Adviesrapport Reinforcement Learning

Oplossing voor doorstroming van het verkeer op kruispunten met Reinforcement Learning

Studenten: Carlo Keizer, Tom Blokland
Datum: 11-08-2021

Inhoud

Inleiding

Voor het vak Adaptive Systems hebben we ons verdiept in Reinforcement Learning. Daarbij hebben we een Q-Network (QN) en een Deep Q-Network (DQN) op een aantal cases toegepast. Er zijn twee notebooks opgesteld door Bas Niesink van InfoSupport. Deze notebooks kunnen met Google Colab uitgevoerd worden. De notebooks zijn gelinkt aan de OpenAI gym van Google, met daarin diverse omgevingen van games bedoeld voor AI-projecten.

Het adviesrapport is gebaseerd op deze twee notebooks over Reinforcement Learning die uitgewerkt zijn en waarbij Reinforcement Learning modellen geoptimaliseerd zijn. Daarvoor is er een kort literatuuronderzoek gedaan naar de invloed van de parameters op de performance van het model. Tijdens en rondom de contacturen van het vak Adaptive Systems hebben wij de tijd gekregen om te experimenteren met de settings van de RL-modellen, met als doel het kennismaken met de omgeving, het implementeren van functies, het aanpassen van parameters en het begrijpen hoe dit soort algoritmes zich gedragen.

Aan de hand van de kennis die we hebben opgedaan geven we advies over het gebruik van Reinforcement Learning voor het regelen van de verkeerslichten bij kruispunten.
Huidige verkeerslichten werken met sensoren. Als een auto op een bepaalde rijbaan voor een verkeerslicht staat kan een sensor dit waarnemen en dit gebruiken voor een verkeerslichtensysteem. Nu is de vraag of het toepasselijk en handig is om Reinforcement Learning te gebruiken voor een verkeerslichtenregeling en of het handiger is om een DQN of een QN te gebruiken.

Voor de case Space Invadors is er een DQN toegepast en een aantal experimenten uitgevoerd. Het verloop van de experimenten met verschillende settings is in Resultaten beschreven. Vervolgens worden de voor- en nadelen gegeven van het inzetten van Reinforcement Learning. Daarna worden de ethische en praktische overwegingen besproken waarbij risico’s aan het licht komen. De specifieke stappen voor het implementeren van een Reinforcement Learning model worden in de Planning toegelicht en, tot slotte, wordt er een conclusie getrokken met het advies over het gebruik van Reinforcement Learning voor het regelen van verkeerslichten.

Resultaten

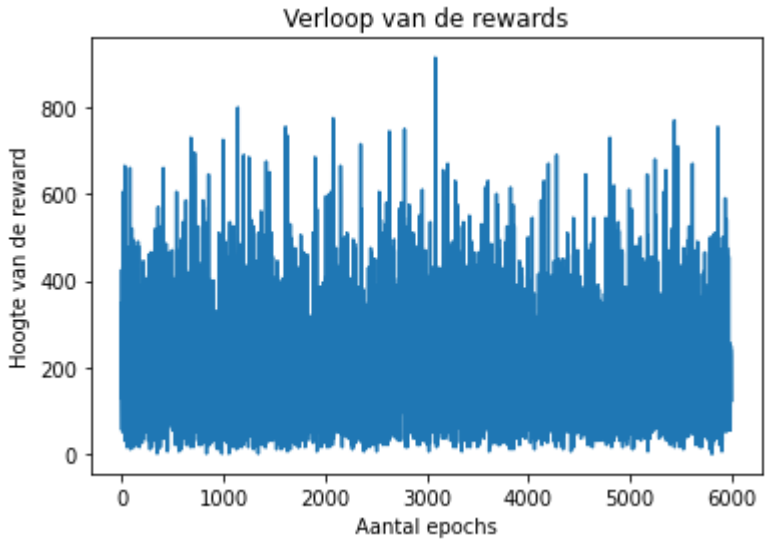

Figure 1: rewards during 10.000 episodes

In het tweede notebook hebben wij voor een game een Reinforcement Learning model getraind. Tijdens het trainen kwamen wij erachter dat ons model nauwelijks tot niet convergeert met verschillende parameters bij 1000 epochs. Om zeker te weten of wij iets verkeerd deden of dat wij ons model niet lang genoeg lieten trainen, hebben wij het model een keer voor 6000 epochs getraind met de standaard parameters. De resultaten hiervan zijn te zien in de grafiek hierboven. Ook in deze grafiek is aan de reward te zien dat het model niet convergeert. Hieruit blijkt dus dat het belangrijker is om de juiste parameters te kiezen, dan om het langer te trainen en te hopen dat het model convergeert.

Een ander vermoeden wat wij hebben is dat ons model te lang blijft exploreren. Oftewel het model blijft willekeurige acties kiezen, in plaats van dat er in verloop van het trainingsproces over wordt gegaan naar de beste acties. Hierdoor is het niet mogelijk voor het model om te convergeren.

Een ander vermoeden dat we hebben is dat het model de beste actie al kiest zonder dat het model iets geleerd heeft. Oftewel het model exploiteert te snel. Hierdoor kiest het model een actie waarvan de Q-waarde niet goed is berekend, waardoor het model nooit goed zal convergeren.

De actie die uit deze resultaten af te leiden is en die wij uitgevoerd hebben, is het aanpassen van de Epsilon en kijken hoe de reward daarop reageert.

Op basis van een paper over het fine tunen van een DQN is de Epsilon in dit project verlaagd [1]. Zij tonen aan dat in een vergelijkbare context de gemiddelde Epsilon van 0.3 optimaal is voor de reward. De huidige Epsilon loopt van 0.1 tot 1.0 dat misschien iets te hoog is, en daarom kan het model te “greedy” zijn. Daarom is het experiment opnieuw

gerund waarbij de Epsilon van 0.1 tot 0.6 loopt binnen 750 runs. Dat betekent dus dat de laatste 250 runs de Epsilon altijd op 0.6 blijft.

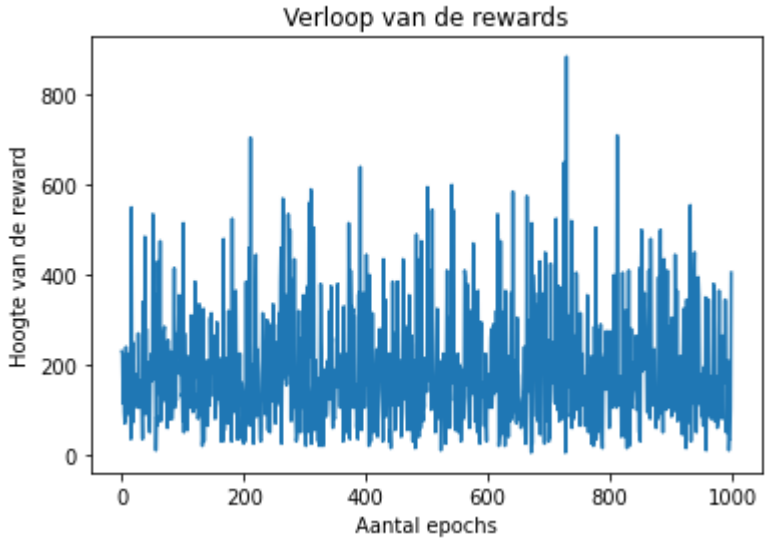

Figure 2: rewards during 1000 episodes with lower Epsilon

Het blijkt dat in de eerste 1000 episodes het model niet beter wordt bij een lagere Epsilon. Dit zou te maken kunnen hebben met het feit de Epsilon in de paper een niet-lineaire functie is terwijl wij een lineaire functie hebben gebruikt in ons onderzoek.

Voor- en nadelen

Er zijn een aantal voordelen en nadelen voor het gebruik van Reinforcement Learning boven andere algoritmen. Een aantal daarvan worden verder toegelicht in de volgende hoofdstukken.

Voordelen

- Ten opzichte van de huidige oplossingen, is een Reinforcement Learning model toepasselijker voor complexe problemen. Reinforcement Learning is toepasselijker omdat deze techniek leidt tot het ideale gedrag van de Agent(s) binnen een specifieke context om zo de performance te maximaliseren. De reward kan daarbij duidelijk en specifiek gedefinieerd worden.

- Ook al is het model een black box model en is het moeilijk te bewijzen of het goed werkt, kan dit ook als voordeel worden gezien. Er hoeft niet precies bewezen te worden hoe het model werkt om een hoge reward te krijgen. Dus er hoeft geen

statistische modellen te maken van real life data. In ethische overwegingen wordt uitgelegd waarom een black box model ook een risico is.

- Reinforcement Learning heeft geen grote gelabelde datasets nodig. Er hoeft dus geen input en output data verzameld worden van het verkeer op kruispunten omdat de omgeving zo gebouwd is dat het model zelf bepaalde acties exploreert en een reward geeft voor elk beschikbare state. Wel moeten data verzameld worden om de omgeving van de simulatie realistisch te maken.

Nadelen

- Ten opzichte van huidige oplossingen, leert een Reinforcement Learning model van het gedrag van het verkeer. Maar het gedrag van de mens is niet altijd constant. De verschillende tijden, dagen, seizoenen, vakantie's en het weer hebben invloed op het gedrag van de mens. Het kan dus voorkomen dat het gedrag van de mens tijdens het trainen van het model anders is dan wanneer het model in gebruik wordt genomen. Dit kan ervoor zorgen dat model niet altijd optimaal werkt. Dit probleem is te voorkomen om het model voor een langere tijd te trainen, zodat het model de fluctuatie van het gedrag van de mens meeneemt en het model dus optimaal werkt op elk soort gedrag van de mens.

- Een Reinforcement Learning model leert tijdens een trainingsproces heel goed te anticiperen op zijn omgeving. Maar als er in zijn omgeving een verandering plaatsvindt, kan het model niet goed meer functioneren. Dit kan bijvoorbeeld voor komen wanneer er bijvoorbeeld een rijbaan met een verkeerslicht is bijgekomen. Dit zorgt ervoor dat het aantal states gewijzigd wordt waardoor het model niet goed meer werkt. Het gevolg hiervan is dat het model opnieuw getraind moet worden. Dit probleem is niet te voorkomen en hier zal van tevoren rekening mee gehouden moeten worden.

- Het ontwikkelen van een reinforcement model kost veel tijd ten opzichte van de huidige oplossingen. Er zijn veel verschillende parameters die aangepast kunnen worden en de parameters hebben ook invloed op elkaar. Dit resulteert erin dat wanneer je een reinforcement model wilt ontwikkelen, dat het model vaak met veel verschillende parameters uitgevoerd moet worden. Wanneer het om een simpel probleem gaat is een Reinforcement model hier redelijk snel mee klaar. Echter, wanneer het om een complex probleem gaat met meer states, zal het model hier langer over doen. De combinatie van het model veelvuldig uitvoeren met verschillende parameters en de lange trainingstijd van het model, zal er dus in resulteren dat het ontwikkelen van een reinforcement model veel tijd gaat kosten.

- Het nadeel dat hierboven beschreven is komt dus voor wanneer het model veel states heeft. Om dit probleem zoveel mogelijk te voorkomen, zou je het environment zo simpel mogelijk moeten houden. Dit zorgt ervoor dat de trainingstijd van het Reinforcement Learning model afneemt.

- Het is niet te bewijzen of het model goed werkt of niet. (Er zijn vuistregels voor Parameters maar niet vaste regels).

- Data waarover het model een beslissing maakt is alleen afkomstig van een afgebakend gebied. Files die ver achter de stoplichten voorkomen worden hierdoor niet meegenomen. Het kan dus zijn dat er belangrijke gevolgen en situaties achter de schermen voorkomen waardoor de meerwaarde van Reinforcement Learning teniet wordt gedaan.

Ethische overwegingen

- Om een Reinforcement model te ontwikkelen dat de doorstroming verbetert is echte data nodig van de drukte van een kruispunt. Als hier geen online bronnen van zijn dan zal er met behulp van camera's of sensoren bijgehouden moeten worden hoeveel auto's het kruispunt passeren. Wanneer dit met camera's wordt gedaan zijn de kentekens en gezichten van mensen ook zichtbaar. Omwille van de privacy moeten de gezichten van de mensen en de kentekenplaten onherkenbaar gemaakt worden.

- Bij een Reinforcement model kan niet bewezen worden waarom er gekozen wordt voor een bepaalde actie. Er is dus geen zekerheid over hoe het model anticipeert in bepaalde situaties. Overheden moeten dit wel weten om een contract te kunnen afsluiten bij een verzekering. Aangezien een verzekering geen schade zal dekken die veroorzaakt is door een systeem waarvan de werking niet transparant is.

- Gaan mensen eerder met de auto wanneer het rustiger op de weg is of als de doorstroming beter is? Wanneer het antwoord op deze vraag "ja" is, dan is het gevolg van de implementatie van deze oplossing dat er meer mensen voor kiezen om met de auto te gaan. Dan is er ook meer uitstoot van uitlaatgassen dat weer slecht is voor het milieu.

Praktische overwegingen

Een aantal praktische overwegingen zijn bepaald op basis van het onderzoek.

Reward/Penalty

Het is lastig om een goede reward te definiëren, vaak is het zo dat een Reinforcement model onverwacht gedrag gaat vertonen, maar het model blijft zich wel aan de afgesproken regels houden. Bijvoorbeeld wanneer het model een reward krijgt voor elke auto die een groen licht passeert. Dan zou het kunnen gebeuren dat er op het drukste moment van de dag, het drukste stoplicht altijd op groen komt te staan. In theorie doet het model het dan perfect, want er worden dan zoveel mogelijk auto's doorgelaten. Maar in de praktijk blijkt dan dat er nog altijd auto's zijn die helemaal geen groen krijgen. Een oplossing hiervoor zou kunnen

zijn om het model een steeds hogere penalty te geven afhankelijk van hoe lang er al een auto voor een rood licht staat te wachten. Hierdoor zou de penalty op een gegeven moment sterker moeten gaan meetellen dan de rewards, waardoor het model de keuze moet maken om de auto's die het langste wachten eerst door te laten rijden.

Data

Om het model te kunnen trainen is er realistische data nodig. Deze data worden verkregen door middel van sensoren op een kruispunt te plaatsen. Deze sensoren moeten per rijbaan bijhouden hoeveel auto's er passeren. Met behulp van deze data kan een computersimulatie gebouwd worden die een realistische weergave kan genereren van de doorstroming van een kruispunt. Vervolgens kan het Reinforcement Learning model data uit deze computersimulatie extraheren om het model mee te trainen. Deze data zien er als volgt uit. Er zal per rijbaan/verkeerslicht bijgehouden worden hoeveel auto's er voor het stoplicht staan te wachten met een bepaald maximum. Een state zou dan gezien kunnen worden als een momentopname waarbij voor elk verkeerslicht de lengte van de wachtrij wordt gegeven. Met de volgende formule kan het aantal mogelijk states berekend worden.

$$amount\ of\ states\ =\ maximum\ amount\ of\ cars\ on\ one\ lane^{\ amount\ of\ lanes}$$

Als we hier uitgaan van een kruispunt waar je vanuit elke richting alle richtingen op kan gaan, dus 3 x 4 stoplichten, en met een maximum wachtrij van 15 dan komt hier het volgende uit 15^12 = 1.3 x 10^14. Dit zijn erg veel states, waardoor het trainen van het model erg veel tijd gaat kosten. We zouden een wachtrij kunnen opdelen in chunks/bins van 4. Als we dit doen komt hier het volgende uit 4^12 = 16.777.216. Dit zijn al een stuk minder states en dit zou tot een sneller trainingsproces moeten leiden. Een praktische overweging zou dus zijn om de lengte van de wachtrij op de delen in 4 bins.

Om het model toe te passen moet er data verzameld worden om de huidige state en omgeving vast te stellen.

DQN en state space

Als data verzameld worden met sensoren die meten hoe de omgeving eruitziet, dan is het misschien beter om met een Network de realiteit te benaderen. De states zijn dan zo complex en zoveel dat de reward voor elke state moeilijk berekend kan worden.

Bij een DQN is het nodig, zo niet verstandig, om limieten erg globaal in te stellen (bijvoorbeeld rijen in bins {1,2,3}). Een voorbeeld voor het gebruik van bins van 12 voor aantal auto's per rijbaan, en bins van 12 bij aantal rijbanen kom je al uit op 1000.000.000.000 states.

Dynamische omgeving

Als het model getraind is op een kruispunt, dan kan het model het minder goed doen op een ander kruispunt.

De situatie kan dus verschillen bij verschillende soorten kruispunten, maar ook op verschillende tijdstippen, dagen, seizoenen, vakantie's en weersomstandigheden.

Planning

Voor het implementeren van Reinforcement Learning is het handig om de planning overzichtelijk te maken. In Planning wordt besproken welke instellingen gebruikt moeten worden voor het trainen van het model, hoe het in praktijk uitgevoerd moet worden en welke taken en handelingen hieraan verbonden zijn.

Settings voor implementatie Reinforcement Learning in praktijk

Door de grootte van de state space wordt dus geadviseerd om een DQN te gebruiken. Om het model toe te passen moeten een aantal taken uitgevoerd worden om dit zo goed mogelijk te doen.

Voor het project met space invaders was de state space 80x105 = 8400. Dit had een learning time van ongeveer 1 uur. Om het model te trainen en eventueel aan te passen adviseren wij de state space niet ver boven 8400 te initialiseren.
Er is een overzicht gemaakt van hoeveel bins er gemaakt moeten worden om onder de 8400 states te blijven. De formule hiervoor is:

$$aantal\ bins\ =\ floor\ (\ \sqrt[aantal\ rijstroken]{max\ bin\ threshold}\ )$$

$$where\ max\ bin\ threshold\ =\ 8400$$

Stel dat de maximale aantal auto's op een rijstrook 10 is, dan kunnen de bins als volgt ingesteld worden om de state space te verkleinen.

Tabel 1: max possible bins and state space given amount of lanes

Aantal rijstroken	Aantal states	Max aantal bins	Nieuwe aantal states	Max wachtrij	Bin threshold
1	12	Nan		12	8400
2	144	91	8281		
3	1728	20	8000		
4	20736	9	6561		

5	248832	6	7776		
6	2985984	4	4096		
7	35831808	3	2187		
8	429981696	3	6561		
9	5159780352	2	512		
10	6191736422 4	2	1024		
11	7430083706 88	2	2048		
12	8916100448 256	2	4096		
13	1069932053 79072	2	8192		
14	1,28392E+15	1	1		
15	1,5407E+16	1	1		
16	1,84884E+17	1	1		

In een paper [2] waar een een soortgelijk project is gedaan met dezelfde package Gym worden de gebruikte parameters voorgelegd. De parameters voor het model zouden volgens deze default waarden moeten worden geïnitialiseerd.

Tabel 2: parameter settings used to train DQN

Parameter	**Value**
Reply memory size	750
Mini-batch size	32
The size of most recent frames	4
Discount factor	0.95
Update frequency for	10000

target network	
Learning rate	0.00025
Initial -greedy policy value	0.6
Final -greedy polci value	0.1
Gradient momentum	0.95
Squared gradient momentum	0.95
Memory size	400000

De memory size kan wel hoger gekozen worden als gebruik gemaakt wordt van sensoren in plaats van images. De omgeving is dan kleiner en er kan dan meer opgeslagen worden.

Om de simulatie in praktijk goed te laten werken moet data verzameld worden en toegepast worden in de simulatie. Variabelen als snelheid van de auto's, reactietijd en drukte heeft veel invloed op het environment. De conclusie uit de simulatie is meerzeggend over de werkelijkheid als de simulatie ook op de werkelijkheid lijkt.

Daarnaast zal het model periodiek getraind moeten worden als de omgeving veranderd. Als er bijvoorbeeld nieuwe rijstroken worden aangelegd zullen de states veranderen en moet het model dus opnieuw getraind worden.

De dynamische omgeving heeft dus invloed op het model. Omdat het model anders werkt op een andere omgeving moet het model ook gefinetuned en getraind worden op verschillende kruispunten.

Omdat er geen settings gevonden zijn om het model te convergeren kan er geen advies geven worden over mogelijke parameters.

Een laatste, maar alsnog een belangrijke overweging is het fine tunen. Het model moet niet te veel gefinetuned worden, anders beperk je het model tot een specifieke omgeving. Als het model voor meerdere diverse kruispunten moet werken dan is een algemener model beter [2].

Benodigde taken en handelingen

Een aantal concrete stappen die ondernomen moeten worden om Reinforcement Learning te implementeren voor de doorstroom van het verkeer bij kruispunten zijn hieronder opgesomd.

1) De gemeente om toestemming vragen voor het plaatsen van sensoren bij kruispunten.
2) Sensoren plaatsen bij kruispunten om realistische data te verkrijgen en te gebruiken om de simulatie realistisch te maken. Voor het verzamelen van realistische gegevens moet

eenmalig een aantal ambtenaren ingezet worden om de sensoren te installeren. Een aantal variabelen die gemeten moeten worden zijn:
   a) Drukte per dagdeel, dag, en seizoen.
   b) Snelheid van auto's.
   c) Reactietijd.
   d) Hoe vaak mensen zich niet aan de regels houden, of onverwachte acties maken.
3) Het model trainen op een groep heterogene kruispunten waarbij er verschillende modellen zijn voor verschillende kruispunten.
   a) Uitproberen met de parameters en de bin's die hierboven beschreven zijn.
   b) Verbeteren door de parameters te veranderen.
4) Sensoren plaatsen voor het toepassen van het nieuwe systeem met Reinforcement Learning. Ook hier zijn een aantal ambtenaren voor nodig.
5) Veranderingen van wegen en veranderingen van de drukte in de gaten houden. De sensoren die gebruikt worden voor het verkrijgen van de input voor het model kunnen gebruikt worden om de drukte te monitoren. Bij grote veranderingen zal het model of opnieuw getraind moeten worden op het desbetreffende kruispunt, of moet het model vervangen worden door een model dat al getraind is op een soortgelijk kruispunt.

Conclusie

Met de notebooks die gebruikt zijn om QN en DQN te gebruiken voor Reinforcement Learning blijkt dat QN wel werkt, gegeven dat het een simpele omgeving betreft, en een DQN niet werkt met de parameters die uitgeprobeerd zijn. Er is geprobeerd om de aantal episodes te verhogen om een hogere reward te verkrijgen, maar de reward bleef even laag en even inconsistent. Op basis van vooronderzoek over het effect van de Epsilon op de reward bij DQN's [1] is geprobeerd de Epsilon te verlagen. Dit verhoogde de rewards niet binnen de eerste 1000 episodes. Een mogelijke oorzaak is dat er een lineaire Epsilon decay wordt gebruikt, en in de paper een niet-lineaire Epsilon decay wordt gebruikt.

Reinforcement Learning heeft veel potentie voor complexe situaties waarin er weinig gelabelde data beschikbaar is. Echter werkt het minder goed in dynamische omgevingen, bij situaties waarin het minimum aantal states erg hoog is of bij situaties waarin zekerheid over resultaten prioriteit heeft.

Een ethisch besluit dat genomen moet worden, is dat er met een verhoogde doorstroming ook meer mensen zullen kiezen om met de auto te reizen. Dit zorgt voor meer auto's op de weg dat leidt tot meer uitlaatgassen.

In praktijk zijn er een aantal zaken om te overwegen. Een eis bij Reinforcement Learning is dat er een duidelijke en "eenzijdige" reward gedefinieerd moet worden. Dan is het belangrijk om in de reward rekening te houden met meerdere factoren, zoals dat de doorstromingen gelijk verdeeld zijn.

De data die worden gebruikt om het model te trainen kunnen worden verkregen uit een computersimulatie, of uit een echt kruispunt door middel van sensoren. Een state van het

model kan hierbij gezien worden als een moment opname van het kruispunt. Daarnaast moeten data verzameld worden door middel van sensoren om de simulatie realistisch te maken.

Ook is het belangrijk om de states te verminderen. Bij een kruispunt met maximaal 15 auto's per rijbaan en 12 rijbanen (4x3) zijn de aantal states zodanig hoog dat het te lang duurt om het model te runnen. Omdat de state space zo groot is, wordt geadviseerd om DQN te gebruiken om zo een Neural Network de realiteit te benaderen in plaats van een QN te gebruiken. Ook is dit de reden waarom er, voor nu, alleen gekeken wordt naar het verkeer van auto's en niet naar het verkeer van fietsers. Als er ook nog rekening gehouden wordt met het verkeer van fietsers wordt de state space zodanig hoog dat de trainingstijd het onmogelijk maakt om het model meerdere keren te runnen en te verbeteren.

Om de state space te verminderen is er in "Planning" in een tabel de maximale aantal bins genoteerd om de state space onder 8400 te houden. 8400 is de state space dat binnen dit project gebruikt is binnen de casus "space invaders" en wordt als limiet gedefinieerd. Hoe meer bins, hoe hogere potentie het model heeft om de reward te optimaliseren. Maar er moet dus een afbakening gemaakt worden om het in praktijk toe te kunnen passen.

Ook moet het model goed om kunnen gaan met dynamische situaties. Als er nieuwe wegen worden aangelegd of als er iets veranderd in de omgeving zal het model opnieuw getraind moeten worden. Ook zal het model waarschijnlijk slechter presteren op kruispunten die net iets anders zijn. Het model is dus niet geschikt voor elk kruispunt als het op één kruispunt is getraind. Er moet dus voor elk kruispunt een apart model getraind worden.

Een manier om met dynamische situaties om te gaan is het model niet te veel finetunen. Dat resultaat namelijk in een model dat beperkt is tot een specifiek kruispunt terwijl een algemener model beter werkt [1].

Tot slot zijn wij tot de conclusie gekomen dat een Reinforcement Learning model veel voordelen biedt voor de case over de doorstroming van het verkeer. Echter kan er moeilijk een conclusie getrokken worden op basis van de resultaten, aangezien het model niet convergeert met de gekozen instellingen. Toch is het advies deze techniek niet links te laten liggen aangezien we positieve resultaten bevonden bij het gebruik van een QN waarbij Reinforcement Learning een oplossing vindt binnen een complexe omgeving dat bijvoorbeeld met Machine Learning moeilijk haalbaar is. Bij het implementeren van Q-Learning raden we wel aan om te kiezen voor Deep Q-Learning gezien de aantal states te veel is voor een algemeen Q-Learning model.

Bronvermelding

[1] https://theses.ubn.ru.nl/bitstream/handle/123456789/5216/Nieuwdorp%2C_T._1.pdf?sequence=1

[2] https://pure.tue.nl/ws/files/46933213/844320-1.pdf