Michał Koziarski, Krzysztof Kwater, Michał Woźniak

Wykorzystywanie programów uczenia w głębokim uczeniu przez wzmacnianie : o istocie rozpoczynania od rzeczy małych

Edukacja - Technika - Informatyka nr 2(24), 220-226

2018

© Wydawnictwo UR 2018 „Edukacja – Technika – Informatyka” nr 2/24/2018
ISSN 2080-9069 www.eti.rzeszow.pl
ISSN 2450-9221 online DOI: 10.15584/eti.2018.2.30

MICHAŁ KOZIARSKI[1], **KRZYSZTOF KWATER**[2], **MICHAŁ WOŹNIAK**[3]

Wykorzystywanie programów uczenia w głębokim uczeniu przez wzmacnianie. O istocie rozpoczynania od rzeczy małych*

Using Training Curriculum with Deep Reinforcement Learning. On the Importance of Starting Small

[1] Magister inżynier, Katedra Systemów i Sieci Komputerowych, Wydział Elektroniki, Politechnika Wrocławska, Polska
[2] Student, Katedra Systemów i Sieci Komputerowych, Wydział Elektroniki, Politechnika Wrocławska, Polska
[3] Profesor doktor habilitowany inżynier, Katedra Systemów i Sieci Komputerowych, Wydział Elektroniki, Politechnika Wrocławska, Polska

Streszczenie

Algorytmy uczenia się przez wzmacnianie są wykorzystywane do rozwiązywania problemów o stale rosnącym poziomie złożoności. W wyniku tego proces uczenia zyskuje na złożoności i wymaga większej mocy obliczeniowej. Wykorzystanie uczenia z przeniesieniem wiedzy może częściowo ograniczyć ten problem. W artykule wprowadzamy oryginalne środowisko testowe i eksperymentalnie oceniamy wpływ wykorzystania programów uczenia na głęboką odmianę metody Q-learning.

Słowa kluczowe: głębokie uczenie przez wzmacnianie, uczenie przez transfer, uczenie się przez całe życie, proces uczenia

Abstract

Reinforcement learning algorithms are being used to solve problems with ever-increasing level of complexity. As a consequence, training process becomes harder and more computationally demanding. Using transfer learning can partially elevate this issue by taking advantage of previously acquired knowledge. In this paper we propose a novel test environment and experimentally evaluate impact of using curriculum with deep Q-learning algorithm.

Keywords: deep reinforcement learning, transfer learning, lifelong learning, curriculum learning

* Niniejszy artykuł był prezentowany na konferencji „Osiągnięcia Studenckich Kół Naukowych Uczelni Technicznych – STUKNUT’17”, której celem jest umożliwienie studentom uczestniczącym w działalności kół naukowych prezentacji osiągnięć z zakresu teorii i praktyki z dziedzin związanych z szeroko rozumianą techniką.

Wstęp

Dział uczenia przez wzmacnianie (Sutton, 1998) rozwinął się gwałtownie w ostatnich latach, częściowo dzięki postępom poczynionym w obszarze głębokiego uczenia się. Głębokie sieci neuronowe zostały z powodzeniem połączone z wypracowanymi algorytmami uczenia się przez wzmacnianie, takimi jak Q-learning (Watkins, 1992), prowadząc do uzyskania przez maszynę w kilku schematycznych grach osiągnięć zbliżonych do ludzkich (Mnih, 2013; Silver, 2016). Skuteczne wyuczanie agentów rozwiązywania zadań o stale rosnącym poziomie złożoności, nawet w relatywnie prostych środowiskach, takich jak gry na ośmiobitową platformę Atari, wymaga wykorzystania ogromnych zasobów obliczeniowych, a często również bardzo precyzyjnego doboru hiperparametrów algorytmu, aby uczynić proces uczenia się możliwym do realizacji.

Zaproponowane zostały różne techniki mające na celu zniwelowanie tego problemu. Ogólnie rzecz biorąc, bazują one na ponownym użyciu uprzednio zdobytej wiedzy i znane są jako transfer wiedzy lub uczenie z przeniesieniem wiedzy. Dodatkowo, wśród technik ponownego wykorzystania wiedzy stanowiących specjalną formę nauki z przeniesieniem wiedzy wyróżnić można naukę programów uczenia. Polega ona na uczeniu optymalizatorów na zadaniach o rosnącym stopniu złożoności z zadaniami początkowymi używanymi do nakierowania algorytmu na ostateczne rozwiązanie.

W niniejszym opracowaniu testujemy strategię nauki programów uczenia w kontekście głębokiego uczenia przez wzmacnianie. Próbujemy odpowiedzieć na dwa pytania. Po pierwsze: czy wcześniej zdobyta wiedza może być przekazana, aby przyśpieszyć proces nauki trudniejszych pojęć? Po drugie: czy użycie części dostępnych zasobów na łatwiejsze zadanie może uczynić zachowania, zbyt skomplikowane w innych okolicznościach, możliwymi do wypracowania? Aby odpowiedzieć na te pytania, proponujemy oryginalne środowisko testowe i oceniamy modele uczone z wykorzystaniem algorytmu głębokiego Q-learningu (DQL) (Mnih, 2013), porównując wykorzystanie programów uczenia z tradycyjnym podejściem do nauki.

Powiązane prace

Bengito (2009) przeprowadził dogłębną dyskusję nad różnymi konfiguracjami nauki programów uczenia, skupiając pracę na środowisku nadzorowanym. Zastosowanie programów uczenia w środowisku nienadzorowanym, a dokładniej w kontekście zagadnień probabilistycznych, badał Tu (2011). Lee (2011) również skupia się na problemie uczenia nienadzorowanego z odkrywaniem kategorii i podkreśla wagę rozpoczynania od prostych problemów. Jiang (2015) podjął próbę wykorzystania programów uczenia przy innej metodzie uczenia, w uczeniu samodzielnym.

Znaczna część badań koncentrowała się na przeniesieniu wiedzy w uczeniu przez wzmacnianie. Rusu (2015) opowiada się za ideą polityki ekstrakcji – podejściem, które prowadzić może do zmniejszenia rozmiaru modelu. Rajendran (2015) przedstawia metodę, w której nowe modele uczone są z wykorzystaniem przewodnictwa kilku już zbudowanych na poprzednich zadaniach, gdzie dodatkowa warstwa algorytmu uczenia przez wzmacnianie wykorzystana jest do wyboru źródła, z którego wiedza powinna zostać przekazana do obecnego stanu.

Nauczanie programów uczenia

Ludzie spędzają olbrzymią część swojego życia na nauce, często w ramach formalnego systemu edukacji oraz wg wysoce zorganizowanego schematu. Unikamy uczenia pojęć zbyt trudnych, aby były zrozumiałe dla uczniów, i rozszerzamy wcześniej wprowadzone zagadnienia. Nauczanie programów uczenia, tak jak ludzka edukacja, bazuje na prostej zasadzie stawiania przed agentami zadań o rosnącym poziomie trudności. W przeciwieństwie do tego, jak potoczyły się losy ludzkiej edukacji, nigdy nie stało się ono główną metodą uczenia maszyn.

Mimo tego nauka programów uczenia wydaje się być obiecującą techniką w przypadku zastosowania w uczeniu przez wzmacnianie. Może ona zmniejszyć wpływ złożoności problemu, pozwalając nam uczyć agentów, jak prawidłowo wykonywać działania bardziej skomplikowane, niż byłoby to możliwe w innych okolicznościach. Dowiedzione zostało, że skraca ona czas potrzebny do nauki (Bengio, 2009), co stanowi jeden z najpilniejszych problemów mających wpływ na algorytmy głębokiego uczenia przez wzmacnianie.

Badania eksperymentalne

Aby ocenić wpływ na proces uczenia, wprowadzamy specjalnie zaprojektowane środowisko treningowe. Definiujemy serię zadań o rosnącym poziomie trudności i wykorzystujemy algorytm głębokiego Q-learningu (DQL) (Mnih, 2013), aby nauczyć agentów, jak je rozwiązywać. W kolejnych sekcjach podajemy szczegółowy opis zaproponowanego środowiska treningowego i zestawu eksperymentalnego. Na koniec omawiamy otrzymane wyniki.

Środowisko testowe

Aby skorzystać z nauczania programów uczenia, określić należy serię zadań o rosnącym poziomie trudności, stopniowo przygotowujących uczony algorytm do pożądanego środowiska. Wprowadzamy proste środowisko treningowe, GridWorld, zaprojektowane specjalnie do testowania strategii uczenia z wykorzystaniem programów uczenia. Umożliwia ono łatwą deklarację zadań, jest dopasowane do tworzenia programów uczenia i pozwala użytkownikowi na łatwą modyfikację złożoności analizowanych środowisk, skracając czas potrzebny do wyszkolenia agentów.

Tabela 1. Opis bloków tworzących GridWorld

Blok	Nagroda	Akcja
Pusty	0	-
Cel	1	koniec gry
Monety	0.1	po wejściu na pole zmienia ono swój typ na „Pusty"
Woda	-0.1	-
Ogień	-1	koniec gry
Ściana	0	nie można wejść na to pole
Portal	0	przenosi agenta losowo do innego portalu
Przełącznik	0	otwiera wszystkie drzwi
Drzwi	0	nie można wejść na to pole przed otwarciem drzwi

Źródło: opracowanie własne.

GridWorld to dwuwymiarowe środowisko luźno oparte na benchmarku MazeBase (Sukhbaatar, 2015). Składa się ono z siatki jednostek zwanych blokami. Celem agenta umieszczonego w tym świecie jest poruszanie się i ewentualne oddziaływanie z nim w taki sposób, aby zmaksymalizować nagrodę otrzymaną od środowiska. Dzięki dostaniu się do bloku otrzymywana jest nagroda oraz, w niektórych przypadkach, wywoływane jest specjalne zdarzenie związane z jego typem. Dostępne bloki zostały szczegółowo opisane w tab. 1. Aby skutecznie działać w środowisku GridWorld, algorytmy muszą wykazać się umiejętnością nawigacji, unikania zagrożeń, radzenie sobie z niepewnością, a nawet, do pewnego stopnia, planowaniem.

Zamiast kreować każdy świat oddzielnie, uczestnicy wybierają rozmiar planszy i liczbę bloków każdego typu. Betonowe światy są wtedy generowane losowo i bazują na podanej specyfikacji. Zapewnia to wystarczającą różnorodność, aby uniemożliwić agentom po prostu zapamiętywanie podjętych wcześniej działań. Na potrzeby tej pracy zdefiniowaliśmy siedem coraz bardziej złożonych zadań.

GridWorld obsługuje dwie reprezentacje stanu świata. Tryb graficzny wymaga od algorytmów nie tylko nawigacji w środowisku, ale również rozpoznawania obrazków składowych i tworzenia wewnętrznej reprezentacji obserwowanego świata. W alternatywnym, uproszczonym trybie stany reprezentowane są w postaci macierzy liczb całkowitych, w której każdy blok ma przypisany numer indentyfikacyjny. W naszej pracy, aby uniknąć długotrwałych obliczeń, zastosowano tryb uproszczony.

Zestaw doświadczalny

W trakcie pracy badawczej strategia nauki programów nauczania porównywana była z tradycyjnym podejściem, w którym algorytmy uczone były od początku na potrzeby każdego zadania. Aby sprawdzić, czy użycie wcześniej zdo-

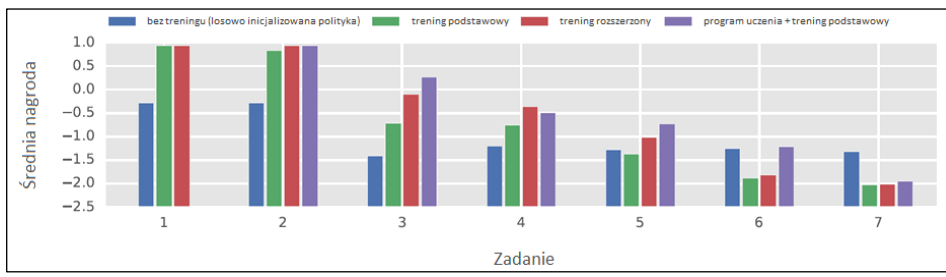
bytej wiedzy daje lepsze rezultaty niż losowa inicjalizacja i czy, biorąc pod uwagę czas uczenia, wykorzystanie części dostępnych zasobów na naukę prostszych zadań może być opłacalną strategią, oceniliśmy wyniki agentów w trzech różnych konfiguracjach. W pierwszej z nich, treningu podstawowym, algorytmy uczone były na konkretnych zadaniach z losowo inicjalizowaną polityką. Następnie algorytmy zostały wyuczone, aby działać w zadaniu o poziom trudniejszym z polityką zainicjalizowaną z użyciem parametrów z zadania prostszego. Na koniec podstawowy trening z pierwszego etapu eksperymentu został rozszerzony poprzez podwojenie czasu jego trwania.

W trakcie eksperymentów do przybliżenia polityki wykorzystano splotową sieć neuronową. Architektura sieci z oryginalnej dokumentacji algorytmu DQL została zmodyfikowana, aby dopasować ją do analizowanych zadań (Mnih, 2013).

Wprowadzone środowisko treningowe, algorytm DQL i wszystkie przeprowadzone eksperymenty zostały zaimplementowane przy użyciu języka programowania Python. Kod został udostępniony publicznie (https://github.com/michal-koziarski/TransferRL).

Wyniki

Zestawienie wyników przeprowadzonej pracy badawczej zaprezentowane zostało na rys. 1. W porównaniu z losową inicjalizacją rozpoczynanie z wiedzą z łatwiejszych zadań skutkuje poprawą wydajności w każdym przypadku. Strategia nauczania programów uczenia prowadziła nie tylko do lepszych rezultatów w większości przypadków, ale również pozwoliła nam osiągnąć lepszą niż losowa średnią nagrodę przy bardziej złożonych zadaniach, w których standardowe podejście do uczenia nie było wystarczające. Niepowodzenie przy próbie nauki obserwowane było przy ostatnim, najbardziej złożonym zadaniu, nawet przy użyciu programów uczenia. Jednak w tym przypadku polityka wypracowana podczas wykonywania prostszego zadania nie była wystarczająca, aby dobrze sobie z nim poradzić.



Rysunek 1. Średnia nagroda otrzymana przez agentów uczonych przy użyciu różnych strategii uczenia na zadaniach o rosnącym poziomie złożoności

Źródło: opracowanie własne.

W związku z powyższym przekazywanie niewystarczającej wiedzy mogło być nieopłacalne. Z drugiej strony wykorzystanie programu uczenia pozwoliło nam osiągnąć wynik lepszy od losowego dla sześciu zadań, dla których w pozostałych przypadkach obserwowano porażkę. Podsumowując, wyniki wskazują znaczną przewagę wykorzystania programów uczenia podczas pracy w zaproponowanym środowisku testowym.

Podsumowanie. Wnioski i przyszła praca

W niniejszym artykule omówiliśmy możliwe znaczenie łączenia nauki programów uczenia z głębokim uczeniem przez wzmacnianie. Zaprojektowaliśmy środowisko testowe dostosowane specjalnie do deklarowania programów uczenia i wykorzystaliśmy je do porównania strategii nauczania programów uczenia z tradycyjnym podejściem do uczenia. Wyniki przeprowadzonej pracy badawczej wykazały znaczną przewagę używania nauczania programów uczenia w zaproponowanym benchmarku.

Po pierwsze i najważniejsze, dalsze badania powinny koncentrować się na rozwoju strategii nauczania programów uczenia w innych środowiskach testowych. Przeprowadzone eksperymenty miały charakter wstępny. Wykorzystany benchmark był w zasadzie dopasowany, aby umożliwiać nauczanie programów uczenia, a kolejne zadania były wyraźnym rozszerzeniem poprzednich. Określenie programów uczenia o tak stopniowo rosnącym poziomie trudności może w praktyce nie zawsze być możliwe. Byłoby zatem konieczne sprawdzenie ograniczeń nauczania programów uczenia w trudniejszych warunkach eksperymentalnych.

Po drugie, w pracy tej programy uczenia wykorzystano z co najwyżej dwoma kolejnymi zadaniami. Zostało to zrobione ze względu na długość procesu uczenia i pozwoliło nam na znaczne przyspieszenie eksperymentu, wciąż dowodząc działania strategii nauczania programów uczenia. Jednak podczas rozwiązywania rzeczywistych problemów chcielibyśmy raczej zastosować inne podejście. Dwie oczywiste strategie to albo nauka pojedynczego algorytmu na całym łańcuchu zadań, albo oddzielne uczenie prostych zadań i późniejsze łączenie zdobytej wiedzy. Zastosowanie programów uczenia do całego łańcucha zadań powinno pozwolić nam częściowo osłabić efekt przekazania suboptymalnych polityk do kolejnych agentów. Jednak aby potwierdzić tę hipotezę, konieczne są dalsze badania.

Podziękowania

Praca została sfinansowana ze środków statutowych Katedry Systemów i Sieci Komputerowych, Wydział Elektroniki, Politechnika Wrocławska. Badania były częściowo wspierane przez Infrastrukturę PLGrid.

Literatura

Bengio, Y., Louradour, J., Collobert, R., Weston, J. (2009). Curriculum Learning. *Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML 2009)* (s. 41–48). New York: ACM.

Jiang L., Meng D., Zhao Q., Shan S., Hauptmann A., G. Self-Paced Curriculum Learning. AAAI Conference on Artificial Intelligence Twenty-Ninth AAAI Conference on Artificial Intelligence. Pobrane z: https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9750/9929 (15.12.2017)

Lee, Y.J., Grauman, K. (2011). Learning the Easy Things First: Self-paced Visual Category Discovery. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (s. 1721–1728), Colorado Springs: IEEE Computer Society.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M. (2013). *Playing Atari With Deep Reinforcement Learning*. Pobrane z: https://arxiv.org/abs/1312.5602v1 (15.12.2017).

Rajendran, J., Prasanna, P., Ravindran, B., Khapra, M.M. (2015). *ADAAPT: A Deep Architecture for Adaptive Policy Transfer From Multiple Sources*. Pobrane z: https://arxiv.org/abs/1510.02879v1 (15.12.2017).

Rusu, A.A., Colmenarejo, S.G., Gulcehre, C., Desjardins, G., Kirkpatrick, J., Pascanu, R., Mnih, V, Kavukcuoglu, K., Hadsell R. (2015). *Policy Distillation*. Pobrane z: https://arxiv.org/abs/1511.06295v1 (15.12.2015).

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Driessche, G. v. d., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D. (2016). Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*, *64* (2), 10–12.

Sukhbaatar, S., Szlam, A., Synnaeve, G., Chintala, S., Fergus. R. (2015). *MazeBase: A Sandbox for Learning From Games*. Pobrane z: https://arxiv.org/abs/1511.07401v1 (15.12.2017).

Sutton, R.S., Barto, A.G. (1998). *Reinforcement Learning: An Introduction*. T. 1. Cambridge: MIT Press.

Tu, K., Honavar, V. (2011). On the Utility of Curricula in Unsupervised Learning of Probabilistic Grammars. *IJCAI Proceedings-International Joint Conference on Artificial Intelligence* (s. 1523–1528). T. 2. Barcelona: AAAI Press.

Watkins, C.J., Dayan, P. (1992). Q-learning. *Machine Learning*, *8* (3–4), 279–292.