Politechnika Wrocławska

Wydział Matematyki

KIERUNEK:
Matematyka Stosowana

PRACA DYPLOMOWA INŻYNIERSKA

TYTUŁ PRACY:

Analiza efektywności metod uczenia przez wzmacnianie w grach komputerowych

AUTOR:

Adrian Galik

PROMOTOR:

dr hab. Janusz Szwabiński

WROCŁAW 2024

1 Wstęp

Rozwój technologii w tempie przekraczającym wszelkie oczekiwania oraz zwiększająca się dostępnosć mocy obliczeniowej doprowadziły do tego że algorytmy uczenia maszynowego stanowią nieoderwalną część życia codziennego każdego z nas. Zastosowanie ich można znaleść w dziedzinach robotyki, rozpoznawania obrazów, przetwarzania języka naturalnego, klasyfikacja spamu, systemy nawigacyjne, diagnostyka chorób, sztuczna inteligencja w grach oraz wiele innych gałęzi technologii które oddziałują na nas w sposób pośredni lub bezpośredni. Jedną z najbardziej fascynujących, a zarazem najstarszych dziedzin uczenia maszynowego jest uczenie przez wzmacnianie. Znana już od lat 50 ubiegłego wieku będzie ona kluczowym działem z którego algorymy będą stanowiły fundament mojej pracy.

Celem niniejszej pracy inżynierskiej jest analiza efektywności wybranych metod uczenia przez wzmacnianie w grach komputerowych. Przede wszystkim badania oraz porównania algorytmów zarówno jeśli chodzi o czas uczenia oraz efektywność zostały przeprowacone na przykładzie gry Pong, która jest bardzo często wykorzystywana jako dobry przykład środowiska testowego do badań nad algorytmami sztucznej inteligencji. W ramach pracy zaimplementowałem trzy popularne metody uczenia przez wzmacnianie: Deep Q-Learning (DQN), Advantage Actor-Critic (A2C) oraz Asynchronous Advantage Actor-Critic (A3C), a w następnym kroku zbadałem ich efektywność na zasadzie różnych parametrów m. in. prędkość uczenia oraz skuteczność gry.

2 Wprowadzenie do uczenia maszynowego

Uczenie maszynowe jest jedną z kluczowych gałęzi sztucznej inteligencji, której celem jest tworzenie algorytmów zdolnych do uczenia się na podstawie danych i podejmowania decyzji bez konieczności programowania reguł działania. Oto nieco ogólniejsza definicja: Uczenie maszynowe to "dziedzina nauki dająca komputerom możliwość uczenia się bez konieczności ich jawnego programowania". - Arthur Samuel, 1959. A tu bardziej techniczna: "Mówimy, że program komputerowy uczy się na podstawie doświadczenia E w odniesieniu do jakiegoś zadania T i pewnej miary wydajności P, jeśli jego wydajność (mierzona przez P) wobec zadania T wzrasta wraz z nabywaniem doświadczenia E". - Tom Mitchell, 1997. Przykładowe dane używane do trenowania systemu noszą nazwę **zbioru/zestawu uczącego** (ang. training set). Każdy taki element uczący jest nazywany **przykładem uczącym (próbką uczącą)**. Część systemu uczenia maszynowego odpowiedzialna za uczenie się i uzyskiwanie przwidywań nazywana jest modelem. Przykładowymi modelami są sieci neuronowe i lasy losowe. Dla przykładu klasyfikacji spamu to zgodnie z definicją Toma Mitchella: naszym zadaniem T jest oznaczenie spamu, doświadczeniem E - dane uczące a do wyznaczenia pozostaje miara wydajności P. Może być nią na przykład stosunek prawidłowo oznaczonych wiadomości do przykjładów nieprawidłowo zaklasyfikowanych. (książka uczenie maszynowe z użyciem Scikit-Learn, Keras i TensorFlow (5 zdań ostatnich))

2.1 Podział uczenia maszynowego

(Można dodać do każdego jakieś wykresy) Algorytmy uczenia maszynowego można podzielić na cztery ogólne kategorie:

2.1.1 Uczenie nadzorowane

To najczęstszy przypadek uczenia maszynowego. W tym przypadku algorytm uczy się na podstawie oznaczonych danych wejściowych które są opisane przez człowieka oraz odpowiadających im wyników. Głównymi zastosowaniami algorytmów uczenia nadzorowanego to klasyfikicja i regresja. Klasycznym przykładem jest klasyfikacja spamu, polega ona na analizie przez algorytm e-maila i przypisanie do niego kategorii "spam" lub "nie spam". Przykład algorytmów: regresja liniowa, drzewa decyzyjne, SVM

2.1.2 Uczenie nienadzorowane

Algorytm analizuje dane bez użycia jakichkolwiek oznacznień w celu znalezienie grup lub ukrytych wzorców. Kluczowymi zadaniami uczenia nienadzorowanego są między innymi: wizualizacja danych, redukcja wymiarowości, analiza skupień, wyrywanie anomalii, wykrywanie nowości, usuwanie szumu oraz uczenie przy użyciu reguł asocjacyjnych. Przykład algorytmów: K-Means DBSCAN

2.1.3 Uczenie częściowo nadzorowane

Jest to specyficzny przypadek uczenia nadzorowanego, lecz ma ono na tyle odmienne zasady działania że tworzy odzielną kategorię. W uczeniu częściowo nadzorowanym algorytm nie używa oznaczeń nadanych przez człowieka, lecz są one wygenerowane na podstawie danych wejściowych (zazwyczaj stosowane są do tego algorytmy heurystyczne). Jest to szczególnie przydatne w sytuacjach, gdy oznaczanie danych jest kosztowne lub czasochłonne jak przykładowo w diagnos tyce medycznej.

2.1.4 Uczenie przez wzmacnianie

Dziedzina która była zaniedbywana do momentu w którym autorzy projektu Google DeepMind wykorzystali ją w celu nauki komputerów gier Atari. Jest to specyficzna forma uczenia maszynowego gdyż w zasadniczy sposób różni się od wszystkich poprzednich metod gdyż alogrytm nie uczy się za pomocą danych lecz na podstawię interakcji z dynamicznym środowiskiem stąd nazwa "wzmacnianie". Agentem nazywamy element który jest odpowiedzialny za interakcję ze środowiskiem, a same interakcje nazywamy akcjami. Algorytm za wykonanie każdej akcji definiowanej przez autora otrzymuje adekwatnie do oczekiwań nagrodę i karę. Na podstawię tej metody algorytm uczy się strategii która pozwala mu maksymalizować nagrodę na podstawię konkretnego stanu środowiska.

3 Teoretyczne podstawy uczenia przez wzmacnianie

3.1 Podstawowe pojęcia i definicje

- **Agent** - Podmiot który wchodzi w interakcje ze środowiskiem wykonując podane akcje/decyzje oraz obserwacje i otrzymując za to nagrody. Zadaniem agenta jest maksymalizacja długoterminowej nagrody. Na przykład w szachach agentem jest gracz lub program komputerowy.
- **Środowisko** - Jest to wszystko co oddziałuje na agenta i z czym wchodzi on w interakcję. Komunikacja środowiska z agentem ogranicza się do obserwacji i nagrody. Na przykład środowiskiem w szachach jest plansza szachowa.
- **Stan** (s) - Informacje które środowisko dostarcza agentowi. Dają one wiadomości na temat tego co dzieje się wokół niego.
- **Akcje** (a) - Wszystkie czynności które agent może wykonywać w środowisku. Na przykład przesunięcie pionka o jedno pole do przodu.
- **Nagroda** (r_t) - informacja zwrodna od środowiska wskazująca na to czy akcja była korzystna. Nagroda ma charakter lokalny czyli odzwierciedla niedawną działalność agenta, a nie wszystkie jego sukcesy. Celem agenta jest maksymalizacja skumulowanej nagrody.
- **Polityka** (π) - Strategia agenta, która pomaga mu podejmować akcje w danych stanach. Polityka może być deterministyczna ($\pi(s) - a$) albo stochastyczna ($\pi(a|s)$)

(jakiś rysunek można dodać)

3.2 Modele Markowa (MDP)

Procesy decyzyjne Markowa (Markov Decision Processes, MDP) są podstawą matematyczną uczenia przez wzmacnianie. Dzięki MPD możemy zdefiniować środowisko uczenia przez wzmacnianie jako pięciokrotkę:

$$M = (S, A, P(s'|s, a), R(s, a), \gamma)$$

(wzór do sprawdzenia) gdzie:

- S - zbiór możliwych stanów ($s \in S$).
- A- zbiór możliwych akcji ($a \in A$).
- $P(s'|s, a)$ - Funkcja prawdopodobieństwa przejścia ze stanu s do stanu s' po wykonaniu po wykonaniu akcji a.

- $R(s, a)$ - Funkcja nagrody, określa wartość nagordy otrzymanej po wykonaniu akcji a w stanie s.
- γ - Współczynnik dyskontowania, określa istotność przyszłych nagród ($0 \leqslant \gamma \leqslant 1$).

Cechą kluczową w procesie decyzyjnym Markowa jest własność Markowa, która zakłada iż przyszył stan środowiska, zależy jedynie od obecnego stanu i podjętej akcji, a nie od historii wcześniejszych stanów:

$$P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, ...) = P(s_{t+1}|s_t, a_t)$$

Dzięki właściwości Markowa jesteśmi w stanie uprościć modelowanie środowiska, pozwalając określić prawdopodobieństwa przejścia między stanami dzięki funkcji przejścia $P(s'|s, a)$.

3.2.1 Proces Markowa z nagrodami

Abyśmy mogli użyć nagrody, trzeba rozszerzyć klasyczny model procesu Markowa o mechanizm przyznawania nagród. Zatem dla naszego przypadku każda para (s, a) jest skojarzona z funkcją nagrody $R(s, a)$, która określa średnią wartrość oczekiwaną nagrody po wykonaniu akcji a w stanie s:

$$R(s, a) = E[r_{t+1}|s_t = s_t, a_t = a]$$

Nagorda może występować w różnych formach, może być ona pozytywna lub negatywna czy też duża lub mała. Jeżeli nagroda jest przyznawana niezależnie od poprzedniego stanu lub za osiągnięcie danego stanu, wtedy można zachować tylko pary stan -> nagroda, co znacząco pomaga w uproszczeniu zapisu nagrody. Ma to zastosowanie tylko i wyłącznie wtedy, gdy wartość nagrody zależy wyłącznie od stanu docelowego.

Kolejnym elementem na którym warto zwrócić uwagę jest **stopa dyskontowa** γ, gdzie $0 \leqslant \gamma \leqslant 1$. Określa ona wpływ przyszłych nagród w chwili czasowej t.

Dla każdego epizodu definiujemy wartość wynikową w czasie t w poniższy sposób:

$$G_t = R_{t+1} + \gamma R_{t+2} + ... = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Dla każdego punktu czasowego obliczamy wartość wynikową jako sumę kolejnych nagród. Dla