Testen met Artificial Intelligence

whitepaper
TestNet werkgroep “Testen en AI”
versie 1.0 - mei 2019

door:
Sander Mol, Hannie van Kooten, Gerald de Vrieze, Rik Marselis,
Derk Harmanny, Lysette Caetano do Rego, Marco Verhoeven,
Marek Lof, Martin Heining, Martin van Helden, Oktorijanto Wahjuwibowo,
Ramin Sadeghi Zadeh, René Hegeman, Reza Samie Fanny, Sander Schrama,
Sebastiaan van Houdt en Taco Verhagen.

Inhoud

1. Introductie

Zelfdenkende en zelflerende systemen spelen een steeds grotere rol in de samenleving. Ook het testvak is behoorlijk aan het veranderen doordat kunstmatige intelligentie (Artificial Intelligence, AI) zijn intrede doet. De TestNet werkgroep “Testen en AI”, tot mei nog onder de naam “Testen met AI”, onderzoekt wat de mogelijkheden zijn van testen met behulp van AI en geeft met deze whitepaper inzicht in de huidige stand van zaken en een blik op de toekomst.

De aanleiding voor het starten van deze werkgroep was een artikel[1] dat Sander Mol en Rik Marselis schreven over “testen met zelflerende en zelfexplorerende testtools”, gevolgd door een presentatie[2] op een TestNet thema-avond in september 2017.

Wat is AI?

Simpel gezegd is AI het vermogen van machines om taken en activiteiten uit te voeren die wij beschouwen als 'intelligent'. AI, breder gedefinieerd, is het vermogen van een intelligent systeem om zijn omgeving te observeren en specifieke taken uit te voeren om zo goed mogelijk een doel te bereiken.

Bij testen is het doel om informatie te verzamelen over de kwaliteit en de risico’s van een informatiesysteem. AI kan dit ondersteunen door het informatiesysteem te observeren en gegevens te verzamelen. Deze verzamelde gegevens kunnen worden gebruikt in een rapportage. Ook kan de AI zelf van de verzamelde gegevens leren en daardoor steeds beter worden in het uitvoeren van de taken.

De voor deze whitepaper relevante vormen van AI zijn gebaseerd op “Machine Learning”: een slim algoritme analyseert gegevens en kan op basis daarvan tot conclusies komen. In het geval van unsupervised learning zorgt het algoritme voor groepering van gegevens. Bij supervised learning helpt de mens bij het vaststellen van de groepering, bijvoorbeeld door de gegevens te labelen. Meestal gaan unsupervised en supervised learning samen.
Als een model goed genoeg is (en dus voldoende heeft geleerd), dan kan het bevroren worden. In andere situaties is het gewenst dat het model ook tijdens de gebruiksfase doorgaat met leren.

Op www.ai-cursus.nl is een algemene inleiding tot AI te vinden. Deze cursus wordt aangeboden door de Nederlandse overheid en is opgesteld in samenwerking met universiteiten en het bedrijfsleven. Wil je vooral de meestgebruikte AI technieken leren kennen, dan is hiervoor een eerste introductie[3] te vinden op onze TestNet pagina[4].

[1] https://www.testnet.org/testnet/download/werkgroep-testen-met-ai/2017-03-testen-met-zelflerende-en-zelf-explorerende-testtools.pdf
[2] https://www.testnet.org/testnet/download/werkgroep-testen-met-ai/2017-09-hoe-gaan-testrobots-ons-testers-helpen.pdf
[3] https://www.testnet.org/testnet/download/werkgroep-testen-met-ai/2018-05-ai-introductie-voor-testers.pdf
[4] https://www.testnet.org/testnet/p000610/werkgroepen/werkgroep-testen-met-ai

AI in het testvak

Als gesproken wordt over “Testen en AI”, dan leidt dit tot verwarring. Want het kan gaan over “Testen van AI” waarbij dus de kwaliteit van een AI-systeem wordt onderzocht, maar ook over het “Testen met AI” waarbij AI gebruikt wordt ter ondersteuning van het testwerk. Dit document gaat over “Testen met AI”, waarbij AI een techniek is die testtools efficiënter en/of effectiever moet maken.

Een regelmatig opduikende discussie is of een testtool wel “echt AI” is. Sommige testtools zijn “slechts” Business Intelligence (BI) of “slechts” statistiek. Wat gezien wordt als “echt AI” is in de tijd en per situatie verschillend, er is geen alomvattende onbetwistbare definitie. Per saldo gaat het allemaal om de interpretatie van veel data (al dan niet live verzameld) om de testen te verbeteren. Dus we zijn vooral op zoek naar ‘tools die grote hoeveelheden data gebruiken om testen slimmer te maken’.

De ultieme situatie is straks dat AI in staat zal zijn om zelfstandig een applicatie te verkennen, de kwaliteit en risico’s te beoordelen en hier een goed leesbaar verslag over te schrijven. Voorlopig zijn we hier nog lang niet. Aan de andere kant is wel al duidelijk dat er niet sprake zal zijn van één AI-testtool dat dit allemaal doet, maar van een keten aan AI-testtools die samen de verschillende taken afhandelen. In het groeitraject naar de ultieme situatie toe zullen we zien dat eerst één taak met AI wordt opgepakt en dat langzamerhand steeds meer taken aan de AI-testtool worden toevertrouwd.

Of het binnen afzienbare tijd zo zal worden dat al het testwerk wordt overgenomen door AI betwijfelen wij. Sommige test-taken zijn eenvoudig over te dragen, maar voor andere taken is inzicht en creativiteit nodig, inbreng die specifiek van menselijke testers afkomstig is.

Deze whitepaper behandelt de huidige stand van zaken (voorjaar 2019) vanuit het hele testproces en kijkt waar AI nu en in de toekomst meerwaarde kan bieden. Het is het resultaat van de discussies binnen de werkgroep en met nationale en internationale experts, onderzoek en demo’s rond bestaande AI-testtools en eigen experimenten met de verschillende concepten.

De werkgroepleden wensen u, onze lezer, veel succes en plezier met het toepassen van AI in het testvak.

2. Mogelijkheden voor testen met AI

Wat kan AI - en dan vooral machine learning - nu écht toevoegen aan het testvak? Om dat verder uit te werken, is eerst een beter begrip nodig van wat AI daadwerkelijk doet. Dit hoofdstuk bespreekt de belangrijkste bouwstenen, in de context van het testproces.

Objecten herkennen

Bestaande testtools hebben twee manieren om objecten te herkennen; door het vastleggen van eigenschappen (zoals afmetingen, tekst in het object of XPath) of door het vastleggen van het uiterlijk en vervolgens de pagina te scannen. In het laatste geval kan een afbeelding worden omgezet naar een tekst (ofwel OCR). Ondanks al deze mogelijkheden, is een veel voorkomend probleem van user interface-testen dat ze traag en broos zijn en veel onderhoud vergen.

Het gebruik van AI kan hiermee helpen. AI kan leren wat het uiterlijk en de eigenschappen van een object zijn. Als hier later iets in verandert, zal de AI een bepaalde mate van zekerheid teruggeven dat het veranderde object tóch is wat je zoekt. De AI vindt bijvoorbeeld het object Login Button niet met 100% zekerheid, maar wel een object dat er 78% op lijkt en een ander object dat er 65% op lijkt. Je kunt als tester vooraf bepalen hoe zeker de AI minimaal moet zijn, om je test toch automatisch te laten vervolgen. Uiteraard moet later ook worden onderzocht wat de reden van de verminderde herkenning was, zodat dit kan worden opgelost.

Ondanks dat deze verbeterde herkenning intelligent oogt en daarmee AI is, is het nog niet de machine learning toepassing die ons écht kan verbazen. Die toepassing zit in het vermogen van een AI om objecten breder te interpreteren op basis van heel veel voorbeelden en op basis van context.

Het leren van voorbeelden houdt in dat we heel veel verschillende objecten aan een AI laten zien, die wat ons betreft tot dezelfde groep behoren. We zullen alles dus moeten labelen, wat het een vorm van supervised learning maakt. Zo zullen alle buttons op ons testobject - lees voor het gemak ‘website’ - waarschijnlijk ongeveer dezelfde eigenschappen hebben. Als een AI daarop traint, kan het later een nieuwe button meteen herkennen én gebruiken (zie verderop bij *Verkennend een doel bereiken*). Bovendien kun je je eigen buttons vergelijken met duizenden andere websites en zo een heel robuust idee krijgen van wat ‘een button’ is. Later in deze whitepaper zien we hoe de tool Test.ai dit doet met webwinkels.

Herkennen van objecten wordt nog betrouwbaarder als de AI de context meeneemt. Het kan een complete pagina inscannen (via een geavanceerd, ‘convolutioneel’ neuraal netwerk) en zo bijvoorbeeld inschatten welke buttons tot een menu behoren. En welke buttons waarschijnlijk naar een volgende pagina leiden. En wat je waarschijnlijk met een bepaald object kunt doen (zoals iets klikken, selecteren of typen in keuzelijsten, checkboxes en links). Ook hier geldt; de AI kan deze trends op het eigen testobject herkennen, maar ook op duizenden andere testobjecten.

Verkennend een doel bereiken

Een AI kan zelf op onderzoek uitgaan, om zo een testobject te verkennen. Machine learning leert van data, dus ook van data die een AI zelf genereert. Dit concept heet reinforcement learning; de manier waarop een AI handelt, wordt *versterkt* door eerdere ervaringen én door de waardering van die ervaringen. Concreet betekent dit, dat wij als tester onze AI testtool bonuspunten en strafpunten geven voor het bereiken van bepaalde doelen. Het klikken op een link of een button zou bijvoorbeeld 10 punten kunnen opleveren, iedere verstreken seconde 1 strafpunt en het vinden van een bepaalde tekst bijvoorbeeld 100 punten. Vervolgens gaat de AI testtool proberen om zo veel mogelijk punten te behalen.

Het is aan de tester om de doelen goed op te stellen. Wat doen we met een 404-pagina, is dat een bonuspunt of een strafpunt? Dat hangt uiteraard af van wat de tester met de test wil bereiken.

Een logische vraag is dan hoe een AI testtool weet wat zoal de mogelijke handelingen zijn. Dit zou een tester zelf in kunnen stellen, handmatig of door het gehele objectenmodel in te laden. Maar we kunnen het de AI testtool via objectherkenning ook zelf laten uitzoeken.

Trends in data herkennen

AI en machine learning zijn bij uitstek geschikt om grote hoeveelheden data te gebruiken en daar nuttige trends in te vinden, via technieken zoals random forests[5] en met name neurale netwerken[6]. Data - in de vorm van grote hoeveelheden tekst en code - is tegenwoordig volop beschikbaar, al dan niet direct in bruikbare vorm.

Bij een bestaande applicatie kun je de logbestanden uit productie gebruiken om de belangrijkste use cases te ontdekken. Ook kun je de gebruikers op basis van hun handelingen clusteren in een aantal categorieën.

Verder kun je data gebruiken om gerichter te testen. Zo kun je een AI op bevindingenregistraties trainen om te ontdekken wat er vaak fout gaat en hoeveel tijd het doorgaans kost om iets op te lossen. Mocht je een goede wijzigingsregistratie hebben, dan kun je ontdekken welke wijzigingen een hoog risico hebben en waar je op zou moeten testen. Een laatste, wat meer complexe toepassing is het analyseren van documentatie, zoals een requirementsdocument of een functioneel of technisch ontwerp. Als testers onderkennen we tijdens de review al onduidelijkheden, onvolledigheden of foutgevoelige onderwerpen. Mits getraind met voldoende voorbeelden, zou een AI hier ook bij kunnen helpen. Dit is echter wat lastiger, omdat deze data veel minder structuur bevat dan logbestanden.

Tenslotte is trendanalyse via AI erg handig voor rapportages en dashboards. Zeker als we eerder onze AI testtool zelf hebben laten verkennen, zullen er talloze paden geprobeerd zijn

[5] Random forests is een techniek om een uitkomst (output) te voorspellen met steeds wisselende variabelen (input), om zo te zien welke combinatie van variabelen de beste voorspellers is. https://en.wikipedia.org/wiki/Random_forest
[6] Neurale netwerken proberen een uitkomst te voorspellen door steeds een kleine aanpassing te doen aan deze voorspelling en te kijken of dit een verbetering is. De input-variabelen berekenen niet rechtstreeks het uitkomst; hier zitten een (vaak groot) aantal lagen met variabelen tussen. Neurale netwerken zorgden voor de opkomst van het begrip ‘deep learning.’ https://nl.wikipedia.org/wiki/Neuraal_netwerk

en zullen we ongetwijfeld wat afwijkend, trend-doorbrekend gedrag en misschien zelfs fouten hebben ontdekt. Een AI testtool kan zelf input leveren voor dergelijke rapportages, al is het op dit moment nog niet duidelijk hoe goed dit voor mensen leesbaar te maken is.

Tekst herkennen

Bij de drie mogelijkheden tot nu toe in dit hoofdstuk hebben genoemd, speelt tekst vaak een rol. Een specifieke AI techniek, Natural Language Processing, helpt om gesproken of geschreven tekst te interpreteren. Deze techniek kijkt niet alleen naar het woord op zich, maar kan ook naar de woorden ervoor en erna kijken om de juiste interpretatie te geven. Dat is ook de reden dat vertaalprogramma's de laatste tijd zo goed zijn geworden.

Data genereren

AI is niet alleen goed in het herkennen van trends in data, het kan met gebruik van deze trends ofwel 'stijlen' ook zelf data genereren. Denk daarbij aan het aanmaken van beelden, zoals de afbeelding hieronder (in dit voorbeeld genereert de AI op basis van de foto's links en boven een nieuwe foto die een soort gemiddelde van de twee foto's is).

Bron: https://www.lyrn.ai/2018/12/26/a-style-based-generator-architecture-for-generative-adversarial-networks/

Voor testers is een meer concrete toepassing het genereren van test-klanten of het genereren van de juiste reactie van de stub van een omliggende applicatie. Ook hier zijn natuurlijk eerst veel voorbeelden nodig om een model te trainen, zoals een klantenbestand of een grote set aan requests en responses van de andere applicatie.
Omdat het zo abstract is, gaan we kort in op de techniek achter data generatie; het Generative Adversarial Network ofwel GAN. Er worden hier twee neurale netwerken

gebruikt. Eén netwerk (de generator) maakt de data, een ander (de discriminator) bepaalt hoe goed de data is (in vergelijking met échte voorbeelden) en geeft vervolgens aanwijzingen aan de generator. Deze cyclus wordt steeds herhaald, totdat de generator in staat is om data te maken die door de discriminator niet meer van echt te onderscheiden is. Je kunt dus een GAN bijvoorbeeld gebruiken om te zien of je geanonimiseerde data kunt herleiden naar productiedata en zo een privacy-toets doen.

3. Uitdagingen bij testen met AI

Terwijl er volop mogelijkheden zijn om testen te verbeteren door het gebruik van AI en machine learning, zien we ook uitdagingen. In dit hoofdstuk beschrijven we de uitdagingen die de werkgroep als meest belangrijk ziet. Verder onderzoek zal nodig zijn om vast te stellen welke onderwerpen het meest urgent zijn. We kijken hierbij niet alleen naar de kortere termijn, maar hebben ook een langere termijn horizon. Sommige van de uitdagingen zullen namelijk vanzelf snel op ons pad komen, bij andere uitdagingen kunnen we er beter preventief aandacht aan geven, zodat we latere repressieve en correctieve maatregelen kunnen voorkomen.

Vertrouwen in AI

AI is voor de meeste mensen iets nieuws. Veel mensen hebben een natuurlijk wantrouwen tegen vernieuwing, bij AI wordt dit versterkt doordat men het moeilijk kan begrijpen en daardoor de mogelijke gevolgen niet kan overzien. De media hebben de neiging om dit beeld te versterken, door veel aandacht te geven aan de kritische geluiden van bijvoorbeeld Elon Musk en Stephen Hawking over het feit dat AI straks misschien doet wat we NIET willen. In Hollywood zijn er genoeg films gemaakt over superintelligentie en singulariteit - het voorbijstreven van menselijke intelligente - met Terminator en Sky Net als cultvoorbeelden. Voorlopig is superintelligentie nog jaren, zo niet decennia of zelfs voor altijd, pure fictie.

Toch zorgt deze angst voor AI voor weerstand. Deze houding leidt tot langzamere ontwikkeling en daardoor concurrentieproblemen ten opzichte van bedrijven en landen die de technologie wél omarmen. In de Europese politiek zie je deze vrees resulteren in richtlijnen, in de GDPR staat onder andere dat een machine klanten niet vol-automatisch mag afwijzen (dit geldt voor zowel AI als ‘normale’ programma’s).

Natuurlijk brengt nieuwe technologie nieuwe risico’s met zich mee. Maar veel van die angst is naar onze mening ongegrond. Hoe kun je het niet reële deel van deze angst wegnemen? Kennis wat AI echt is helpt om het mysterie tastbaar te maken, zoals via de nationale AI cursus die we in de inleiding al noemden.

Naast algemene kennis helpt het om praktisch inzicht te geven in de werking van AI, door geleidelijke invoering en controle over en op AI en haar resultaten helpen. Laat de AI eerst suggesties geven, gebruik het parallel aan de bestaande manier van werken. Pas het toe op simpel en herhaald werk, zodat het snel waarde kan toevoegen. Breid het pas uit zodra er genoeg vertrouwen is.

Vertrouwen in AI als testhulpmiddel

Het thema ‘vertrouwen’ speelt uiteraard ook bij het gebruik van AI testtools. Wij zijn er van overtuigd dat AI gaat helpen bij allerlei testtaken, zoals we in hoofdstuk 4 zullen toelichten. Maar om ons belangrijke testwerk over te laten aan een testtool moeten we op zo’n tool

kunnen vertrouwen. De eerste vraag die daarbij opkomt is: “hoe weten we dat de AI-testtool goed is? Wie heeft dat getest?”

Hoe weet je of een AI-testtool goed (genoeg) is?

Ook voor AI-testtools geldt, dat er nog weinig praktijkervaring beschikbaar is vanuit de organisaties die ze gebruiken. Op internet zijn al enkele beperkte ervaringsverhalen te vinden. Daarnaast doen sommige AI-testtool-leveranciers grote moeite om de mogelijkheden en kwaliteiten van hun testtools over het voetlicht te brengen. Toch is de beste manier om te bepalen of zo’n tool in de praktijk toegevoegde waarde levert, het gewoon uitproberen. Bijvoorbeeld via een pilot op kleine schaal, om vervolgens met de ervaringen de aanpak bij te stellen.

Hier komen echter andere praktische uitdagingen bij kijken. Het aanschaffen van een AI-testtool via een leverancier brengt kosten met zich mee, terwijl het door het beperkte inzicht in AI en machine learning maar de vraag is of er genoeg voordeel mee wordt behaald. Aan de andere kant is zelf een testtool ontwikkelen een uitdaging, doordat er diverse inzichten en vaardigheden nodig zijn, die bij een traditioneel geprogrammeerd programma níet nodig waren (zoals kennis van machine learning modellen en statistiek en vaardigheid in het bewerken van een grote hoeveelheid data).

Het helpt dan ook om, voordat er een AI-testtool wordt gekozen, een duidelijk doel te stellen. En vervolgens op basis van praktijkervaringen dit doel bij te stellen. Dat haakt meteen aan bij het volgende onderwerp: verwachtingsmanagement.

Te hoge verwachtingen

Terwijl men aan de ene kant van het gebruikers-spectrum heel sceptisch is, is men aan de andere kant misschien juist over-enthousiast. Afnemers van AI testtools denken soms dat op magische wijze al hun problemen worden opgelost, al dan niet daartoe aangezet door leveranciers van deze tools.

In hoofdstuk 5 gaan we dieper in op praktische toepassingen, maar het is duidelijk dat AI op dit moment vooral een hulpmiddel is bij simpele taken, waarbij getraind kan worden op een groot aantal kant-en-klare voorbeelden. Denk daarbij aan verbeterde objectherkenning of het automatisch verkennen van een beperkt aantal functionele paden.

Het toepassen van documentanalyse, het genereren van een bruikbaar model of het bijdragen aan non-functionele testen staan écht nog in de kinderschoenen. Het is van belang om hier, afhankelijk van de omgeving waarin je AI testtools gaat toepassen, voorlopig niet of anders heel voorzichtig hiermee te starten.

AI testen is afhankelijk van data

AI, en dan specifiek machine learning, is het herkennen en toepassen van patronen in gegevens. Het analyseren van grote aantallen gegevens levert de algoritmes op waarmee

nieuwe gegevens beoordeeld kunnen worden en een bijpassende actie kan worden uitgevoerd. De kwaliteit van deze gebruikte gegevens is dan ook rechtstreeks verbonden met de kwaliteit van het gegenereerde model.

Transparantie van beslissingen

Bij een AI heb je soms wel en soms niet volledige toegang tot de invoerdata, het model en de uitvoer. Om transparantie te bereiken wil je een “glass-box” benadering waarbij herleidbaar is hoe het AI-algoritme tot conclusies en acties is gekomen. Met bepaalde soorten machine-learning is deze glass-box benadering haalbaar. Andere vormen van machine learning (zoals deep-learning algoritmes) zijn zodanig complex dat het in de praktijk onmogelijk is om volledig transparant te zijn. Zelfs als je logging bijhoudt van alle keuzestappen van het model, kom je op een ondoorgrondelijke berg van miljoenen deelbeslissingen.

Een voorbeeld van haalbare transparantie is aangeven welke groep pixels van een plaatje de doorslag geven voor de AI om het betreffende plaatje bijvoorbeeld te classificeren als een kat. Een plaatje is echter 2-dimensionaal en daardoor makkelijk te visualiseren. Een model dat traint op 100 variabelen, is 100-dimensionaal en is daardoor veel moeilijker te visualiseren. Wel wordt eraan gewerkt om dit zo goed mogelijk te doen, door in dit geval de 100 variabelen terug te leiden tot maximaal 3 variabelen om het toch te kunnen visualiseren. Voor software zijn er technieken ontwikkeld om glass-box te testen, voor AI zijn we dit nog aan het ontdekken. Een term die in dit verband steeds vaker opduikt is “explainable AI”, afgekort tot XAI.

Paradigma’s geven subjectiviteit

Het trainen van een AI doe je vanuit een set beschikbare data. Per definitie geeft deze data een gekleurd beeld, bijvoorbeeld doordat in het verleden is besloten wát je vastlegt, wannéér je het vastlegt en hóe je het vastlegt. Daarnaast komt deze data altijd uit een subset van de werkelijkheid. Bij een ziekenhuis in Den Haag is dit bijvoorbeeld ‘alle mensen die zich ooit in dit ziekenhuis hebben aangemeld’, waarmee je talloze groepen mensen niet meeneemt bij het model dat je gaat maken.

Daarnaast gebeurt het trainen van een AI altijd vanuit een bepaalde achtergrond. De selectie en waardering van de data die je gebruikt voor het trainen van het model, is ook weer een bewuste en daarmee gekleurde keuze. Het is dus van invloed op het model wie deze keuzen maken. Denk daarbij ook aan cultuur, godsdienst, opleiding, geslacht, afkomst, rijkdom en dergelijke.

Dit samen kan leiden tot een getraind model dat gekleurde resultaten geeft. Een inmiddels klassiek voorbeeld is de selectie van medewerkers bij Amazon[7]: historisch gezien zijn bijna alle hoog gewaardeerde managers mannelijk, omdat er vroeger vrijwel geen vrouwelijke managers waren. Een AI leert op basis van deze historische data dan ook een voorkeur te

[7] https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G

hebben voor mannelijke managers, tenzij je zorgt dat geslacht expliciet niet meegenomen wordt tijdens het trainen van het machine learning algoritme.

Bruikbare data krijgen kost veel tijd

Naast het feit dat AI vaak niet transparant en potentieel onethisch is, is er nog een andere uitdaging bij het gebruik van veel data; meestal is deze data niet meteen geschikt om op te trainen. Dit is bij Business Intelligence oplossingen een wezenlijk probleem. Data- en business analisten worstelen hier al jaren mee en het is niet een probleem puur voor het testen met AI, maar over de hele breedte van het toepassen van AI.

Datasets zijn niet compleet, bevatten onduidelijke waarden en zijn gedurende de jaren volgens wisselende richtlijnen gevuld. Bovendien kan de waarde van de data via verschillende standaarden gevuld zijn; denk maar aan verschillende meeteenheden (zoals lengte in meters of inches) of standaarden (zoals postcodes in Nederland of België).
Het opschonen van de data voordat deze door de AI gebruikt kan worden, moet dus expliciet en weloverwogen in de plannen worden meegenomen.

Ontbrekende data geeft nog een extra uitdaging. Als het niet mogelijk is om het hele record te verwijderen, bijvoorbeeld omdat dan de helft van de dataset vervalt, dan zullen er duidelijke en eerlijke conversieregels moeten worden bedacht om deze data aan te vullen.

AI projecten zullen hier in het algemeen hinder van ondervinden, maar dit geldt ook voor het gebruik van data voor AI testen. Denk bijvoorbeeld aan bevindingenregistraties waaruit je patronen wilt herkennen.

Technisch issue: overfitting

Bij overfitting heeft het AI-systeem een algoritme geleerd dat erg precies overeenkomt met de voorbeelden van de trainingsset, maar is het systeem niet goed in staat om nieuwe situaties in te schatten. Bijvoorbeeld: de AI herkent perfect de 20 katten- en honden-plaatjes waarmee getraind is maar kan een nieuwe kat, of een kitten of zeldzaam ras, dat niet was opgenomen in de trainingset, niet correct classificeren.

Bij testen met AI kan dit bijvoorbeeld optreden als je een object, zoals button of link, wilt herkennen binnen de applicatie die je aan het testen bent. Als je de AI traint om precies de bestaande buttons en links te herkennen, zal hij later misschien nieuwe buttons en links níet herkennen. Hetzelfde geldt bij het trainen op bevindingenregistraties; ook hier moet het model voldoende blijven generaliseren om een waardevolle voorspelling van bijvoorbeeld ernst of doorlooptijd te kunnen geven.

Interpreteren van AI testresultaten

Een eigenschap van AI - en dan vooral weer machine learning - is dat deze zelf patronen herkent in grote hoeveelheden voorbeelden en zelf paden zoekt op basis van een groot aantal verkenningen. Dit maakt het een uitdaging om de resultaten van zo’n AI testtool te interpreteren.

Het eerste deel van deze uitdaging is variabele resultaatvoorspelling; als de testtool een object niet (meer) herkent, is dan het object daadwerkelijk verdwenen of heeft de testtool getraind op nieuwe, afwijkende voorbeelden?

Het tweede deel is de hoeveelheid resultaten die een testtool teruggeeft. Na een vrije verkenning van de applicatie komt deze misschien terug met duizenden verkende paden en honderden situaties die de AI het bekijken waard vindt. Zeker als er net wordt gestart met een zelflerende AI testtool, kan het een enorme klus zijn om alle uitkomsten te beoordelen.

En zelfs nadat we onderscheid hebben gemaakt tussen relevante en niet-relevante bevindingen van de AI testtool, dan hebben we waarschijnlijk een veel groter aantal metingen waar we één eindoordeel over moeten geven. Testers en diens opdrachtgevers zullen het een uitdaging vinden om dit oordeel te vellen, wat mede de opkomst van het vakgebied ‘decision science’ verklaart; het nemen bedrijfsmatige beslissingen op basis van data en modellen. Meer over dergelijke gevolgen voor het testvak is te vinden in het hoofdstuk ‘Toekomst van het testvak’.

4. Gebruik van AI in testactiviteiten

In het testvak worden veel verschillende testactiviteiten beschreven en er zijn diverse modellen. Voor de test-activiteiten zijn de TMap-activiteiten als uitgangspunt genomen, maar ze zijn uitgebreid om enkele onderwerpen wat beter te kunnen uitwerken. Wij hebben ervoor gekozen om de volgende indeling van testactiviteiten te gebruiken:

Test management	Plannen	Risico management	Beheer	Rapportage	Bevindingen-beheer	
Test engineering	Voorbereiden	Specificeren	Uitvoeren	Afronden	Infrastruc-tuur	Testdata-management

Voor elk van de activiteiten beschrijven we kort wat de activiteit inhoudt en vervolgens hoe AI kan helpen bij deze activiteit.

Testmanagement

Plannen

Bij het plannen van de testen zal de testmanager zich eerst oriënteren op de wijziging(en) in het IT-landschap (software en hardware) om het **doel**, de **scope** en de **risico's** van de testactiviteiten te bepalen en een doorlooptijd te schatten. Deze informatie kan komen uit geordende en ongeordende documenten en uit spraakopnamen. AI-technieken zoals Natural Language Processing (NLP) kunnen helpen om deze teksten en spraak om te zetten, te analyseren en zelfs samen te vatten.

En AI kan vervolgens ook ingezet worden als nieuwe informatie over de wijziging beschikbaar komt, als aanvulling op eerder gebruikte informatie en dus voor het bijstellen van het doel, de scope en het risico.

Risico management

Als bekend is wat de wijziging inhoudt, zal de testmanager onderzoeken wat de risico's zijn zodat de testinspanning verdeeld kan worden op basis van die risico's.
Een AI-tool kan de requirements- of ontwerpdocumenten interpreteren en de belangrijkste woorden en woordcombinaties uit de zinnen halen. De gebruikte AI-techniek is het recurrent neural network (RNN[8]), met daarin een geheugenmodule (Long / Short Term Memory, LSTM). Deze woorden en woordcombinaties krijgen vervolgens een risicoscore. Deze score

[8] Een RNN geeft geen losstaande voorspellingen (zoals 'dit is een kat'), maar neemt daar informatie bij mee die het eerder heeft verzameld. Bij teksten kijkt het naar de samenhang van letters en woorden. De voorspelling welke letter of welk woord er als volgende zou moeten komen, hangt dus af van zowel het vorige woord als een gekozen aantal woorden dat daarvóór stond. https://en.wikipedia.org/wiki/Recurrent_neural_network

moet al eerder zijn opgebouwd in een baseline, ofwel door handmatig een score toe te kennen ofwel door te trainen op voorbeeldteksten en bijbehorende risicoscore.

De bronnen om tot de baseline te komen, hoeven niet alleen gescoorde requirements of ontwerpdocumenten te zijn. Gescoorde bevindingen vanuit een bevindingenregistratie zijn ook mogelijke input. Uiteraard geldt bij alle databronnen dat deze eenduidige en voldoende compleet zijn.

Volgens eenzelfde techniek kunnen risico's worden vertaald naar business impact. Dit is echter een stuk lastiger, want een AI heeft geen zicht op 'echte wereld'-scenario's, ofwel een context. Het kent alleen de dingen waarop het is getraind en het bepalen van business impact is toch vaak maatwerk.

Het laatste deel van risicomanagement is het inzichtelijk maken van de risico's aan degene die beslist over welke risico's moeten worden beperkt (en daarvoor het budget inbrengt). Een AI kan de gevonden risico's samenvatten als basis voor het verdelen van de testinspanningen, en kan ook tijdens het testen de bevindingen en bijbehorende productrisico's overzichtelijk maken en rapporteren.

Beheer

Na de inrichting van het testproces, is het van belang om dit goed te monitoren en om waar nodig bij te sturen.
AI kan daarbij helpen door via smart-dashboarding inzicht te geven in de stand van zaken, op verschillende aspecten zoals dekking van de tests, aangetoonde kwaliteit van het testobject, afgedekte productrisico's en inzicht in de status van bevindingen.
Bovendien kan de voortgang van het testen worden gelogd, zodat een AI hier via data-analyse nuttige inzichten uit kan genereren. Zo kan het suggesties geven over probleemgebieden, zodat de testmanager sneller kan bijsturen.
Bovendien kan via dergelijke analyses en smart-dashboarding worden geëvalueerd of de exit-criteria voor het testen zijn behaald. Een voorwaarde is dan wel dat de exit-criteria heel helder en meetbaar zijn geformuleerd.

Rapportage

Rapportages zijn belangrijk, daarom hebben wij besloten om rapportage als aparte testtaak op te nemen (in TMap is het onderdeel van beheer). Bij rapportage wordt informatie uit de smart-dashboards van monitoring als input gebruikt. Daar wordt een interpretatie gedaan die informatie verschaft om besluitvorming te ondersteunen.

Met smart-dashboards is het niet alleen mogelijk om de huidige situatie inzichtelijk te maken, het kan ook, op basis van gegevens uit testresultaten en gegevens uit monitoring van de live-omgeving, mogelijke toekomstige veranderingen van de kwaliteit van het informatiesysteem voorspellen. Dit maakt het mogelijk om corrigerende maatregelen te nemen nog voordat kwaliteitsproblemen werkelijk optreden.

Bevindingenbeheer

TMap behandelt bevindingenbeheer op meerdere plaatsen, daarom is het hier apart opgenomen. AI kan in sommige gevallen zelfstandig afwijkingen tussen verwachte en werkelijke situatie vaststellen, en op basis daarvan zelfstandig een bevinding vastleggen. Voorwaarde is dat de verwachte en werkelijke situatie duidelijk beschreven en vergelijkbaar zijn, bijvoorbeeld wanneer een oude en nieuwe versie van de applicatie worden vergeleken. Aan de hand van de eerder vastgelegde risico-analyse kan de AI ook een inschatting van de ernst van de bevinding doen aangezien dit meestal samenhangt met het risico.

Ook kan een AI helpen met de beoordeling van door testers ingevoerde bevindingen. Op basis van eerdere bevindingen die al geclassificeerd of zelfs afgehandeld zijn, kan de AI bekijken of een bevinding binnen een patroon valt en zo iets zeggen over de categorie, ernst of doorlooptijd. En wanneer de melder van de bevinding een categorie en ernst heeft ingevoerd, kan de AI kwaliteitscontrole doen op de ingediende bevinding. Om dit alles te kunnen doen zal de AI door de bevinding heen moeten lopen, met dezelfde techniek die werd gebruikt bij het beoordelen van requirements en risico's (RNN met LSTM).
Doordat we de AI via deze techniek laten kijken naar de inhoud van een melding, ontstaat er nog een andere nuttige toepassing: het opsporen van dubbele registraties door patronen in bevindingen te herkennen en de overeenkomst-percentages te rapporteren.

Maar we kunnen nog een stap verder gaan, als we naast bevindingenbeheer ook probleembeheer gebruiken. De AI analyseert een bevinding, doet root cause analyse en geeft tips met betrekking tot het oplossen van bevindingen. Enerzijds door het voorstellen (of misschien zelfs uitvoeren) van concrete aanpassingen in de programmatuur waardoor een bevinding wordt opgelost, maar anderzijds ook door aanpassingen in het proces zodat eenzelfde fout in de toekomst niet meer zal optreden.

Test engineering

Voorbereiden

De test engineer zal beginnen met het voorbereiden van de tests. Dit betekent het doornemen van specificaties, om de functionaliteit en non-functionals te bepalen. Deze informatie kan vastgelegd zijn in geordende en ongeordende documenten maar ook in spraakopnamen'. De AI kan helpen om informatie uit de documentatie te halen via Natural Language Processing met de AI-techniek van RNN / LSTM. Het doel is in dit geval om per object te achterhalen: wat zijn, per status of context, de mogelijke handelingen.
De AI kan deze informatie samenvatten door er een model van te maken, gevuld met alle objecten en paden. Op basis van dit model, gecombineerd met de eerder gevonden risicogebieden, kan een AI vervolgens ontbrekende, onjuiste of conflicterende ontwerpbeslissingen identificeren. En door het meenemen van de context, kan de AI vaststellen dat er verschillende woorden worden gebruikt voor hetzelfde onderwerp.

Ofwel; een AI zou allerlei soorten teksten kunnen analyseren ten behoeve van ordening, maar ook ter ondersteuning van de review.

Specificeren

Bij het specificeren van de testgevallen kan een AI helpen door het kiezen van de juiste testspecificatietechniek. Dit wordt mogelijk zodra de AI is getraind op een groot aantal stukken documentatie (input) en de optimale specificatietechniek (output). Bovendien kun je de AI een score geven van de dekking die een techniek heeft behaald, op basis van het eerder opgestelde model van de applicatie. Zo zou de AI kunnen zien wat het toevoegt om een beslistabeltest te specificeren als er eerder al een procescyclustest is opgesteld.

Als een specificatietechniek is gekozen (of meerdere), dan is het zaak om deze toe te passen op het model. Ofwel; een AI kan helpen om met een testontwerptechniek testsituaties te genereren en die vervolgens in dezelfde of een andere techniek om te zetten naar logische testgevallen en fysieke testgevallen, inclusief benodigde testdata. Vervolgens kan de AI helpen om eventuele dubbele testgevallen (vanuit de verschillende testontwerptechnieken) te minimaliseren, door het aantal handelingen binnen de gehele testset te optimaliseren. Op dezelfde manier kan er ook een optimale regressietest worden gegenereerd.

Als er een koppeling wordt gemaakt met de bevindingenregistratie, dan is het mogelijk om te herkennen welke testen nog niet uitgevoerd kunnen worden. De oorzaak kan dan zijn dat bepaalde delen van het systeem nog niet beschikbaar of betrouwbaar zijn.

Een heel andere toepassing van AI in het kader van testspecificatie is de selectie van testgevallen die tijdens een ontwikkelfase zijn gemaakt, ten behoeve van de regressietest. Aangezien een regressietestset over het algemeen beperkter is dan de oorspronkelijke progressietestset kan de AI op basis van diverse criteria (bijvoorbeeld een relatie met af te dekken risico) een selectie maken van de best passende testgevallen.

Voor de volledigheid; met name in deze paragraaf lopen machine learning en ‘geprogrammeerde intelligentie’ door elkaar heen. Dit sluit echter aan bij het beeld dat AI en machine learning een plaats zullen hebben in het testproces, zonder direct alles over te nemen.

Uitvoeren

Zeker de testen op lager niveau, zoals unit testen, zullen autonoom kunnen worden uitgevoerd door een AI op basis van standaarden en best-practices.

Functionele testen kunnen, conform specificatie, worden uitgevoerd door bestaande testtools. Hier is geen AI voor nodig, tenzij we willen dat een AI een alternatief pad zoekt zodra de test op basis van specificatie vastloopt.

Een interessante variatie op ‘testen conform testspecificatie’ is ‘testen zonder testspecificatie.’ Eigenlijk is dit een exploratory test, waarbij specificatie en uitvoering van de

testen live gecombineerd worden. De AI ontdekt de mogelijkheden, probeert deze mogelijkheden uit en probeert deze mogelijkheden net zo lang te combineren tot het doel bereikt is. Of, als er geen doel is gesteld, tot alle mogelijkheden zijn geprobeerd (wat zeer onwaarschijnlijk is) of de testuitvoering wordt gestopt. Om deze aanpak te hanteren, is er zowel een AI-techniek nodig om de plaatsbepaling te doen (in de vorm van een, meestal convolutioneel, neuraal netwerk) en om de route te bepalen (opnieuw RNN met LSTM).

Bij zowel gespecificeerde testen als exploratory testen, kan een AI helpen om een voorstel te doen voor een bevinding, inclusief categorisering en classificatie. Echter, ook traditioneel geprogrammeerde testtools zijn hier al ver in gevorderd. Hier zal niet veel winst zitten in het geval van losse, enkelvoudige bevindingen. Wel kan het helpen als de AI op basis van trends en patronen de volledige scope van de bevinding rapporteert, in de vorm van “op pagina A, B, G, H en I komt deze bevinding voor.”

Afronden

Een AI kan helpen bij het beoordelen van het testproces. De AI analyseert alle informatie over wat er tijdens het ontwikkelen, tijdens het testen en tijdens de retrospective wordt vastgelegd en gezegd en kan daar trends uit halen die leiden tot suggesties over mogelijke verbeteringen.

Daarnaast legt een tester doorgaans de testware vast. Een AI kan ondersteunen bij versiebeheer, configuratiebeheer, het opslaan van testware en het weer terugvinden van testware.

Testondersteuning

Infrastructuur

In theorie is het mogelijk dat een AI helpt bij het definiëren en klaarzetten van de testomgeving, op basis van de uit te voeren testen. Op basis van gegevens over welke testgevallen er zijn analyseert de AI welke eisen er aan de testomgeving zijn. Op basis van die eisen kan een AI de testomgeving inrichten en klaarzetten. Deze toepassing heeft echter niet heel vastomlijnde inputs en outputs. Voor een losse implementatie lijkt dit niet lonend, maar misschien is het als standaard dienst wél interessant.

Een meer concrete toepassing is het creëren van stubs en drivers. Bij een bestaande applicatie zou dit goed mogelijk zijn; laat de AI leren welke requests en responses er zijn en laat hem vervolgens zelf het resultaat genereren. Bij nieuwe applicaties is eerst de vorige stap (data-generatie) nodig.

Testdata management

AI kan helpen bij het definiëren en klaarzetten van de dataset, op basis van de uit te voeren testen. AI kan op basis van een analyse van live-data synthetische testdata genereren. Daarmee wordt dan gelijk het probleem van privacy-gevoelige testdata voorkomen.

Hier is vrij specifieke kennis van data voor nodig, maar áls het kan, dan zal het via LSTM zijn. Op basis van het model weet je welke data-componenten er nodig zijn, deze kun je vervolgens genereren. Dit maakt het per definitie fictieve data via RNN.

5. Aan de slag

Dit hoofdstuk laat de manieren zien om aan de slag te gaan met AI in het testproces. Omdat vaak de ervaring én het vertrouwen in AI en AI testtools nog moet worden opgebouwd, zal de toepassing ervan vrijwel altijd klein beginnen en stap voor stap worden verbeterd. In dit hoofdstuk geven we drie manieren om te beginnen;

1) Bestaande, kant-en-klare AI testtools gebruiken
2) Gebruik maken van AI diensten en deze koppelen aan de eigen testtools
3) Zelf AI toepassingen ontwikkelen ter ondersteuning van testen

Afwegingen

De kans is groot dat er uiteindelijk meerdere AI (test) toepassingen samenwerken binnen het testproces. De mogelijkheden zijn heel divers, daarom is het van groot belang om af te wegen waar je het beste kunt beginnen. Daarbij zien we drie hoofdvragen die je jezelf kunt stellen.

Wat is het doel?

Hoewel het in deze periode van hype aantrekkelijk is om ‘iets met AI’ te willen doen of machine learning te willen gebruiken, loont het om vooraf na te denken wat je ermee zou willen bereiken. Zoals beschreven in het vorige hoofdstuk, lijkt de meeste winst te behalen in deze onderwerpen:

1. Flexibel en robuust maken van je bestaande testscripts
2. Veel automatische testen genereren, om meer fouten te vinden
3. Zorgen voor uitgebreide unittesten
4. Efficiënter testen; de juiste test op het juiste moment
5. Ondersteuning van het testproces
6. Betere rapportages en dashboards

Als er een idee is welk doel er bereikt zou moeten worden, dan is de volgende vraag; heb ik hier AI en machine learning voor nodig? Kan ik met een eenvoudige query of een draaitabel in Excel niet hetzelfde bereiken?

Overigens kan het doel ook zijn dat je machine learning gewoon eens in de praktijk wilt zien werken. In dat geval is de afweging tussen kosten en baten natuurlijk niet relevant.

Waar sta je nu?

Het toevoegen van waarde gebeurt vanuit de plek waar je nu staat. Misschien heb je al een set met Selenium testscripts. Of werk met je organisatie volledig vanuit Java. Is de applicatie

die je wilt testen een mobiele app? Dit is van grote invloed op de testtools die je kunt gebruiken, want veel AI testtools - zeker de kleinere en daarmee goedkopere - werken alleen in specifieke situaties.

Verder bepaalt de aanwezige technische kennis wat er mogelijk is. Is deze er in het geheel niet, dan ligt een testtool van een leverancier meer voor de hand. Heb je de beschikking over programmeerkennis en wat basiskennis van machine learning en statistiek, dan zou je met zelfbouw of AI diensten aan de slag kunnen gaan.

De derde afweging is in hoeverre er al vertrouwen is in AI, zowel bij jezelf als bij je collega's en management. Wat is de meest handige plek om te beginnen, zodat je meteen toegevoegde waarde hebt en het bovendien de 'koudwatervrees' wegneemt? Dit bepaalt of je ambitieus van start kunt gaan, of eerst wat voorzichtige proefjes zult moeten doen. In dat laatste geval is het aanschaffen van een licentie voor een bestaande AI testtool misschien nog een brug te ver.

Hoeveel ruimte is er voor iets nieuws?

Of je groots of klein begint, is vanzelfsprekend ook een kwestie van budget. Standaard testtools helpen je snel op weg en vragen waarschijnlijk minder onderhoud, maar ze hebben aanschafkosten en zorgen er misschien voor dat je er zelf minder van leert, waardoor je het uiteindelijk minder effectief toepast. Dat laatste is de component tijd; hoe lang mag je ermee experimenteren voordat er resultaat behaald moet worden.

Een goede manier om beschikbare tijd te creëren, is het parallel draaien van de AI-ondersteunde testen aan de bestaande testoplossing. Na een opstartperiode zal zichtbaar moeten worden dat de AI-variant de effectiviteit van de bestaande testoplossing benadert en uiteindelijk inhaalt.

Bestaande testtools gebruiken

In de zomer van 2018 hebben wij (leden van de werkgroep) testtools met AI bekeken, met toolmakers gesproken, websites bestudeerd en webinars gevolgd. Inmiddels is het geruime tijd later en de toolmakers hebben niet stilgestaan. Ook zijn er nieuwe testtools op de markt gekomen. Dit hoofdstuk is dus per definitie een momentopname, wij raden aan dat je zelf nader onderzoek doet uitgaande van de informatie in dit hoofdstuk.

Testtools kunnen we op verschillende manieren indelen, op testsoort, op hoe ze werken, op welke omgevingen ze kunnen draaien, hoe bruikbaar ze zijn. De tools die in dit hoofdstuk zijn opgenomen, hebben allemaal een echte AI component en worden dagelijks in de praktijk toegepast of hebben de mogelijkheid dit binnenkort te gaan doen. De bijlage van dit document bevat een uitgebreid overzicht van deze gegevens.

We noemen vanaf nu wel de testtool, maar richten ons vooral op de soorten toepassingen die we zien.

Dashboards

Veel testtools met ai bevatten dashboards, overzichten van test runs, de kwaliteit en wat en wanneer er getest is, zoals bijvoorbeeld Eggplant.

Een testtool die zich op dit gebied gespecialiseerd heeft is Sealights. Sealights is een testmanagement tool met dashboards. Ze gebruiken AI om informatie over de optimale testdekking te maken. Deze AI is gebaseerd op programmeercode; testers die exploratory testen uitvoeren, moeten hun werk op de code mappen en daarmee de dekking vastleggen. Er wordt gekeken welk deel van de code de meeste bugs oplevert. Het doel lijkt te zijn om zeer zuinig te testen en misschien op den duur te kunnen voorspellen waar mogelijke problemen zouden kunnen ontstaan.

Record playback als basis voor AI testen

Er zijn nogal wat testtools die record playback als techniek gebruiken om een webapplicatie te testen. Als tester neem je scenario's op en de testtool matched alle elementen, zoals een knop, een plaatje of een tekstveld op zoveel mogelijk zaken, zoals naam, plaats van het element in de pagina, tabvolgorde etc. Dat maakt de test stabieler.

De opgenomen testscenario's vormen de baseline, de basis voor de testen. Als er in een volgende run een afwijking wordt geconstateerd dan geeft de tool dat aan. Vaak in de vorm van een visuele vergelijking van de oude en nieuwe situatie. De tester checkt het verschil en geeft aan of dit een fout is of een juiste wijziging. Als dat laatste het geval is neemt de tool de wijziging op in de baseline. Hierdoor kan de tool op den duur leren wat een juiste en een onjuiste wijziging zou kunnen zijn.

Deze tools maken het mogelijk om de scenario's te kiezen die je wil testen. Testtools die op deze manier werken zijn ReTest, Testim en Mabl.

Crawlers

Er zijn testtools die als een spin door de applicatie gaan en paden zoeken en aflopen. Deze tools kunnen over het algemeen geen grote applicaties aan en zijn geschikt voor apps of kleine webapplicaties. De tools werken online en hebben soms ook een groot platform om de apps op iOs of Android of allebei te testen. Elke tool heeft zijn specialiteit.

Test.ai is gespecialiseerd in webshops. Door deze specialisatie is het mogelijk dat deze tool webapps autonoom kan testen.

Sofy doet meer dan de bovenstaande tools en noemt zichzelf een testassistent. De app en paden worden visueel weergegeven dmv een sankey diagram. Daarnaast detecteert Sofy welk framework wordt gebruikt en zoekt actief naar known bugs met het framework op het internet op veelgebruikte sites zoals stackoverflow. Het rapporteren van bugs gaat met een druk op de knop en een reproductiepad en automatische test worden aan de bug rapportagetool toegevoegd.

Nieuwe tools zijn AppTest.ai en Testrigor. Testar is een tool waarmee je crashes, hangers en fouten kunt opsporen. Je kunt hem zelf verder uitbreiden door checks toe te voegen en te programmeren. Het is de bedoeling dat de tool verder ontwikkeld wordt, zodat de tool een model van de geteste applicatie maakt en zo de belangrijkste testen weergeeft.

Visual testing

Er zijn tools die zich specialiseren op het gebied van het ontdekken van verschillen tussen afbeeldingen. In het verleden werkte dit niet goed omdat een tool vaak over een pixel verschil struikelde. Applitools is een tool die probeert slim om te gaan met visuele verschillen tussen twee test runs. Het idee is dat de tool op den duur kan voorspellen welke wijziging correct is. Deze tool kun je als add-on op de bestaande testen gebruiken.

Bestaande test tools die AI toevoegen

Er is een groep van bestaande test tools die ai toevoegen als uitbreiding. Vaak hebben deze tools al functionaliteit als functionele, security of performance testen in hun repertoir. Functionize is zo'n bestaande tool die machine learning inzet. Zij proberen elementen in een webpagina op veel verschillende manieren vast te leggen. Dus niet één identifier, maar veel meer. De tool checkt op al deze identifiers en geeft een score mee. Daarmee kunnen ze proberen ze achterhalen waar de mogelijke oorzaak van een fout ligt, de root cause analyse. Het is mogelijk om gebruik te maken van monitoring van productie-gebruik. Ze hebben een uitgebreide tool die opgenomen data van productie bewerkt tot testdata. Deze testdata wordt beheerd in de tool en naar de testomgeving gepushed.

Appvance laat op zijn website weten vergelijkbare functionaliteit te bieden. Zij gebruiken externe tools voor het opnemen van gebruikerssessies en het bewerken tot testdata en testen. EggPlant gebruikt zijn bestaande testen voor een uitbreiding met AI. Als gebruiker moet je alle velden mappen en de applicatie zo in kaart brengen. Vervolgens maakt Eggplant hier testen mee.

Deze tools beschikken net als de andere tools over een visuele interface die het werken met de tool makkelijker maakt.

Code testen

Diffblue is een tool die java unit testen kan genereren op basis van java code. Dat kan handig zijn als er weinig unit testen zijn op een grote code base. Ze gebruiken hier AI voor.

Hoe intelligent zijn deze tools?

Het beoordelen van een tool is vrij lastig, mede omdat de tools verschillen, maar ook omdat weinig toolmakers erg expliciet over zijn en ai een marketing woord is. En het maakt uit waarvoor de tool gebruikt wordt en welke problemen de tool zou moeten oplossen. Wordt een tool ingezet als een extra tool op bestaande testen, of wordt de tool ingezet voor de totale automatisering van de app of applicatie?

AI afnemen als een service

Alle grote cloud computing-providers bieden nu cloud-gebaseerde AI-producten aan. Het zijn vooral grote bedrijven, omdat het opzetten van deze producten hoge kosten met zich meebrengt. Deze services bestaan uit zowel software als hardware. Samen zorgen ze ervoor dat je data kunt sturen naar een cloud service. Deze service bewerkt de data met behulp van AI en stuurt je vervolgens de resultaten.

Algemene toepassingen

De mogelijkheden zijn vrij uitgebreid. Overal waar interpretatie gedaan moet worden en waar gestandaardiseerde uitkomsten mogelijk zijn, kan een dergelijke oplossing gebruikt worden. Bijvoorbeeld:

- Het bewerken en analyseren van data
- Het analyseren van beeldmateriaal
- Het omzetten van beeldmateriaal naar tekst
- Het analyseren van tekst
- Het omzetten van tekst naar spraak, of andersom
- Het vertalen van tekst

De voorwaarde is, dat er vooraf is getraind op grote hoeveelheden woorden en hun samenhang (ofwel taal) of dat er grote hoeveelheden afbeeldingen zijn getraind op basis van labels. Maar dit is juist waar clouddiensten goed in zijn, deze zijn getraind met miljoenen teksten en afbeeldingen.Het aanbieden van je eigen woorden of afbeeldingen zorgt daarom al snel voor een scherpe herkenning.

Toepassing bij het testen

Het toepassen van deze technieken in het testen lijkt misschien niet voor de hand te liggen, maar ook daar zijn diverse mogelijkheden. Hoe kunnen we dit gebruiken in ons testproces?

- Upload data die vervolgens geconverteerd wordt om dashboards te vullen
- Upload rapportages over bugs om de AI te laten voorspellen waar toekomstige bugs zullen ontstaan
- Upload code coverage, defect rapportage en code wijzigingen, om de AIte laten bepalen welke testen je moet uitvoeren
- Upload software requirements en laat de AI testgevallen genereren
- Upload digitale documenten en gebruik ai om deze te structureren en ordenen
- Laat de AI zoeken naar objecten op het scherm

Een AI systeem heeft zeer veel data nodig tijdens de trainingssessie. Pas na training is er een zinvol algoritme gevonden dat door het systeem ingezet worden. Er bestaan nu niet veel algoritmes die meteen bruikbaar zijn. Want al zijn er standaard getrainde AI modellen beschikbaar, er moet dus getraind worden op de eigen voorbeelden. Dit geldt ook bij het gebruik van cloud diensten.

De huidige belangrijkste aanbieders van deze diensten zijn IBM (Watson), Amazon (AWS), Microsoft (Azure), Google (Cloud AI), Oracle en Salesforce.

Zelf iets bouwen

Naast het afnemen van testtooling met AI, of standaard AI services is het ook mogelijk om zelf AI componenten te ontwikkelen om je testactiviteiten mee te verrijken. Dit is met name van nut voor reeds (lang-) lopende projecten waarbij de testautomatisering al is opgezet. In dergelijke trajecten is al veel geïnvesteerd en is het niet reëel om over te stappen naar nieuwe tooling en technologieën. Zelfbouw AI-componenten hebben hier het voordeel dat ze gemodelleerd kunnen worden naar de bestaande automatiseringsoplossing. Daardoor kunnen ze heel specifieke problematiek helpen oplossen. Aanschaf van dure pakketten en tools is voor zelfbouw bovendien niet nodig: er zijn legio open source bibliotheken beschikbaar die kosteloos experimenteren mogelijk maken. Hieronder lichten we globaal een aantal belangrijke stappen en overwegingen toe.

Probleemstelling: heb ik AI nodig?

Allereerst is het van belang om het probleem dat je wilt oplossen zo helder en specifiek mogelijk te formuleren. De vervolgstap is om grondig te controleren of AI technologie de beste manier is om het op te lossen. De werking van AI is anders (lees: veel meer datagedreven) dan ‘regulier’ programmeren en scripten, maar dit brengt ook heel andere complexiteit met zich mee. Data is het sleutelwoord. In hun online training ‘Machine Learning – Problem Framing’ hanteert Google de volgende vuistregel: voor het succesvol trainen van een simpel model zijn een paar duizend samples aan bruikbare data nodig, complexe modellen (zoals neurale netwerken) vragen al snel om honderdduizend samples, of meer. Heb je niet zoveel data beschikbaar, overweeg dan eerst een ‘klassieke’ rule-based oplossing.

Data selecteren

Beschikbaarheid van data is dus bepalend bij het implementeren van AI functionaliteit. In theorie hebben we binnen testautomatisering aan data doorgaans geen gebrek; denk maar aan alle test en systeemlogs, foutrapportages, testdata en evidence dat loskomt bij een gemiddelde testrun. De hamvraag om mee te beginnen is echter: hoe kunnen we data uit dergelijke bronnen zodanig met elkaar combineren dat het iets zegt over het probleem, en wijst in de richting van een oplossing? Hoe specifieker en smaller de scope van het probleem, des te groter is de kans dat de huidige AI technologie er een oplossing voor kan bieden: wie zijn testset op meerdere fronten wil ‘verslimmen’ zal voor elke specifieke taak een datastructuur met bijbehorend AI model moeten opzetten. Het bouwen van een ‘alwetende’, multitaskende AI testassistent ligt zeker in het kader van zelfbouw nog lang niet binnen handbereik.

Data voorbereiden voor het trainen

Na het in kaart brengen van relevante data is het zaak om deze samen te brengen in een dataframe – iets wat je kunt zien als een grote tabel, dat gelezen kan worden door een AI algoritme. Kolommen representeren hier de eigenschappen - features - van de data en de rijen zijn individuele samples. De 'Orchids'-dataset, het equivalent van 'Hello World' binnen AI/Machine Learning, is een leuke illustratie: het doel van deze AI functionaliteit is het kunnen bepalen van het soort orchidee, op basis van features zoals de lengte en breedte van het kelkblad, en lengte en breedte van het bloemblad.

De samples bevatten concrete afmetingen van de verschillende bladeren. Aangezien algoritmes 'denken' in cijfers, is het omzetten van allerlei soorten data (ook tekst en beeld) naar numerieke waarden een belangrijke stap in 'feature engineering', zoals het vormgeven van een dataframe ook wel wordt genoemd. Gelukkig hoeven we hiervoor het wiel niet opnieuw uit te vinden: nagenoeg alle data science-bibliotheken bevatten tools voor dergelijke conversies. Hetzelfde geldt voor de AI/Machine Learning-algoritmes zelf, ook deze kunnen in bibliotheken gedownload worden. Pakketten als scikit-learn voor Python en WEKA voor Java bevatten out-of-the-box algoritmes voor classificatie, regressie en clustering. Een framework als TensorFlow is multi-platform en biedt functionaliteit voor het meer geavanceerde deep-learning met neural networks - en dat allemaal open source. Wie het zichzelf echt gemakkelijk wil maken kan Anaconda (www.anaconda.com) overwegen: een one-stop pakket voor Python dat alle noodzakelijke bibliotheken en handige editors al inbegrepen heeft.

Model trainen

Met een dataframe beschikbaar en de benodigde tools geïnstalleerd kan het echte experimenteren beginnen: een algoritme kiezen en trainen, testen en bijstellen. Het kan niet vaak genoeg benadrukt worden hoe belangrijk de kwaliteit van de trainingsdata is: de kwaliteit van de data heeft procentueel een veel grotere invloed op de accuratesse van de het getrainde model dan het soort algoritme dat erachter ligt. Het loont om verschillende algoritmes te proberen, want ze hebben allemaal hun sterktes en zwaktes. Lukt het echter niet om het model accuraat genoeg te trainingen, ongeacht het algoritme? Grote kans dat het dan toch een onzorgvuldigheid in de trainingsdata betreft – en tijd om je feature engineering nog eens onder de loep te nemen.

Zelf AI kennis opdoen

Zoals bovenstaande verhandeling aan wil tonen krijgt de testautomatiseerder die met AI aan de slag wil hoe dan ook te maken data science - en dat is niet voor niets een vakgebied op zich. Data science tooling zoals eerder genoemde bibliotheken zijn open source, maar bevatten talloze mogelijkheden die het bestuderen waard zijn. Gelukkig zijn er online tegenwoordig uitstekende tutorials, lessen en cursussen te vinden die ook ons als testers en testautomatiseerders betrekkelijk snel up-to-speed kunnen krijgen in het werken met AI technologie – veelal kosteloos en goed behapbaar.

Met name de Machine Learning trainingen van Kaggle[9] en Google[10] zijn het overwegen waard als eerste stap: naast het technische aspect besteden deze trainingen ook aandacht aan het formuleren van cases voor AI en het voorbereiden van data. Bovendien draaien alle oefeningen online via je browser, dus het installeren van tools is niet noodzakelijk. Andere interessante bronnen en trainingen zijn terug te vinden in de bijlagen. De kennis en tools voor het zelf bouwen van AI componenten zijn dus goed beschikbaar: ben jij de creatieve tester die de nieuwste implementatie voor ons vakgebied gaat ontwerpen?

[9] Kaggle is een platform waar machine learning competities worden georganiseerd. Het is echter ook een toegankelijke community, waar nieuwkomers trainingsmateriaal kunnen vinden om later aan de competities deel te nemen. https://www.kaggle.com/learn/

[10] Google is graag een bekende naam in de AI wereld en stelt deze cursus aan iedereen beschikbaar: www.developers.google.com/machine-learning/crash-course

6. Een fictief, maar concreet voorbeeld

We nemen als voorbeeld een testobject dat bestaat uit een online invoersysteem (via GUI) en wat transactieverwerking op de achtergrond. De tester wil haar activiteiten ondersteunen met AI-testtooling.

Als eerste stap wordt de GUI getest. Hiervoor kan een reeds bestaand zelf-explorerende testtool gebruikt worden, zoals bijvoorbeeld Testar. De testtool doet diverse testen waarbij de kwaliteit van de GUI beoordeeld wordt. Bijvoorbeeld of knoppen logisch zijn en of er geen “broken links” zijn.

Vervolgens gebruikt de tester een zelf gebouwd op AI-gebaseerde testtool (gebruik makend van standaard machine learning algoritmes) om een analyse te doen van de productiedata om op basis daarvan synthetische testdata te genereren die elke relevante unieke situatie uit de productiedata precies een keer in zich heeft en daardoor dus een beperkte maar optimale set testdata oplevert.

Een volgende stap is om met een AI-gebaseerde testtool testscripts te genereren die de testdata kunnen invoeren in het testobject en vervolgens deze scripts uitvoeren en het resultaat beoordelen. Met name deze beoordeling is een complexe activiteit waarvoor heel veel kennis, inzicht en ervaring nodig is. Het lijkt waarschijnlijk dat de AI-testtooling vooralsnog alleen suggesties doet en dat de tester de uiteindelijke “pass / fail”-beslissing neemt.

Als laatste stap wordt het resultaat van de beoordeling verwerkt door een rapportagetool dat automatisch de informatie voor de verschillende groepen stakeholders in een voor hen optimale presentatievorm weergeeft.

Op het moment van schrijven (april 2019) zijn nog niet alle hierboven geschetste tools werkelijk beschikbaar. Maar op het moment dat deze werkgroep ontstond (maart 2017) konden wij nog geen enkele van deze tools vinden, dus de ontwikkelingen gaan enorm snel.

7. Toekomst van het testvak

AI en machine learning zijn volop in ontwikkeling. De techniek wordt steeds krachtiger en er verschijnen steeds meer aansprekende toepassingen. Dit zien we ook terug in het testvak. De AI testtools worden geavanceerder en frameworks om zelf iets te ontwikkelen worden toegankelijker.

Dat leidt tot de vraag wat de rol van de tester nog is zodra de AI toepassingen volledig zijn ontwikkeld binnen het testvak. Ofwel: ‘neemt AI onze baan als tester in?’ Hopelijk heeft de whitepaper tot dit punt duidelijk gemaakt, dat we voor dat laatste niet bang hoeven zijn.

Wel is duidelijk dat het testvak zal veranderen door de komst van AI, net zoals de huidige testautomatisering dit heeft gedaan. En dit zal ook gevolgen hebben voor de rol en de vaardigheden van de tester.

Hoe verandert het testvak

Omgaan met zelflerende AI testtools

Hoewel getrainde AI testtools ons veel werk uit handen zullen nemen, zullen we een oplossing moeten vinden voor het feit dat getrainde modellen zich niet 100% volgens verwachting zullen gedragen, in tegenstelling tot geprogrammeerde testtools. Machine learning leert immers van voorbeelden, in plaats van dat het vaste regels volgt. Dat betekent dat de testtools en onze eigen configuratie daarvan, zelf ook getest moeten worden. Per geval moet worden gekeken of de inspanning van het testen van de AI testtool opweegt tegen het voordeel dat de AI testtool brengt.

Bovendien moeten testresultaten op een andere manier worden geïnterpreteerd. Een traditioneel geprogrammeerde testtool loopt vast als een bepaald object niet wordt gevonden. Een AI testtool vindt waarschijnlijk een omweg om de test af te ronden en zal je melden dat ‘de test voor 87% geslaagd is.’

Keuze uit meerdere tools

De makers van tools hebben we gevraagd naar hun toekomstbeeld van testen en tools. We hebben ze gevraagd waar ze nu denken dat het naar toe gaat. De mensen die we gesproken hebben zijn er van overtuigd dat het gebruik van testtools met ai zal toenemen de komende tijd.

Trends zijn dat tools met ai naast traditionele tools gebruikt zullen gaan worden. Tools met ai zijn op dit moment gespecialiseerd voor één specifieke taak. In de toekomst zal het gebruikelijker worden om meerdere testtools naast en met elkaar te gebruiken en zo een palet aan tools te hebben dat de totale range aan testactiviteiten bestrijkt.

Een andere trend is dat kleinere tool leveranciers hun diensten samen aan gaan bieden. Bijvoorbeeld tools die de applicatie kunnen verkennen combineren met tools die goed zijn in slimme image recognition. En mogelijk ook dat een tool als Selenium IDE gebruikt kan gaan worden in combinatie met een of meer AI tools.

Het tegenovergestelde is specialisatie zoals je dat nu bij tools als test.ai tegen komt. Deze tool richt zich uitsluitend op webshops. Het doel is om op termijn een testservice voor webwinkels aan te bieden die volledig zelfstandig kan werken.
Er zijn tools met ai die zich specifiek op de app markt richten. Voor apps is het ook belangrijk om snel en op zoveel mogelijk platforms te kunnen draaien.

Bij grotere testframeworks begint men ook interesse voor het aanbieden van AI te krijgen. Functionize, Appvance en Eggplant zijn voorbeelden van Amerikaanse testplatforms die dat al langer doen en elk op een eigen manier.

Testen wordt creatiever

Door de komst van tools zoals Selenium, werden herhaalde testen weggehaald bij de tester en kon het handmatige testen zich meer richten op de creatieve testsituaties. Dit wordt versterkt als AI testtools nog meer standaard testen van ons overnemen.

Bovendien zal AI helpen om veel van de (technische) unittesten te schrijven en uit te voeren, zodat de tester zich meer kan richten op functionele en complexere non-functionele testen. Het handmatige, creatieve testen verschuift naar het verifiëren van de requirements en naar het leveren van waarde voor de organisatie.

Hoe verandert de rol van de tester

Het werk van een tester zal blijven bestaan. Wel zullen de werkzaamheden, die nodig zijn om inzicht in kwaliteit en risico's te geven, veranderen. Daarmee zullen ook de nodige kennis en vaardigheden veranderen. We zien de volgende ontwikkelingen.

Duidelijk formuleren van doelstellingen

Deze whitepaper heeft laten zien, dat een AI ons op allerlei punten kan ondersteunen bij het testproces. Er is zelfs een mogelijkheid dat de AI zelf op verkenning gaat. Dit betekent, dat we duidelijk moeten aangeven wat we eigenlijk verwachten van de applicatie en wat we graag getest willen hebben. Ofwel; meer aandacht voor requirements, acceptatiecriteria én testdoelen. Want in het meest extreme scenario zou de AI zonder enige sturen kunnen verkennen en zouden we maanden bezig zijn om de resultaten te beoordelen. Een goede set aan verwachtingen vooraf kan dit voorkomen.

Overigens betekent dit niet, dat we moeten stoppen met zelf de applicatie te verkennen via bijvoorbeeld exploratief testen. Het wordt misschien nog wel belangrijker; de menselijke

tester doet een goede verkenning van het speelveld, het grootschalige en herhaalde testen laten we aan de AI.

Managen van onzekerheid in het testproces

Zoals de eerste paragraaf van dit hoofdstuk liet zien; het toepassen van zelflerende AI (ofwel machine learning) zorgt ervoor, dat we geen absolute 'goed' of 'fout' uitslagen krijgen. Als tester zullen we zelf een interpretatie moeten doen van een zekerheidspercentage die de AI terug geeft. Dus als de AI een testgeval voor 87% geslaagd vindt, hoe rapporteren we dit en welke vervolgactie geven we hieraan. Dit zal per project anders zijn; het is dit mensenwerk om dit te bepalen, en soms waar nodig aanvullend handmatige testen te doen.

Ook de testmanager zal deze onzekerheid ervaren. Als de meeste testen voor minimaal 90% geslaagd zijn en enkele testen voor 75%, zijn we dan klaar? Dit is een interpretatie die de testmanager moet doen en komt mooi terug op het vorige thema; zorg dat er duidelijke verwachtingen vooraf zijn. Bovendien vraagt het om de vaardigheid om grote hoeveelheden cijfers te interpreteren. En dat brengt ons bij het volgende punt.

Kennis van statistiek

Het interpreteren van grote hoeveelheden uitslagen en zekerheidspercentages, vraagt van de tester en de testmanager dat er enige statistische kennis aanwezig is. Zelfs als kant-en-klare tools beweren dat het de interpretatie graag namens jou uitvoert, is het van belang om zélf een gefundeerd oordeel te hebben over alle cijfertjes die een AI teruggeeft. Moeten de tester en testmanager dan een part-time data scientist of data engineer worden? Dat is nog moeilijk te zeggen, maar wat basiskennis helpt je zeker op weg.

Betere kennis en begrip van testtools

Er is een zeer diverse hoeveelheid tools beschikbaar. Los van of ze 'echt AI' zijn, hebben ze allemaal een verschillende manier waarop ze waarde bijdragen, in verschillende omgevingen. De trend die nu al zichtbaar is, is dat een groot aantal tools aan elkaar wordt gekoppeld om de testdoelen te bereiken. Hier zullen AI-ondersteunde tools bij komen.

De verwachting is dat het geautomatiseerde gedeelte van het testproces groter wordt, ten opzichte van handmatige test- en testondersteunende activiteiten. Kennis van welke tools je kunt inzetten, in welke combinatie en vooral met welk *doel*, wordt daardoor nóg belangrijker.

Ofwel: er zal altijd een rol blijven voor iemand die over kwaliteit nadenkt, die kritische vragen stelt en die de communicatie over doelen en prioriteiten verzorgt. Om Jeremias Roessler[11] aan te halen; de kans dat we ooit automatisch software gaan *bouwen* is groter dan dat we deze volledig automatisch gaan *testen*. Want juist als een AI op basis van een model zelf een applicatie bouwt, is het van belang dat dit níet volgens datzelfde model getest wordt.

[11] https://www.sealights.io/webinars/when-will-ai-take-my-job-as-a-qa-manager/#fvp_WhenWillAITakeMyJobAsAQAManager

Testen ván AI

Deze whitepaper behandelt het testen mét AI, niet het testen ván applicaties met AI. Tot dit punt hebben we daarom het testen ván AI als onderwerp vermeden. Toch moeten we het hier kort benoemen, omdat dit impact heeft op het testen mét AI.

- Het zal ervoor zorgen dat data een meer centrale rol speelt in onze testtrajecten. Bij het trainen én testen van een klantapplicatie met AI zullen we heel bewust de gebruikte data moeten kiezen. De benodigde data van een testgeval zijn zo breed als het aantal variabelen waarop het model is getraind. We kunnen vervolgens AI gebruiken om te bepalen of we de juiste data hebben, of deze zelfs via AI genereren.
- Het accepteren van het eindresultaat zal sterk gebaseerd zijn op de hoeveelheid goede en foute voorspellingen die de AI-applicatie doet. Dit betekent dat statistiek een belangrijke rol in het testvak zal krijgen, naast het aantal geslaagde en gefaalde testgevallen die we tot nu toe als belangrijke metriek gebruikten.
- Meer sturen op het proces; in een project waar AI patronen herkent in data, is er een voortdurende interactie tussen de modelleurs die het model optimaliseren en mensen met inhoudelijke kennis, die de uitkomsten en uitzonderingen beoordelen. Het toepassen van AI smart-dashboards kunnen helpen deze complexiteit inzichtelijk te maken en kunnen helpen de juiste kwaliteitscontroles in te plannen.
- Verbreden van de kijk op kwaliteit: we zullen als tester ook ethiek en moraliteit moeten meenemen. Bovendien moeten we ongeschreven / common sense requirements in de gaten houden, want een AI kent geen context.

Deze whitepaper is slechts het begin

Terwijl we als werkgroep met deze whitepaper bezig waren, ontwikkelde het testen met AI zich in hoog tempo verder. Sommige delen hebben we vijf keer herschreven. Deze whitepaper beschrijft de ‘state of the art’, maar is dus absoluut een momentopname.

De werkgroep gaat enthousiast verder met het volgen van dit vakgebied. Ook het testen ván AI en de combinatie met het testen mét AI, zullen via de TestNet werkgroep aandacht blijven krijgen.

Wil je met ons meedoen bij het volgen én vormgeven van deze nieuwe ontwikkelingen? Je bent van harte welkom! Neem contact op met werkgroepen@testnet.org en we gaan graag samen met jou aan de slag.

Bijlage 1: bronnen

AI en testen algemeen

- Jason Arbon heeft veel verschillende artikelen geschreven, webinars en interviews gegeven.
  https://www.centercode.com/blog/2019/04/ai-and-the-future-of-testing
  https://www.linkedin.com/pulse/links-ai-curious-jason-arbon
  https://huddle.eurostarsoftwaretesting.com/resources/artificial-intelligence/ai-will-soon-test-everything/

- Test talks met Joe Colantonio
  Veel interviews over testen met ai. Een voorbeeld:
  https://www.joecolantonio.com/testtalks/178-third-wave-test-automation-joe-colantonio/ uit 2017

- Boek over zowel testen mét AI als testen ván AI, onder andere over evolutie naar forecasting op basis van informatie uit testen en monitoren:
  “Testing in the digital age; AI makes the difference”, 2018, Tom van de Ven, Rik Marselis, Humayun Shaukat. ISBN: 978 90 75414 87 5

Test tools

- Een aantal bestaande tools
  Zie vooral de links in bijlage 2
  https://www.ministryoftesting.com/dojo/lessons/ai-in-gui-based-software-testing?utm_source=linkedin.com&utm_medium=social&utm_campaign=artificial-intelligence-ai-and-machine

- Zelfbouw
  Machine learning library Weka: Getting Started with Weka - Machine Learning Recipes #10

Opinie/Visie

- Computable
  https://www.computable.nl/artikel/nieuws/overheid/6630922/250449/kabinet-komt-voor-de-zomer-met-ai-actieplan.html
- EuroSTAR
  https://huddle.eurostarsoftwaretesting.com/title-will-ai-kill-software-testing-heres-wont/
- NRC (Robbert Dijkgraaf)
  https://www.nrc.nl/nieuws/2018/10/19/gevangen-in-onze-verbeelding-a2672745
- Salves
  https://www.testforward.nl/newsroom/#artificial-intelligence

- Sogeti
  https://labs.sogeti.com/to-test-is-human
- TED
  Grady Booch: Don't fear superintelligent AI
- Tegenlicht
  https://www.facebook.com/37586539901/posts/10156280444544902/
  https://www.facebook.com/37586539901/posts/10156289342059902/
- TorontoTech
  Chris McKillop of turalt presents Email, empathy, ethics & AI

Bijlage 2: overzicht van tools

Hieronder staan in twee tabellen informatie over tools.
De eerste tabel geeft een overzicht van wat de tool voor de gebruiker kan doen en wat niet.
De tweede tabel geeft weer wat het webadres is en voor welk soort applicatie/framework de tool geschikt is.

Tabel 1

Naam van tool	**wat doet de tool voor de gebruiker?**	**wat moet de gebruiker zelf doen?**
applitools	Visuele validatie, slim visuele verschillen herkennen, objecten herkennen. Nieuw is applitools met selenium IDE.	Testen maken met recordings. Valideren van afwijkingen met de baseline. Testen toevoegen aan bestaande testen met andere tooling zoals Selenium.
appvance	Test framework met functionele, load en performance tools. De tool genereert testgevallen voor geautomatiseerde testen op basis van recordings uit productie of vanuit logfiles. Analytics op dashboards.	Vastleggen van de baseline en valideren van verschillen.
functionize	Test framework met functionele,security, load en performance tools. Functionize gebruikt gegevens van productie logs en functionele testen. Deze data bewerken ze in een tool (adaptive event analysis system). Deze tool maakt de data klaar voor modellen. De modellen pushen ze naar de Elastic Cloud omgeving, waar de testen gerund worden op alle mogelijke platforms. Ze zijn in staat om statische en dynamische scenario's te beoordelen met ai. Analytics op dashboards.	Starten en managen van testsets en runs. Toevoegen van testen. Valideren van gevonden verschillen door de tool.
eggplant	Test framework met functionele testen. Als de applicatie gemapt is in de tool kan deze al bestaande functionele testen hier op draaien (gemaakt met Eggplant functional. Kan zelf zijn weg zoeken en testen	Gebruiker moet de applicatie in beeld brengen. Beoordelen van testruns. Het zelf toevoegen van testen is mogelijk en ook testen een hogere rating geven zodat de kans dat ze gerund worden

	genereren. Kiest testgevallen/scenarios. Analytics op dashboards.	groter is.
testim	De ai tool analyseert alle DOM-objecten van een pagina en extraheert de objecten en de eigenschappen ervan. Op basis hiervan wordt de beste locatiestrategie om een bepaald element te lokaliseren gemaakt. Dit maakt de elementen vindbaarder en testen stabieler en, blijft de test lopen. Testim houdt scores bij van elementen in de GUI. Als deze objecten niet of weinig veranderen dan krijgen ze een andere score dan objecten die veel veranderen. (testen die falen/slagen) Analytics op dashboards.	Zelf testen maken.
mabl	Mabl verzamelt meer dan één identifier voor elke test stap. Ze zeggen dat ze daarmee makkelijker om kunnen gaan met kleine wijzigingen in de applicatie zonder dat de test breekt. De test geeft dat aan door auto-healed te tonen, die je ook weer terug kunt draaien als het resultaat niet klopt. Bij elke stap zijn verschillen zichtbaar door screenshots. Machine learning haalt daar de verschillen uit in de statische delen van de applicatie	Gebruiker moet zelf testen maken met record playback, assertions toevoegen etc.
retest	Alle gemaakte testen opnieuw runnen met de tool en met de gekozen intensiteit. (Monkey test.) Op basis van deze test wordt automatisch een rapport gegenereerd. De tool kijkt ook naar zoveel mogelijk eigenschappen in de DOM en probeert slim te matchen met en zonder ai.	Gebruiker neemt zelf testen op met de visuele test tool.
sofy	Noemt zich testassistent. Testen van apps, loopt zelf door de app heen. Afwijkingen kunnen met een druk op de knop naar Slack of jira gekopieerd worden met een gegenereerde test en uitgeschreven reproductiepad.	Gebruiker beoordeelt de test runs. Maakt bugs aan met de tool..

	Zoekt naar problemen met bekende mobile frameworks op stackoverflow en rapporteert daarover met evt workarounds.	
testrigor	De tool gaat zelfstandig aan het werk en gaat door de pagina's heen en herkent input velden en drukt op knoppen. Na een nieuwe deployment van de applicatie kan een hertest gedaan worden.	Het beoordelen van verschillen en het aanpassen van de gegenereerde testen.
test.ai	Tool voor webshops. Loopt de paden af in de app en maakt daar testen van. Checkt op visuele wijzigingen en beoordeelt deze. Zorgt dat testen door blijven lopen en niet breken bij een klein verschil.	Beoordelen van verschillen in test runs. Testen toevoegen.
apptest.ai	Testen van Android apps. Na het uploaden van de.apk crawlt de tool door de applicatie. Kan op mobile devices naar keuze gedraaid worden, echte of emulators. De software levert in vijf categorieën de resultaten: ● Summary is een samenvatting van device info en testresultaten; ● Activity Map geeft een uitgebreide visualisatie van alle pagina met links en connecties; ● Screen is een snapshot van alle app-pagina met highlight op actieknop en andere beschikbaar knop(pen).; ● Performance geeft een graaf van CPU en geheugen tijdens testen; ● Log is een lijst van tag, bericht en bug mogelijkheden.	Beoordelen van verschillen in testruns/devices.
testar	Het is een tool waarmee je crashes, hangers en fouten kunt opsporen. Je kunt hem zelf verder uitbreiden door checks toe te voegen en te programmeren. Het is de	Starten van de tool. Checken welke fouten zijn opgespoord. De functionaliteit is nu nog beperkt.

	bedoeling dat de tool verder ontwikkeld wordt	
diffblue	Werkt op de programmeercode in Java. Scant de code en maakt unit testen aan waar ze het meest nodig zijn.	Gebruiker is ontwikkelaar. Deze moet de testen beoordelen en runnen.
sealights	Analyse platform voor coding en testen. Filosofie: zuinig testen, risico's afdekken. Dashboards en overzichten geven inzicht in welke testen er gedaan moeten worden, zowel automatisch als handmatig. Tool bevat ook een commit-assistant die inchecken van code zonder testen verbiedt.	Tool geeft inzicht. Van de tester wordt verwacht dat handmatige testen op basis van de wijzigingen in programmeercode gebeurt, zodat de tool dat kan registreren.

Tabel 2

tool website	tool geschikt voor	webservice/lokaal
applitools.com	als add-on op bestaande tools, werkt op veel verschillende systemen en ontwikkelomgevingen Nieuw is Applitools op IDE.	sdk installatie communicatie met webservice Om het werkend te krijgen is wat codering nodig, voorbeelden op website.
appvance.ai VS	webapplicaties en apps	webservice
functionize.com VS	webapplicaties en apps	webservice
eggplant.com (voorheen testplant) VS	Diverse delen van de tool zijn gericht op management,, app s, network, testing en generen van testen.	lokale installatie op server
testim.io VS	Chrome extension record playback. Webapplicaties en apps (beta)	Webservice Op dit moment moet je opnieuw beginnen met het maken van testen als je Testim wil gebruiken,. Er wordt gewerkt aan integratie met andere bestaande tools zoals Selenium.
mabl.com VS	Chrome-extension Record playback .	Webservice
retest.de	Applicaties geschreven in Java.	Lokale installatie met lokale ai.

Duitsland	Er wordt gewerkt aan een tool voor (web) apps.	NB een Ai is te downloaden naast een voorbeeld tool.
sofy.ai bedrijf Quantyzd VS	Sofy richt zich op apps op IoS en Android. Kan van	De app wordt naar de cloud geupload en sofy gaat aan de slag.
testrigor.com VS	Webapplicaties en apps Bedoeld voor kleine applicaties en apps.	webservice
test.ai VS	Webshop apps en applicaties	webservice
apptest.ai Zuid Korea	Apps Android	webservice, uploaden van je app naar de apptest.ai cloud
testar.org NL/SP	De tool werkt op Windows, MacOs en Android. De tool gebruikt het OS en vraagt widget trees aan de api van het OS uit van een doelapplicatie.	Lokale installatie met download
sealights.io Israel	Plugin op de ci tool doet dynamische analyse van de build, testen etc.	webservice
diffblue.com Qxford GB	Op https://github.com/diffblue is de tool te vinden. Er wordt aan het ondersteunen van andere programmeertalen gewerkt.	