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.

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

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

Gdzie:

- G_t - Skumulowana nagroda, która jest całkowitą wartością nagród jakie otrzyma agent od momentu t w przyszłości. Jest to miara, która oceania, jak dobrze agent postępuje, biorąc pod uwagę zarówno natychmastowa, jak i przyszłe nagrody.
- γ - Współczynnik dyskontowania z zakresu $[0, 1]$ który jest miarą tego jak agent ocenia przyszłe nagrody w porównaniu z nagrodami natychmiastowymi. na przykład:
  - Dla $\gamma = 0$, agent skupia się wyłącznie na nagrodach natychmiastowych.
  - Gdy γ jest blisko 1, agent korzysta z długoterminowych strategii co może przyniać się do bardziej sensownych akcji.
- r_{t+k+1} - Nagroda otrzymana przez agenta w kroku czasowym $t + k + 1$. Są to nagrody będące sygnałami zwrotnymi otrzymanymi od środowiska, mające na celu informowanie agenta o jakości jego działań.
- k - indeks czasowy który określa zasięg możliwości przyszłych decyzji agenta, dzięki któremu jest w stanie obliczyć skumulowaną nagrodę. Sumowanie zaczyna się od $k = 0$ co wskazuje nagrodzie otrzymanej po wykonaniu akcji w stanie s_t.

Skumulowana nagroda G_t jest ma kluczowe znaczenie w uczeniu przez wzmacnianie ponieważ określa ocenę jakości działań agenta. Stanowi ona podstawyw cel, który agent stara się maksymalizować poprzez optymalny wybór akcji. Istnieją dwa kluczowe zastosowania G_t w uczeniu przez wzmacnianie:

- Funkcje wartości:
  - Funckja wartości stanu $V^\pi(s)$ - Oczekiwana skumulowana nagroda, którą agent może uzyskać, zaczynając od stanu s i postępując zgodnie z polityką π.

$$V^\pi(s) = E_\pi[G_t|s_t = s]$$

  - Funkcja wartości akcji $Q^\pi(s, a)$ - Oczekiwana skumulowana nagroda, którą agent może uzyskać, wykonując akcję a w stanie s, a następnie postępując zgodnie z polityką π.

$$Q^\pi(s, a) = E_\pi[G_t|s_t = s, a_t = a]$$

- Algorytmy uczenia:
  - W algorymach takich jak Q-learning, funkcja wartości Q działa na zasadzie aktualizacji skumulowanej nagrody G_t w celu znalezienia odpowiedniej polityki.
  - W metodach takich jak aktor-krytyk (A2C) aktor (polityka) oraz krytyk (funkcja wartości) są aktualizwoane w celu maksymalizacji oczekiwanej skumulowanej nagrody.

Przykład zastosowania skumulowanej nagrody G_t w grze pong: Niech agent będzie graczem który otrzumuje następujące nagrody w kolejnych krokach czasowych:

- $r_1 = +1$ - zdobycie punktu
- $r_2 = -1$ - utrata punktu
- $r_3 = +1$
- $r_4 = +1$
- $r_5 = -1$

Zakładając że $\gamma = 0.9$, wtedy skumulowana nagroda G_0 zaczynając od chwili $t = 0$ będzie obliczana jako:

$$G_0 = \gamma^0 r_1 + \gamma^1 r_2 + \gamma^2 r_3 + \gamma^3 r_4 + \gamma^4 r_5 + ...$$

$$G_0 = 1 * 1 + 0.9 * (-1) + 0.9^2 * 1 + 0.9^3 * 1 + 0.9^4 * (-1) + ...$$

$$G_0 = 1 - 0.9 + 0.81 + 0.729 - 0.6561 + ...$$

Agent będzię skupiał się na maksymalizacji sumy tych wartości, dzięki czemu sprawi to zachętę do podejmowania działań prowadzących do długoterminowych korzyści.
(Jakiś rysunek by się przydał)

3.3 Równanie Bellmana

Jednym z fundamendalnych narzędzi w teorii uczenia przez wzmacnianie jest równanie Bellmana. Umożliwia ono formalizację relacji między wartością stanów a akcjami, co jest niezbędne do optymalizacji polityk agenta. Równanie to pozwala na rekurencyjne obliczanie wartości funkcji, co jest kluczowe dla wielu algorytmów uczenia przez wzmacnianie. "Równanie Bellmana stanowi podstawę dla większości algorytmów uczenia przez wzmacnianie, ponieważ pozwala na efektywne obliczanie wartości stanów i akcji poprzez iteracyjne aktualizacje" - Sutton i Barto, 2018.

3.3.1 Równanie Bellmana dla funkcji wartości stanu $V^\pi(s)$

Funkcja wartości stanu $V^\pi(s)$ określa oczekiwaną sumę zdyskontowanych nagród które agent może uzyskać zaczynając od stanu s i postępując zgodnie z polityką π. Wzór na równanie Bellmana dla wartości stanu:

$$V^\pi(s) = E_\pi[R_{t+1}\gamma V^\pi(S_{t_1})|S_t = s]$$

, gdzie:

- $V^\pi(s)$ - Funckja stanu wartości dla polityki π, określająca sumę zdyskontowanych nagród od stanu s.
- E_π - Oczekiwana wartrość przy użyciu plityki π.
- R_{t+1} - Nagroda otrzymywana po przejściu ze stanu s do stanu s_{t+1} która jest wynikiem podjęcia kacji zgodnych z polityką π.
- γ - Współczynnik dyskonotwania ($0 \leqslant \gamma \leqslant 1$).
- S_{t+1} - Stan osiągnięty po wykonaniu akcji w stanie s.

Równanie to można interpretować poprzez równość wartości stanu s a oczekiwanej nagrodzie otrzymanej po przejściu do kolejnego stanu plus zdyskontowanej wartości nowego stanu, zakładając, iż agent działa zgodnie z polityką π.

3.3.2 Równanie Bellmana dla funkcji wartości akcji $Q^\pi(s, a)$

Funkcja wartości akcji $Q^\pi(s, a)$ określa oczekiwaną sumę zdyskontowanych nagród, które agent może uzyskać, wykonując akcję a w stanie s, zgodnie z polityką π. Równanie jest wyrażone poprzez wzór:

$$Q^\pi(s, a) = E_\pi[R_{t+1} + \gamma Q^\pi(S_{t+1}, A_{t+1})|S_t = s, A_t = a]$$

Gdzie:

- A_{t+1} - Akcja podjęta w stanie S_{t+1} zgodnie z polityką π.

Równanie to mówi, że wartośc akcji a w stanie s jest równa oczekiwanej nagrodzie otrzymanej po wykonani akcji a plus zdystkontowaniej wartości A_{t+1} w nowym stanie S_{t+1}, zakładając, że agent działa zgodnie z polityką π.

3.3.3 Równanie Bellmana dla polityki optymalnej $V^*(s)$ i $Q^*(s, a)$

Polityka optymalna π^* maksymalizuje funkcję wartości: $V^*(s) = max_\pi V^\pi(s)$. Równanie Bellmana dla funkcji wartości stanu w optymalnej poltyce wyraża się poniższym wzorem:

$$V^*(s) = max_a E[R_{t+1} + \gamma V^*(S_{t+1})|S_t = s, A_t = a]$$

Analogicznie poniżej równanie dla funkcji wartości akcji w polityce optymalnej:

$$Q^*(s, a) = E[R_{t+1} + \gamma max'_a Q^*(S_{t+1}, a')|S_t = a, A_t = a]$$

Równania można interpretować jako:

- $V^*(s)$ - Najlepsza możliwa wastość stanu s, uzyskana poprzez wybur najlepszej akcji.
- $Q^*(s, a)$ - Najlepsza możliwa wartość akcji a w stanie s, uwzględniająca przyszłe optymalne decyzje.

Równania te stanowią podstawę dla optymalnej polityki algorytmów takich jak Value iteration i Q-learning.

3.3.4 Metoda iteracji wartości

Algorytm, który pozwala na uteracyjną aktualizację funkcji wartości stanu $V(s)$ zgodnie z rówaniem Bellmana dla optymalnej polityki, do momentu osiągnięcia zbieżności.
Składa się ona z poniższych kroków:

- Zainicjalizuj wszystkie stany V_i z pewnymi wartościami początkowymi. Zawyczaj $V(s) = 0$ dla wszsytkich $s \in S$.
- Dla każdego stanu $sinS$ w procesie decyzyjnym Markowa wykonaj aktualizacje:

$$V(s) \leftarrow max_a \sum_{s'} P(s'|s, a)[R(s, a) + \gamma V(s')]$$

- Powtarzaj poprzedni krok poprzez wykonanie wielu iteracji do momentu gdy maksymalna zmiana $V(s)$ jest mniejsza niż zadany próg.

3.3.5 Metoda iteracji polityki

Algorytm składający się z dwóch głównych kroków: ewaluacji polityki i jej ulepszania. Składa się on z poniższych kroków:

- Zainicjalizuj początkową politykę π_0 oraz $V(s)$
- Oblicz wartość $V^\pi(s)$ dla bierzącej polityki π za pomocą poniższego wzoru:

$$V^\pi = \sum_a \pi(a|s) \sum_{s'} P(s'|s, a)[R(s, a) + \gamma V^\pi(s')]$$

- ulepszenie polityki poprzez wybur akcji a maksymalizującej wartość oczekiwaną dla każdego stanu s za pomocą poniższego wzoru:

$$\pi'(s) = argmax_a \sum_{s'} P(s'|s, a)[R(s, a) + \gamma V^\pi(s')]$$

- Sprawdzenie zbieżności. Jeżeli polityka π' jest taka sama jak π, algorytm kończy działanie. W przeciwnym wypadku, ustaw $\pi = \pi'$ i powtórz poprzednie 2 kroki.

3.4 Metoda entropii krzyżowej w uczeniu przez wzmacnianie

Entropia krzyżowa jest miarą różnicy pomiędzy dwoma rozkładami prawdopodobieństwa. Entropia krzyżowa w kontekście uczenia przez wzmacnianie jest używana do oceny jak dobrze nowa polityka agenta $\pi_{new}(a|s)$ zbliża się do idealnego rozkładu akcji, który ma na celu maksymalizację oczekiwanej skumulowanej nagrody. Wzór na entropię krzyżową między dwoma rozkładami $p(a)$ i $q(a)$ wyraża się następująco:

$$H(p, q) = -\sum_{a \in A} p(a) log(q(a))$$

gdzie:

- $p(a)$ - Jest rozkładem prawdopodobieństwa akcji a według starej polityki $\pi_{old}(a|s)$.
- $q(a)$ - Jest rozkładem prawdopodobieństwa akcji a według nowej polityki $\pi_{new}(a|s)$.

3.4.1 Twierdzenie o próbkowaniu istotnościowym

Próbkowanie istotnościowe pozwala na przekształcenie rozkładu prawdopodobieństwa, aby oszacować wartość oczekiwaną funkcji $f(x)$ przy użyciu próbek pobranych z innego rozkładu prawdopodobieństwa. W kontekście uczenia przez wzmacnianie jest to przydatne w momencie gdy próbki akcji są zbierane na podstawie starej polityki $\pi_{old}(a|s)$, a chcemy oszacować wartości dla nowej polityki $\pi_{new}(a|s)$.
Twierdzenie o próbkowaniu istotnościowym:

$$E_{x \sim p(x)}[H(x)] = \int_x p(x)H(x)dx = \int_x q(x)\frac{p(x)}{q(x)}H(x)dx = E_{x \sim q(x)}[\frac{p(x)}{q(x)}H(x)]$$

gdzie:

- $p(x)$ - Rozkład próbkowania (np. stara polityka)
- $q(x)$ - Rozkład docelowy (np. nowa polityka)
- $H(x)$ - Funkcja entropi w stanie x zdefiniowana jako:

$$H(\pi) = -\sum_{a \in A} \pi(a|s) log(\pi(a|s))$$

3.4.2 Dywergencja Kullbacka-Leiblera

Pozawla ona obliczyć odległość między dwoma rozkładami prawdopodobieństwa $p(x)$ i $q(x)$. W kontekście uczenia przez wzmacjnanie jest ona używana do oceny jak bardzo nowa polityka różni sie od starej polityki.
Definicja dywergencji Kullbacka-Leiblera:

$$KL(p(x)||q(x)) = \sum_x p(x)\frac{p(x)}{q(x)}$$

W kontekście uczenia przez wzmacnianie:

$$KL(\pi_{old}(a|s)||\pi_{new}(a|s)) = \sum_a \pi_{old}(a|s) log(\frac{\pi_{old}(a|s)}{\pi_{new}(a|s)})$$

Dywergencja Kullbacka-Leiblera w kontekście uczenia przez wzmacnianie jest używana do:

- Regularizacji polityki - Ogranicza stopień smiany między starą a nową polityką, co skutecznie niweluję problem nagłych i dużych zmian, które mogą mieć negatywny wpływ na proces uczenia.
- Kontrola eksploracji - Służy do utrzymywania balansu między eksploracją nowych akcji a eksploracją znanych akcji.