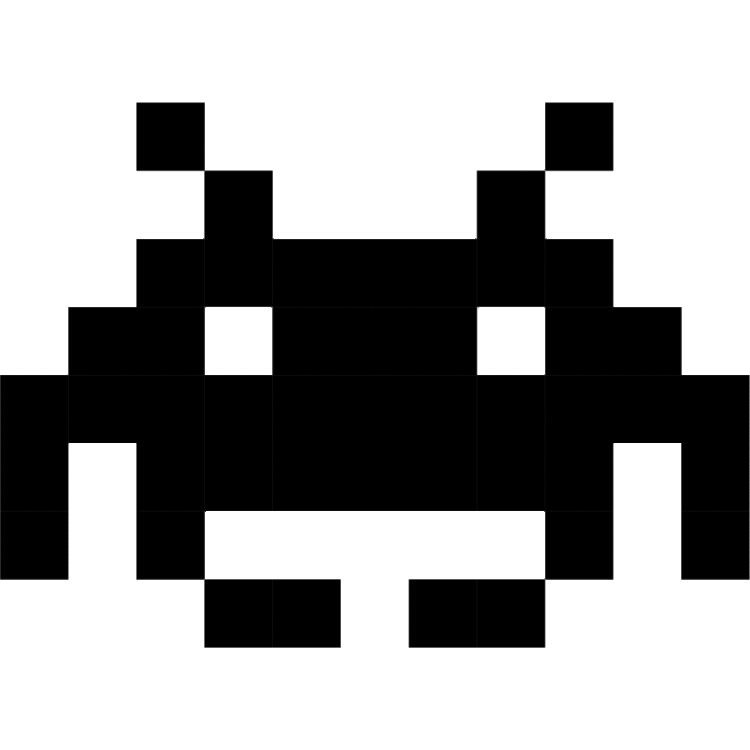
Rapport Space Invader Agent

Nima Agha Ghafar Hamedani (22085947)

Tommi Lander

Isa Dijkstra

Vince Ammerlaan

09-05-2025

Inhoudsopgave

[Inleiding 3](#_Toc197725401)

[Probleemanalyse: Waarom is Reinforcement Learning (RL) geschikt? 4](#_Toc197725402)

[Probleem en doel van dit project 4](#_Toc197725403)

[Kenmerken van het probleem: 4](#_Toc197725404)

[Waarom past RL goed bij dit probleem? 5](#_Toc197725405)

[Verschil met supervised learning en andere AI-oplossingen 5](#_Toc197725406)

[Methodologie (GeeksforGeeks, 2025) 6](#_Toc197725407)

[Resultaten 7](#_Toc197725408)

[Evaluatie 9](#_Toc197725409)

[Literatuurlijst 10](#_Toc197725410)

# Inleiding

In deze opdracht richten we ons op het klassieke computerspel Space Invaders. In dit spel bestuur je een kanon dat zich onderaan het scherm bevindt. Je kunt dit kanon naar links en rechts bewegen en schieten op inkomende aliens.

Ons doel is om een agent te ontwikkelen die zelfstandig leert spelen en een zo hoog mogelijke score behaalt. Dit doen we met behulp van een reinforcement learning-algoritme, waarbij de agent via rewards (de punten die je krijgt als je een alien raakt) leert welke acties in welke situaties het meest effectief zijn.

Space Invaders begint met meerdere rijen aliens die langzaam van links naar rechts en naar beneden bewegen. De speler bestuurt het kanon onderaan en kan bewegen naar links, rechts of schieten. Tussen het kanon en de aliens staan bunkers die dienen als bescherming, maar die geleidelijk afbrokkelen door vijandelijk vuur.

Het spel eindigt op drie manieren:

* Alle aliens zijn verslagen, waarna je doorgaat naar het volgende, moeilijkere level. 🡪 gewonnen
* Je verliest je laatste leven, doordat je meerdere keren geraakt wordt door een alien. 🡪 verloren
* Een alien bereikt de onderste verdedigingslinie. 🡪 verloren

# Probleemanalyse: Waarom is Reinforcement Learning (RL) geschikt?

## Probleem en doel van dit project

**Wat is het probleem?**We willen een slimme agent maken die zelf leert om het spel SpaceInvaders goed te spelen. Deze agent moet:

* Leren hoe de vijanden bewegen,
* Goed kunnen bewegen en op het juiste moment schieten,
* Zich aanpassen aan veranderingen in het spel.

**Wat is het doel?**De agent moet:

* Zelf leren van zijn ervaringen en een goede strategie ontwikkelen,
* Beter presteren dan een agent die zomaar wat doet (bijvoorbeeld een agent die willekeurig beweegt),
* Slim reageren op patronen in het spel, zodat hij langer overleeft en een hoge score haalt.

## Kenmerken van het probleem:

* **Veranderende omgeving:** In het spel SpaceInvaders bewegen de vijanden steeds anders. Soms zijn ze sneller of gebruiken ze andere patronen. Hierdoor verandert het spel constant.
* **Beslissingen in stappen:** Je moet meerdere acties achter elkaar goed uitvoeren, zoals schieten en opzij bewegen, om een goede score te halen.
* **Beloningen komen later:** Je krijgt niet meteen punten voor elke actie. Soms weet je pas later of iets goed was, bijvoorbeeld als je een vijand raakt.

## Waarom past RL goed bij dit probleem?

* **Leren door te doen:** Een RL-agent leert van het proberen en krijgt feedback als iets goed of fout gaat. Hij leert dus zonder dat je hem eerst voorbeelden hoeft te geven.
* **Geen gelabelde data nodig:** In tegenstelling tot andere methodes (zoals supervised learning), heeft RL geen juiste antwoorden vooraf nodig.
* **Kijkt naar de toekomst:** De agent maakt niet alleen keuzes die nu goed zijn, maar kijkt ook wat op de lange termijn het meeste oplevert.
* **Kan zich aanpassen:** Omdat het spel verandert, is het belangrijk dat de agent flexibel is. RL kan zich goed aanpassen aan nieuwe situaties.

## Verschil met supervised learning en andere AI-oplossingen

Bij supervised learning train je een model op voorbeelddata met gelabelde uitkomsten. Voor Space Invaders zou dat betekenen dat je voor elke schermsituatie handmatig moet opschrijven wat de beste zet is, iets wat in de praktijk onmogelijk is omdat er duizenden verschillende situaties zijn.

Eenvoudige scripts werken met voorgeprogrammeerde regels, bijvoorbeeld telkens schieten zodra een alien recht boven je staat. Zulke regels zijn niet adaptief; ze houden geen rekening met onverwachte patronen of verrassende vijandbewegingen. RL daarentegen zoekt voortdurend naar de strategie die de hoogste totale score oplevert, waarbij hij zowel directe als toekomstige beloningen meeneemt in zijn beslissingen.

# Methodologie (GeeksforGeeks, 2025)

We hebben gekozen voor Q-learning, omdat het model hiermee zelf leert door beloningen te krijgen wanneer het iets goed doet. In tegenstelling tot technieken zoals supervised learning, hebben we hier geen gelabelde data nodig. De agent leert alles zelf, puur door te spelen en te experimenteren.

Omdat de observaties in het spel bestaan uit veel getallen (bijvoorbeeld de positie van vijanden, projectielen en het ruimteschip), hebben we deze gegevens eenvoudiger gemaakt door de waarden op te delen in zogeheten "bins". Dit betekent dat we de omgeving in vakjes hebben verdeeld, zodat het model makkelijker kan inschatten in welke situatie het zich bevindt.

De agent kan verschillende acties uitvoeren, zoals:

* Verplaatsen naar links
* Verplaatsen naar rechts
* Schieten
* Niets doen

Welke actie wordt gekozen, hangt af van een strategie genaamd **epsilon-greedy**. Dit betekent dat de agent soms willekeurig iets probeert (exploratie), en soms kiest wat tot nu toe het beste werkte (exploitatie). Hoe vaak hij iets willekeurigs doet, wordt bepaald door de waarde van epsilon (ε).

Tijdens het trainen van de agent hebben we meerdere keren geëxperimenteerd met verschillende instellingen:

* **Alpha (α)**: bepaalt hoe snel het model leert van nieuwe informatie (learning rate).
* **Gamma (γ)**: bepaalt hoe belangrijk toekomstige beloningen zijn (discount factor).
* **Epsilon (ε)**: bepaalt hoe vaak het model iets nieuws uitprobeert.

Voor elk van deze drie parameters hebben we een aantal waarden getest. De agent werd getraind over 100 episodes per instelling. De behaalde scores per episode werden opgeslagen, zodat we later konden analyseren wat het effect van elke parameter was op de prestaties van de agent.

Na afloop hebben we drie aparte grafieken gemaakt waarin we per parameter (alpha, gamma, epsilon) kunnen zien hoe de leerprestaties zich ontwikkelen. Zo kunnen we bepalen welke instellingen het beste werken voor het spel Space Invaders.

# Resultaten

A graph with blue and orange lines

AI-generated content may be incorrect.

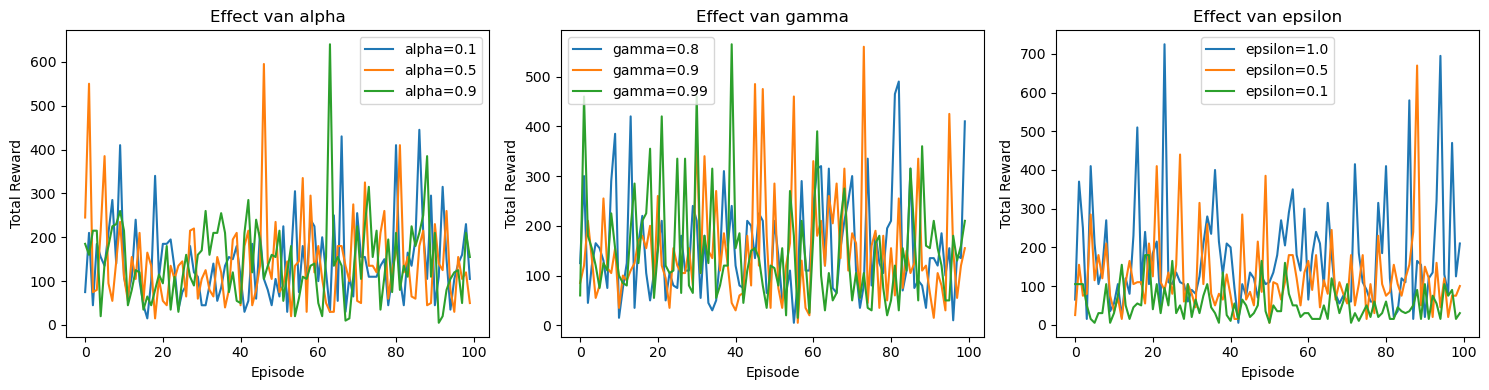
In de grafiek hierboven worden de rewards van de Q-learning agent vergeleken ten opzichte van de Random Agent. De episodes staan op de x-as en de total rewards staan op de y-as.

Wat opvalt is dat de Q-learning agent vanaf ongeveer episode 60 duidelijk beter te preseteren dan de Random-Agent. Dit komt door het effectief leren over tijd. De Q-Learning Agent heeft gemiddeld een betere en stabielere prestatie dan de Random Agent.

A graph with blue and orange lines

AI-generated content may be incorrect.

In de grafiek hierboven hebben we de agents opnieuw getraind maar dan met 140 episodes. In deze grafiek zie je ook net als bij de eerdere grafiek dat bij episode 60 de Q-learning Agent duidelijk beter scoort dan de Random Agent. Je ziet nu wel dat de Q-learning lijn erg fluctuerend is en soms onder de Random Agent zit. Het leerproces is dus een beetje instabiel. Een mogelijkheid om de Q-learning agent beter te laten presteren is door de hyperparameters te optimaliseren. Dus een andere combinatie van alpha, gamma en epsilon.



De afbeelding hierboven toont drie lijngrafieken die het effect van verschillende hyperparameters op de totale reward van een agent in onze reinforcement learning-omgeving visualiseren.

In grafiek 1 zie je het effect van alpha. Drie alpha waardes worden met elkaar vergeleken, 0.1, 0.5 en 0.9. De reward fluctueert veel voor alle alpha waardes, maar er zit wel verschil in stabiliteit en piekrewards. Hierdoor denken we dat de alpha/leerfactor invloed heeft op hoe stabiel de agent leert.

In grafiek 2 zien we het effect van gamma. Drie gamma waardes worden met elkaar vergeleken, 0.8, 0.9 en 0.99. Ook deze lijnen fluctueren veel.

In grafiek 3 zien we het effect van epsilon. Drie epsilon waardes worden met elkaar vergeleken, 0.1, 0.5 en 1.0. Bij een epsilon van 0.1 loopt de lijn wat stabieler maar minder hoge rewards (exploitatie) en bij een epsilon van 1.0 is de spreiding van beloning groot maar er worden wel vaak hoge beloningen bereikt (exploratie).

Uit deze grafieken halen we de conclusie dat de agent gevoelig is voor de hyperparameters, vooral voor de alpha en epsilon.

# Evaluatie

We hebben onderzocht welke hyperparameters het beste werken voor ons model. Uit de experimenten bleek dat de optimale waarden zijn: alpha = 0.9, gamma = 0.9 en epsilon = 1.0. In de eerste trainingsronde van de Q-Learning Agent gebruikten we nog de standaardwaarden alpha = 0.1, gamma = 0.99 en epsilon = 1.0.

Omdat we nu weten dat andere hyperparameters betere resultaten opleveren, verwachten we dat de Q-Learning Agent met de nieuwe instellingen duidelijk beter zou presteren dan de Random Agent. Helaas konden we dit niet aantonen in een grafiek, omdat onze kernel tijdens het trainen meerdere keren crashte.

# Literatuurlijst

* GeeksforGeeks. (2025, 25 februari). *QLearning in reinforcement learning*. GeeksforGeeks. <https://www.geeksforgeeks.org/q-learning-in-python/>
* Kovalchuk, G. (2024, 14 november). A Beginner’s Guide to Q-Learning: Understanding with a Simple Gridworld Example. Medium. <https://medium.com/@goldengrisha/a-beginners-guide-to-q-learning-understanding-with-a-simple-gridworld-example-2b6736e7e2c9>
* Dobilas, S. (2024, 16 december). *Reinforcement Learning (RL) - What Is It and How Does It Work?* Towards Data Science. <https://towardsdatascience.com/reinforcement-learning-rl-what-is-it-and-how-does-it-work-1962cf6db103/>
* Sutton, R. S., & Barto, A. G. (2014). Reinforcement Learning: An Introduction (Second edition, in progress). The MIT Press. <https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf>
* Alpha and Gamma parameters in QLearning. (n.d.). Stackoverflow. <https://stackoverflow.com/questions/1854659/alpha-and-gamma-parameters-in-qlearning>
* GeeksforGeeks. (2023, January 10). EpsilonGreedy Algorithm in Reinforcement learning. GeeksforGeeks. <https://www.geeksforgeeks.org/epsilon-greedy-algorithm-in-reinforcement-learning>
* ChatGPT - Beste hyperparameters vergelijken. <https://chatgpt.com/share/e/681e3ff5-7084-800d-95b6-c25409bb932d>