonbogen fra den ene ende til den anden.

»Derved kan den ene algoritme være hurtigere end den anden i forhold til at løse opgaven, der her drejer sig om at søge i data, som står i alfabetisk rækkefølge. Taler vi om en algoritme, der som ek-

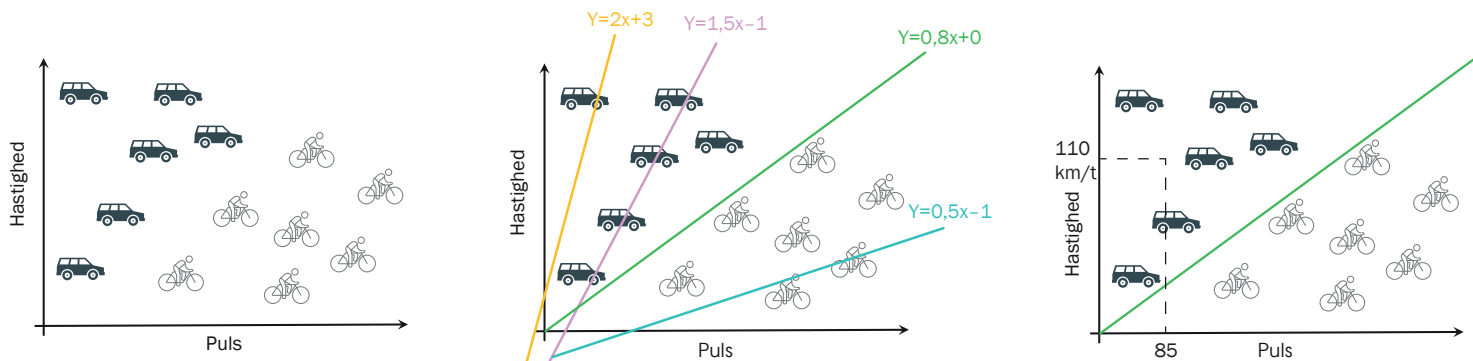
Input: Data med sammenhørende værdier for puls og hastighed fra personer der enten har kørt i bil eller på cykel.

Computeren finder den bedste skillelinje ved, at en algoritme afprøver en masse forskellige værdier af a og b i ligningen $y=ax+b$



Puls: 85
Hastighed: 110 km/t

eller ?



Eksempel på machine learning, som træner en algoritme til at afgøre, om man cykler eller kører i bil ud fra en måling af puls og den hastighed, man bevæger sig med.

Machine learning løser fremtidens problemer

Machine learning er sat på opgaven at løse nogle af de problemer, som forskellige dele af samfundet står over for nu og i fremtiden.

Et eksempel er selvkørende biler. Før bilerne kan slippes løs på vejene, er det dog nødvendigt, at vi kan være sikre på, at de algoritmer, der styrer bilerne gennem trafikken, træffer de rigtige beslutninger undervejs – hver eneste gang. Drejer det sig om at køre ligeud ad en motorvej med velafmærkede vejstriber, kan det allerede i dag lade sig gøre, men problemet bliver straks større, når det kommer til for eksempel et højresving, og hvor det er vigtigt at kunne aflæse fodgængere og cyklisters intentioner.

Tesla arbejder allerede i dag på at udvikle de algoritmer, der skal gøre selvkørende biler til en naturlighed på alle verdens veje om fem eller ti år. Det gør bilproducenten blandt andet ved at opsamle data fra kameraer rundt om på de mange Teslaer, der kører rundt i blandt andet Danmark. Bilerne optager, hvad der sker i omgivelserne, og hvad chaufførerne gør, og sender data tilbage til store Tesla-ejede datacentre, hvor machine learning gnasker sig igennem den enorme datamængde for at finde ud af, hvad der er det rigtige at gøre i givne situationer, eksempelvis i et kompliceret vejkryds med mange andre trafikanter, fodgængere og cyklister. Jo mere data, Tesla kan indsamle, des bedre bliver de algoritmer, der skal bremse i tide, når en gammel dame ikke får set sig godt nok for, inden hun krydser vejen.

Et andet eksempel på brugen af machine learning er indenfor udvikling af lægemidler. I årtier har forskere forsøgt at finde en model for, hvordan man ud fra et gen kan afgøre, hvordan det protein, som kommer ud af genet, ser ud og bliver foldet. Dette problem omkring foldning af proteiner løste Google med en algoritme, der med data på alle kendte proteiner og deres tilhørende gensekvenser lærte, hvordan den kan forudsige proteinstrukturen ud fra en gensekvens. Denne viden kan blandt andet benyttes indenfor lægemiddeludvikling, hvor man nu meget præcist kan forudsige, hvordan en DNA-sekvens skal se ud for at få et protein med givne medicinske egenskaber.

sempel skal finde en hjemmeside indenfor et splitsekund, er det helt afgørende, at den ikke skal gennemgå alle eksisterende hjemmesider fra en ende af, for det vil simpelthen tage alt for lang tid,« forklarer Kasper Green Larsen.

Machine learning har bragt algoritmer til næste niveau

Over hele verden arbejder forskere og programmører på at udvikle hurtigere algoritmer, der kan løse nogle af de opgaver, som vi som samfund står overfor, hurtigere og mere præcist. Det er dog i mange tilfælde ikke længere nok, at forskere og programmører selv tænker kloge tanker om, hvad algoritmerne skal gøre. Vi er kommet dertil i algoritmernes udvikling, at vi bliver nødt til at bede dem om selv at tænke nogle af de kloge tanker for os. Vi taler i den sammenhæng om machine learning eller kunstig intelligens.

Ved machine learning fodrer vi algoritmer med datasæt, som algoritmerne kan lære ud fra og benytte til at finjustere præcisionen i deres forudsigelser. Man kan betragte det som algoritmer, der selv producerer algoritmer.

Forestil dig som eksempel en simpel algoritme, der skal kunne afgøre, om en person cykler eller kører i bil gennem byen. Algoritmen har nogle få data til rådighed, herunder hastighed og puls.

Når det gælder bykørsel, kan man forestille sig, at det ikke alene ud fra hastigheden

er muligt at sige, om en person kører i bil eller på cykel, men kobler man det til personens puls, vil det intuitivt give mening, at hurtig kørsel og høj puls indikerer, at man kører på cykel, mens hurtig kørsel og lav puls peger på, at man sidder i en bil.

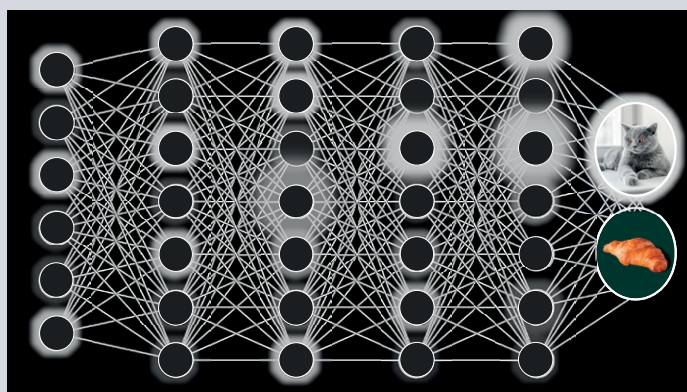
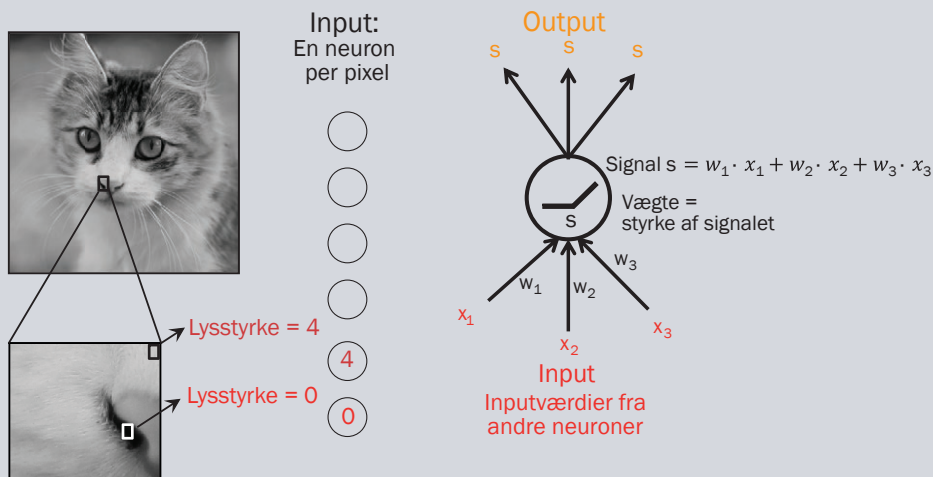
Tanken er så den, at man præsenterer en algoritme for tusindvis af data fra personer, der er cyklet eller har kørt i bil gennem byen. Disse data kan komme fra et smartwatch, og algoritmen får også at vide, hvordan personen har bevæget sig frem i trafikken. En simpel machine learning-algoritme kan så plotte data i et koordinatsystem, hvor hastighed og puls går ud af hver deres akse. Det vil lede til en masse prikker i koordinatsystemet, og disse prikker vil formentlig gruppere sig i to område, der modsvarer det at køre på cykel eller i bil. Her er det så op til en programmør eller algoritmen selv at trække en linje gennem koordinatsystemet, hvor alt på den ene side af linjen svarer til at køre på cykel, mens alt på den anden side af linjen svarer til at køre i bil. Næste gang en person kører gennem byen, og algoritmen bliver præsenteret for puls og hastighed, kan den hurtigt se, om prikken i koordinatsystemet falder på den ene eller anden side af linjen. Den har kigget på data, og den har lært sig selv, hvad der svarer til det ene eller andet.

»Man skal bruge en masse data på at lære algoritmen at lave denne linje og endnu mere data på at lære den at finjustere linjen. I gamle dage satte programmøren selv linjen, men med machine learning bliver det meget mere præcist, og algoritmen kan også finde mønstre i data, som det er umuligt for mennesker at overskue,« siger Kasper Green Larsen.

Kræver mere strøm end Manhattan i en uge

Går vi et skridt op i kompleksitet, kræver billedgenkendelse meget mere databehandling end eksemplet med de to datapunkter for en tur gennem byen. Nogle af de store teknologigiganter udviklede som noget af det første en algoritme, der kan genkende katte på billeder. Algoritmen har 100 milliarder vægte (se boks), den kan skrue på (modsat de 2 for linjen, hvor det kun var

Billedgenkendelse og neurale netværk



Fundamentet i billedgenkendelse er neurale netværk, der efterligner den måde, hjerne-celler (neuroner) er forbundet på i hjernen. Neuronerne i netværket er organiseret i forskellige lag illustreret ved de lodrette søjler. Den information, man starter med i det første lag, vil således blot være værdier for lystyrken af en given pixel i billedet. De enkelte neuroner er forbundet til andre neuroner i de nærliggende lag, således at en neuron får input fra de andre neuroner og sender et signal videre. En matematisk model for en neuron afgør så styrken af det signal, der sendes videre. Ligesom i hjernen kan forbindelsen mellem to neuroner være svag eller stærk (udtrykt ved forskellige vægte "w1-w3"). Og når netværket trænes, går det altså ud på at justere vægtene mellem forbindelserne, således at man får resultatet "kat", når det faktisk er et billede af en kat, man bruger som input.

a og b), for at lære, hvad der kendetegner datamønstret i et billede af en kat, og hvad der kendetegner datamønstret i et billede af noget andet, for eksempel en hund.

Algoritmen bliver som i tilfældet med cyklisterne og bilisterne præsenteret for måske 10 millioner billeder af katte, og når den så har brugt den enorme mængde data til med stor præcision at kunne udpege en kat på et billede, kan den gøre det ud fra helt nye billeder, som den ikke har set før. Her opstår der så også et problem, som alle forskere og udviklere inden

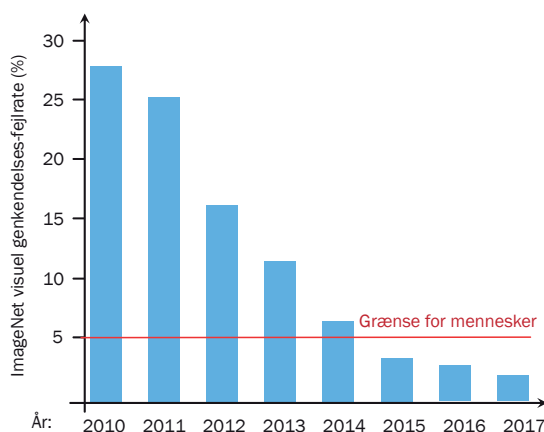
for feltet netop nu forsøger at gøre noget ved.

Skal machine learning lære ud fra to datapunkter, puls og hastighed, skal den ikke bruge mere computerkraft, end hvad der er i en Nokia 3310. Når datasættene til gengæld bliver i milliard-størrelsen med tilsvarende 100 milliarder parametre, kommer træningen af modellen til at kræve enorme mængder beregningskraft. Det kræver foruden enormt kraftige computere også elektricitet, der ved træningen af de mest komplekse modeller kommer op i priser i milliardstørrelsen i elforbrug.

Forsker i selv at gøre algoritmer bedre

Kasper Green Larsens egen forskning går ud på at gøre algoritmer bedre med færre data. En af de mest berømte algoritmer indenfor området hedder Adaboost, der kan forbedre svage machine learning-algoritmer. Adaboost fungerer på den måde, at man lægger den oven i en algoritme, der måske ikke er så god, som man kunne ønske.

Forestil dig en algoritme, som skal genkende en kat på et billede, men kun rammer rigtigt i lidt over 50 % af tilfældene. Adaboost fungerer ved at øge prioriteten af de data, som algoritmen lavede fejl på under træningen, og så køre træningen af algoritmen igen. Det bliver man så ved med, indtil man har trænet modellen en hel masse gange med forskellige udfald. Når algoritmen så tages i brug, laver den en demokratisk afstemning mellem alle udfaldene af de mange træninger og benytter så det, som de fleste peger på. Er det en kat, eller er det ikke en kat? Matematiske beregninger har vist, at denne metode kan gøre selv dårlige algoritmer ret præcise og kan gøre algoritmer på baggrund af få data lige så stærke som algoritmer på baggrund af mange data.



ImageNet er en samling på en million – forholdsvis stiliserede – billeder, som man bruger til at teste forskellige algoritmer til billedgenkendelse. Grafen viser, at allerede tilbage i 2015 var algoritmerne så gode, at de laver færre fejl end mennesker, når de skal vurdere, hvad der er på billederne i denne samling.

»Derfor er det også kun de store spillere som Google og Facebook, der laver og bruger disse avancerede modeller for machine learning. Samtidig bliver det ikke bedre i fremtiden, hvor modellerne formentlig bliver mere komplekse og skal lære ud fra endnu flere data. Nogle beregninger har vist, at hvis vi fortsætter udviklingen, vil det om ti år kræve den samme mængde strøm at træne én model, som man bruger på hele Manhattan i New York på en uge. Vi taler om en eksplosion i udgifter, jo mere præcise man vil have sine modeller, og jo mere de skal kunne,« fortæller Kasper Green Larsen.

Machine learning skal gøres smartere

Målet med forskningen og udviklingen af algoritmer er på den måde ikke blot et spørgsmål om at gøre dem mere komplekse, men også at gøre træningen af dem hurtigere – og dermed billigere.

Problemet er i den sammenhæng, at træningen af modellerne næsten ikke har udviklet sig de seneste 30 år. Fra 1990'erne og frem til i dag er det den samme datalogiske tilgang, der i 2012 for første gang gjorde machine learning mere effektivt til at genkende billeder, end hvad en programmør kunne kode ind i en algoritme. Det er de samme

grundlæggende regler for læring.

Den begrænsende faktor var indtil 2012, at computerne var for langsomme, men da blev hardwaren endelig kraftig nok til, at man kunne træne på en tilpas stor mængde billeder.

»Det var altså ren hardware-innovation på det tidspunkt. Siden da har algoritmerne stadig ikke ændret sig ret meget, man har bare skiftet hardwaren ud med mere moderne hardware, som har tilladt at træne på mere data og bruge flere parametre. I princippet kunne man godt fortsætte på den måde, men så bliver omkostningerne i form af strømforbrug så store, at det ikke giver mening. Derfor er den eneste reelle mulighed for forbedringer at finde på bedre algoritmer. Bedre algoritmer gør træningen hurtigere og sparer på den måde både strøm og tid,« forklarer Kasper Green Larsen.

Han fortæller, at der er behov for at udvikle nye metoder til at træne algoritmerne. En mulighed er blandt andet at udvikle algoritmer, som ikke kræver så store datasæt at træne på, men som samtidig kommer med de samme præcise forudsigelser.

En anden mulighed er at komprimere data, hvilket er noget af det, som Kasper Green Larsen blandt andet selv arbejder med.

»Ved eksempelvis billedgenkendelse er det måske ikke nødvendigt med alle data i alle pixels for at kunne genkende, hvad der er på billedet. Drejer det sig om en mand i en blå skjorte, skal vi lære algoritmen, at den ikke behøver at tage alle de blå pixels med i beskrivelsen af hele billedet. Man skal finde ud af, hvor få parametre man skal bruge for at kunne beskrive den enkelte pixel og det enkelte punkt på billedet. Den slags fremskridt er nødvendige for at bringe kravene til algoritmerne ned, så de kan udvikles mod fremtiden uden at blive for omkostningstunge,« siger Kasper Green Larsen. ■