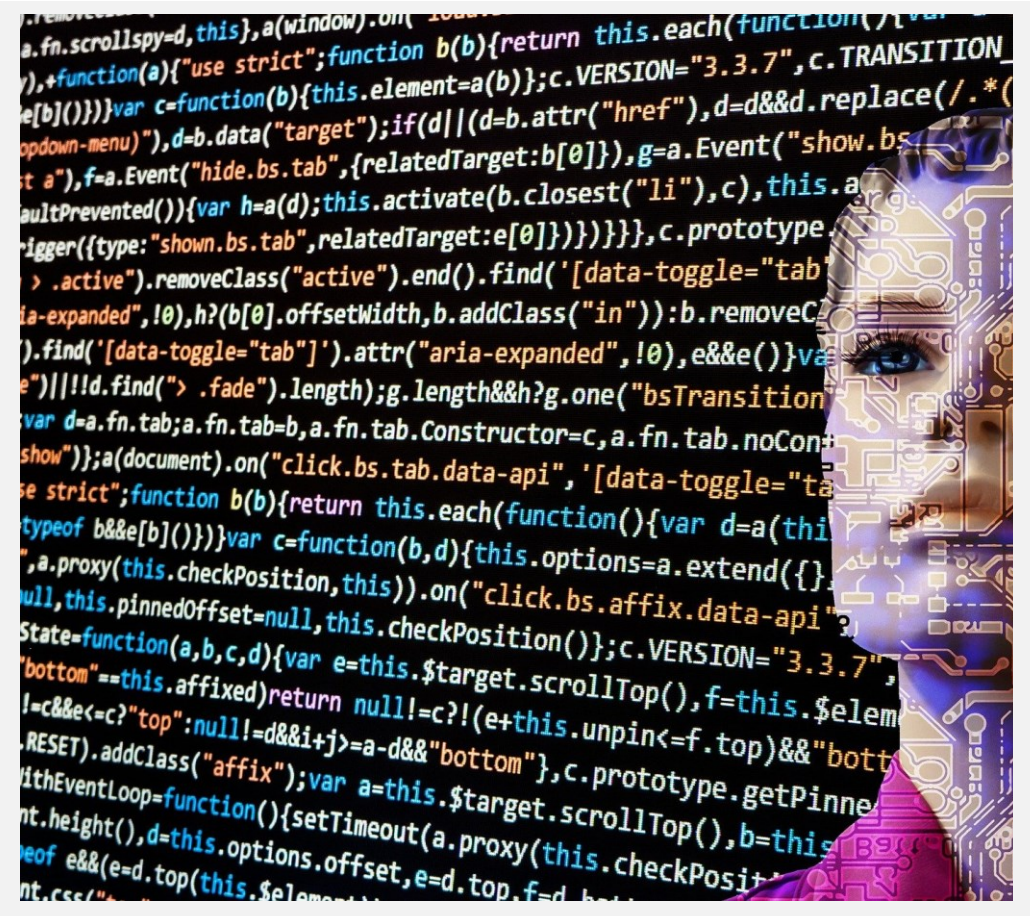
Trendalert

S E P T E D

Zichzelf programmerende computers

Samenvatting

De digitale revolutie heeft de afgelopen decennia een grote invloed op ons dagelijks leven gehad en ook de watersector komt wat dit betreft op stoom. Het is in vrijwel ieder proces binnen de watersector denkbaar dat inzichten uit data tot verbeteringen kunnen leiden. Het realiseren van dit inzicht vergt vaak het schrijven van code en/of algoritmen, wat vooralsnog specialistisch werk is. Als iedere werknemer van een waterbedrijf *zelf* de analyses zou kunnen programmeren die haar of hem helpen het werk nog beter uit te voeren, hoe veel sneller zou de sector dan de vruchten van haar digitalisering kunnen plukken? Nieuwe technieken voor het genereren van computercode op basis van natuurlijke taal zullen dit op korte termijn mogelijk maken. Het is nodig om in een vroeg stadium mechanismen te implementeren voor validatie van programma's en resultaten.



Consequenties voor u

	Laag	Middel	Hoog	Beknopte uitleg
Impact				inzichten uit data voor niet-specialisten
Zekerheid				doorontwikkeling lijkt kwestie van tijd

Trendbeschrijving en achtergrond

Van idee naar realisatie

De digitale revolutie heeft de afgelopen decennia een grote invloed op ons dagelijks leven en op we met elkaar omgaan gehad. In de watersector is deze digitalisering zich op dit moment ook aan het voltrekken (IWA, 2019). Deze transitie bestaat enerzijds uit het realiseren van infrastructuur en processen voor het verzamelen, beheren en toegankelijk maken van data en anderzijds het daadwerkelijk gebruiken van de data om tot betere inzichten en beslissingen te komen. In vrijwel ieder proces binnen de watersector is het denkbaar dat inzichten uit data tot verbeteringen kunnen leiden. Het realiseren van dit inzicht vergt vaak het schrijven van code en/of algoritmen. Dit is vooralsnog specialistisch werk dat dus door specialisten wordt uitgevoerd, maar hiermee worden veel kansen gemist. Als iedere werknemer van een waterbedrijf de analyses zou kunnen programmeren die haar of hem helpen het werk nog beter uit te voeren, zonder afhankelijk te zijn van een programmeur, hoe veel sneller zou de sector dan de vruchten van haar digitalisering kunnen plukken? Programmeren is namelijk de brug tussen een idee (om inzicht te verkrijgen uit data) en zijn realisatie. Echter, net als het leren van een nieuwe taal, vergt het leren programmeren heel veel tijd en niet iedereen heeft interesse of bereidheid om te leren programmeren, ondanks de wens om hun ideeën te realiseren. De afgelopen decennia is er al werk verricht om het schrijven van code toegankelijker te maken; de volgende, grote, stap wordt gezet door gebruik van kunstmatige intelligentie. Kunnen we binnenkort programmeren zonder al te veel kennis over programmeren? Het antwoord zal in de nabije toekomst 'Jazeker' zijn.

Low-code en no-code programmeren

Vanaf de jaren ‘90 van de vorige eeuw zijn er zogenaamde low-code-ontwikkelingsplatforms geweest. Deze maakten de creatie van softwareapplicaties mogelijk en gemakkelijk(er) via een grafische gebruikersinterface waarin de belangrijkste elementen van een programma met muisklikken aan elkaar gekoppeld konden worden en waarbij een beperkter deel van het werk daadwerkelijk in een programmeertaal hoefde te worden geschreven.
Later kwamen hier de zogenaamde no-code-ontwikkelplatforms bij. Hierbij is voor specifieke taken helemaal geen schrijven van code in een programmeertaal meer nodig. Het traditionele low-code ontwikkelplatform vereist nog steeds kennis van de opbouw van software en het gebruik van een dergelijk platform. Met andere woorden, gebruikers moeten nog getraind worden om deze te leren gebruiken en zij zijn niet voor iedereen toegankelijk. Dit is met traditionele no-code-platforms in mindere mate het geval, maar deze kennen andere beperkingen (zie Tabel 1).
Een volgende stap in het schrijven van computerprogramma’s zou zijn als verzoeken geformuleerd in natuurlijke taal direct over te zetten zouden zijn naar computercodes. Als analogie: vroeger moesten we de schakelaar gebruiken om het licht aan of uit te zetten; nu kun je gewoon je homecontrolsysteem verbaal de opdracht geven om dit te doen. Van deze

Tabel 1: Vergelijking van de belangrijkste kenmerken van low-code en no-code programmeren (gebaseerd op https://warewolf.io/blog/no-code-vs-low-code/).

Low-code	No-code
• vaardige ontwikkelaars, achtergrondkennis nodig	• kan door vrijwel iedereen worden gebruikt
• training en certificering vaak nodig	• weinig training nodig
• flexibel m.b.t. specifieke behoeften	• minder flexibel m.b.t. specifieke behoeften
• uitbreidbaar (extern te koppelen)	• tools zijn vaak specifiek voor een bepaald toepassingsveld
• diverse usecases – bijna alles wat met traditioneel programmeren gemaakt kan worden is mogelijk	• bruikbaar voor eenvoudige programma’s

S E P T E D

automatische generatie van code op basis van gesproken taal worden nu de eerste implementaties ontwikkeld - dit is de trend die hier wordt gesignaleerd.

Van natuurlijke taal naar programmeertaal

Taalkundigen studeren al eeuwen op de vraag hoe verschillende natuurlijke talen zich tot elkaar verhouden. In de laatste jaren zijn er mogelijkheden gekomen om automatische vertalingen (op basis van kunstmatige intelligentie) tussen verschillende natuurlijke talen te genereren. Er bestaan ook manieren om programmeertalen, die elk ook hun strikte syntaxis (grammatica) hebben, in elkaar om te zetten.
Maar hoe zit het met de omzetting van een natuurlijke taal naar een programmeertaal? Computers kunnen nog niet denken, maar we kunnen computers zeker leren ons te begrijpen. Ook hier bieden de zich snel ontwikkelende computerwetenschap en kunstmatige intelligentie perspectief.
Microsoft Word biedt in zijn laatste versie de mogelijkheid om teksten te dicteren in plaats van ze in te typen. En zoals eerder genoemd is het mogelijk om met platforms als Google Translate of DeepL een directe vertaling tussen natuurlijke talen te laten maken. En velen van ons hebben inmiddels een slimme assistent in huis of op onze telefoon (Alexa, Siri). In alle gevallen is de achterliggende techniek natural language processing (NLP). Dit is een op deeplearning gebaseerde benadering om computers betekenis uit context en in samenhang te laten begrijpen. Een andere veelbelovende toepassing van NLP vormt de verbinding tussen menselijke talen en programmeertalen. Zou het niet prachtig zijn als we tegen onze computer zouden kunnen zeggen ‘download KNMI-gegevens en voorspel rivierafvoer Rijn’ en als de computer dan code zou genereren om deze opdracht uit te voeren?
De eerste stappen in deze richting zijn reeds gezet. Zo heeft OpenAI (2021) het Codex-platform met API opgezet, dat natuurlijke taal in een twaalftal programmeertalen kan omzetten. Een demo hiervan die het bedrijf op YouTube heeft gedeeld is indrukwekkend (zie https://www.youtube.com/watch?v=Ru5fQZ714x8&ab_channel=OpenAI). Het begrijpen van menselijke talen is echter veel moeilijker dan het genereren van een code, gezien de complexiteit van menselijke talen. Het idee heeft daarom veel tijd nodig om uitontwikkeld te worden. Codex vereist een enorme hoeveelheid codevoorbeelden om deeplearningmodellen te trainen en honderden programmeurs om modellen te valideren. De verwachting is echter dat directe vertaling van opdrachten in natuurlijke taal naar een programmeertaal, zoals in het voorbeeld van CodeX, naar onze inschatting binnen 3-5 jaar voldoende rijp zal zijn voor toepassing in de dagelijkse praktijk van drinkwaterbedrijven.

Relevantie

Zoals veel medewerkers van waterbedrijven nu al eenvoudige en complexere berekeningen in Excel uitvoeren, zullen zij met een dergelijke AI-aangedreven tool kunnen zelf programma’s maken die precies en op eenvoudige wijze uitrekenen of presenteren wat zij nodig hebben voor hun dagelijkse werk, zonder daadwerkelijk te hoeven programmeren. We komen in de buurt van wat de meesten van ons echt willen van computers - we zeggen wat we willen en zij doen het - op een efficiëntere en effectievere, beter georganiseerde manier.
We moeten de complexiteit van menselijke talen echter niet onderschatten. Zelfs in de dagelijkse gesprekken tussen mensen moeten we de taal in een bepaalde context begrijpen en soms onze onduidelijke uitdrukkingen verduidelijken. Daarom kunnen we niet verwachten dat computers al onze verzoeken (in de beginfase) perfect begrijpen. In plaats daarvan is een interactie tussen de computer en de mens nodig om het verzoek te bevestigen, zoals 'wil je de *uur*- of *dag*gegevens van het KMNI downloaden?'
Hoewel het niet perfect zal zijn als het op de markt komt, verwachten we dat veel overbodig werk kan worden verminderd en dat de precisie van de autocodering geleidelijk zal stijgen met de actieve feedback (validatie) van gebruikers. Iedereen,

S E P T E D

programmeur of niet, zal zijn of haar ideeën in programma's kunnen realiseren.
Met deze techniek in handen zullen vrijwel alle medewerkers van drinkwaterbedrijven in staat worden gesteld om veel meer te halen uit de data die zijn voor hun dagelijkse werk nodig en beschikbaar hebben, om daarmee hun beslissingen nog beter geïnformeerd te nemen en hun werk nog beter uit te voeren.

Validatie wordt nog belangrijker

Professionele programmeurs en datascientists zullen nodig blijven voor de meest complexe problemen, standaardisatie, veiligheid, opschalen, integratie etc. Want naarmate de complexiteit van berekeningen toeneemt (zoals in deze stap van excel naar "autocode"), neemt ook de kans op fouten toe en de kans dat deze worden opgemerkt neemt af. Dit kan resulteren in een toename aan "foute" beslissingen. Om dit te voorkomen is het nodig om reeds in een vroeg stadium mechanismen voor validatie van programma's en resultaten in te bouwen in procedures en gedrag. Dit moet ook onderdeel zijn van de training van mensen, en zal de beschikbaarheid en inzet van een toegewijde groep professionals vergen.

Meer informatie

- IWA (2019) Digital Water – Industry leaders chart the transformation journey.
- OpenAI (2021) https://openai.com/blog/openai-codex/, bezocht op 22 november 2021.

Keywords

Hydroinformatica; natural language processing; programmeren