Politechnika Warszawska

WYDZIAŁ ELEKTRYCZNY

Instytut Sterowania i Elektroniki Przemysłowej

Praca dyplomowa inżynierska

na kierunku Informatyka Stosowana

w specjalności Informatyka Stosowana

Porównanie wydajności wybranych języków programowania w realizacji sieci neuronowych do przetwarzaniu obrazów

Piotr Heinzelman

numer albumu 146703

promotor

dr inż. Witold Czajewski

WARSZAWA 2024

Porównanie wydajności wybranych języków programowania w realizacji sieci neuronowych do przetwarzaniu obrazów

Streszczenie

W niniejszej pracy poruszono zagadnienia tworzenia wysokowydajnych systemów obliczeniowych w celu realizacji modeli głębokich sieci neuronowych. Duże modele wymagają efektywnych sposobów obliczania odpowiedzi sieci i prowadzenia procesu uczenia, ponieważ wraz ze wzrostem wielkości modelu wyraźnie wzrasta czas wykonywania obliczeń.

Przypomniano sposoby zwiększania wydajności sprzętu, zwłaszcza te, które mogą być zastosowane w modelach sieci neuronowych.

We wprowadzeniu przybliżono matematyczne modele sieci CNN i MLP, a następnie rozpisano realizację tych modeli jako sekwencję działań arytmetycznych. Zwrócono uwagę na możliwość zwiększenia wydajności realizując operacje przetwarzania w sposób równoległy.

Zaprezentowano pokrótce zadania postawione do wykonania porównywanym modelom.

W pracy dokonano porównania takich własności języków jak: popularność, dostępność bibliotek, wygoda instalacji, wsparcie obliczeń na kartach graficznych, stopień trudności języka, koszt licencji. Zaprezentowano również fragmenty kodu, a także opisano doświadczenia w trakcie pisania kodu.

Omówiono wybrane języki: Python, Maltab, Java, C++. Do budowania sieci wykorzystano biblioteki: YOLO11, (TensoRT? PyTorch) oraz TensorFlow, Scikit-Learn dla Python. LibTorch dla C++, Deep Learning Toolbox dla Matlab oraz własne implementacje dla języka Java.

W podsumowaniu zebrano wyniki, oraz zaprezentowano przesłanki, które mogą być pomocne przy doborze języka w celu realizacji głębokich sieci neuronowych.

Słowa kluczowe: uczenie głębokich sieci neuronowych, sieci splotowe CNN, YOLO, wydajność, klasyfikacja obrazów, obliczenia równoległe, dodawanie sekwencyjne

Comparison of the performance of selected programming languages in the implementation of neural networks for image processing

Abstract

This paper addresses the issues of creating high-performance computing systems for implementing deep neural network models. Large models require effective methods of calculating network responses and conducting the learning process, because the size of the model significantly increases the time of performing calculations.

Methods of increasing hardware performance are recalled, especially those that can be used in neural network models.

The introduction presents mathematical models of CNN and MLP networks, and then describes the implementation of these models as a sequence of arithmetic operations. Attention is drawn to the possibility of increasing performance by implementing processing operations in a parallel manner.

The tasks set for the compared models are briefly presented.

The paper compares such language properties as: popularity, availability of libraries, ease of installation, support for computing on graphics cards, level of language difficulty, and license cost. Code fragments are also presented, and experiences during writing the code are described.

Selected languages are discussed: Python, Maltab, Java, C++. The following libraries were used to build the network: YOLO11, (TensoRT? PyTorch) and TensorFlow, Scikit-Learn for Python. LibTorch for C++, Deep Learning Toolbox for Matlab and our own implementations for Java.

The summary summarizes the results and presents premises that may be helpful in selecting a language for implementing deep neural networks.

Keywords: deep learning, convolution network, image clasification, efficiency, parallel computing

Spis treści

Rozdział 1

Wstęp

[4] [5] [12] [9] [10] [14] [1] [7] [8] [13] [2]

Inspiracją do podjęcia tematu pracy związanego z sieciami neuronowymi był kontakt autora z projektem budowy "Sieci do rozpoznawania zanieczyszczeń w powietrzu" realizowanej na wydziale chemii PW około roku 1995 (był to model sieci MLP). A w konsekwencji lektura [3]. Na wybór konkretnego tematu i dobór języków miała wpływ także lektura [11] oraz [6].

Dzisiejsze prawie powszechne zainteresowanie sztucznymi sieciami neuronowymi w środowiskach zarówno neurobiologów, fizyków, matematyków jak i inżynierów wynika z potrzeby budowania bardziej efektywnych i niezawodnych systemów przetwarzania informacji, wzorując się przede wszystkim na metodach jej przetwarzania w komórkach nerwowych. Fascynacje mózgiem człowieka i jego właściwościami (np. odpornością na uszkodzenia, przetwarzaniem równoległym informacji rozmytej i zaszumionej, i innymi) w latach 40-stych dały początek pracom w zakresie syntezy matematycznej modelu pojedynczych komórek nerwowych, a później na tej podstawie struktur bardziej złożonych w formie regularnych sieci. Należy pamiętać, że z obliczeniowego punktu widzenia, rozwijane i budowane sztuczne sieci neuronowe są oparte na nowych regułach wynikających z zasad neurofizjologii.[3]

Sztuczne sieci neuronowe zdobyły sobie szerokie uznanie w świecie nauki poprzez swoją zdolność łatwego zaadaptowania do rozwiązywania różnorodnych problemów obliczeniowych w nauce i technice. Mają właściwości pożądane w wielu zastosowaniach praktycznych: stanowią uniwersalny układ aproksymacyjny odwzorowujący wielowymiarowe zbiory danych, mają zdolność uczenia się i adaptacji do zmieniających się warunków środowiskowych, zdolność generalizacji nabytej wiedzy, stanowiąc pod tym względem szczytowe osiągnięcie sztucznej inteligencji.[6]

Za protoplastę tych (głębokich) sieci można uznać zdefiniowany na początku lat '90 wielowarstwowy neocognitron prof. Kunihiko Fukushimy. Prawdziwy rozwój tych sieci zawdzięczamy jednak profesorowi Yann A. LeCun, który zdefiniował podstawową strukturę i algorytm uczący specjalizowanej sieci konwolucyjnej CNN. Aktualnie sieci CNN stanowią podstawową strukturę stosowaną na szeroką skalę w przetwarzaniu sygnałów i obrazów. W międzyczasie powstało wiele odmian sieci będącej modyfikacją struktury podstawowej (R-CNN, AlexNET, GoogLeNet, ResNet, U-Net, YOLO)[11]

Ważnym rozwiązaniem jest sieć YOLO (ang. You Only Look Once), wykonująca jednocześnie funkcje klasyfikatora i systemu regresyjnego, który służy do wykrywania określonych obiektów w obrazie i określaniu ich współrzędnych. Obecnie dostępna jest już wersja 12.[11]

Biblioteki dostarczające implementacje modeli sieci neuronowych powstały dla większości języków programowania ogólnego przeznaczenia. Niektóre z nich wykorzystują procesory graficzne do wysokowydajnych równoległych obliczeń, co powoduje gwałtowny wzrost wydajności i znaczące obniżenie czasu uczenia sieci. Budowanie własnych modeli sieci jest dziś w zasięgu osób prywatnych, hobbystów, studentów czy małych zespołów badawczych i nie wymaga ogromnych nakładów finansowych.

1.1 Cel pracy

Podstawowym celem pracy jest ułatwienie podjęcia decyzji o wyborze języka i środowiska w fazie projektowej dla realizacji aplikacji wykorzystujących głębokie sieci neuronowe CNN.

Celem dydaktycznym jest dogłębne zapoznanie się z tematyką sieci MLP [3] oraz CNN [5] [12] poprzez realizacje i testy własnego rozwiązania zwłaszcza z wykorzystaniem możliwości obliczeniowych karty graficznej.

1.2 Układ pracy

Warunkiem niezbędnym do prowadzenia efektywnych badań nad głębokimi sieciami i dużymi modelami jest zdolność efektywnego wykorzystania systemów o dużych mocach obliczeniowych. W pierwszej części opisano metody zwiększania wydajności systemów cyfrowych i zagadnienia przetwarzania równoległego.

W drugiej części opisano stan wiedzy z zakresu działania głębokich sieci neuronowych tj. Perceptronu wielowarstwowego (ang. Multilayer Perceptron, MLP) oraz Konwolucyjnej sieci neuronowej (ang. Convolutional Neural Network, CNN).

Zaprezentowano propagację sygnałów przez sieć, propagację wsteczną i oparty na niej proces uczenia sieci.

W trzeciej części zaprezentowano zadania, które będą rozwiązywane przez badane modele sieci.

W czwartej części dokonano porównania języków, opisano wybrane cechy, informacje o wykorzystanych bibliotekach, pokazano fragmenty kodu. A także zaprezentowano wyniki pomiarów z podziałem na języki.

Ostatnia część zawiera wyniki, oraz przesłanki, które mogą być pomocne przy doborze języka w celu realizacji głębokich sieci neuronowych w zależności od konkretnych wymagań i możliwości stawianych projektowanym rozwiązaniom.

1.3 Kod źródłowy i dane uczące

Przykłady rozwiązań w Python i Matlab zaczerpnięto z [11] [6] [9] [10], a także z instrukcji i przykładów załączonych do wykorzystanych bibliotek.

Obrazy pisma odręcznego pochodzą z bazy MNIST (yann.lecun.com), Zdjęcia twarzy wykorzystane w treningu sieci osób pochodzą z internetu - głownie z serwisów google.com oraz filmweb.pl

Pełen kod dostępny na github: https://github.com/piotrHeinzelman/inz/tree/main/MixedProj
W analizie nie brano pod uwagę czasów czytania plików oraz przygotowania danych.

Rozdział 2

Wydajność

W rozdziale "wydajność" Autor przedstawi stosowane rozwiązania zwiększające wydajność sprzętową systemu, takie jak zestaw rozkazów MMX, AVX oraz możliwości kart graficznych GPU.

Ponadto opisana będzie konieczność realizacji kluczowych operacji takich jak: równoległe mnożenie oraz wieloskładnikowe dodawania niezbędne do efektywnego budowania sieci MLP oraz CNN.

Rozdział 3

Modele matematyczne

W tym rozdziale autor opisze stan wiedzy - matematyczny i sygnałowy model warstwy splotowej CNN oraz warstwy MLP głównych elementów składowych sieci głębokich. A także działanie warstw pomocniczych - łączących, i redukujących. Opisany będzie sposób obliczania odpowiedzi sieci na zaprezentowany wzór - tj. przepływ sygnału wprost. Ponadto opisany zostanie algorytm uczenia "wsteczna propagacją błędu" jako operacja matematyczna i proces przesyłania sygnału wstecz przez sieć.

Rozdział 4

Zadania

W tym rozdziale Autor opisze skrótowo zadania które wykonywały badane modele sieci, oraz zaprezentuje wyniki badań - czyli czasy realizacji zadania przez zbudowane modele.

Rozdział 5

Porównanie języków

W tym rozdziale opisane będą aspekty języków, przedstawiono informacje o wykorzystanych bibliotekach, pokazano fragmenty kodu.

Bibliografia

[1] Hoey, J. V., *Programowanie w asemblerze x64*, Werner, T. G., red. Helion S.A., 2024, ISBN: 978-83-289-0109-4.

[2] Jefkine, *Backpropagation In Convolutional Neural Networks*, 2016. adr.: https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/.

[3] Józef Korbicz Andrzej Obuchowicz, D. U., *Sztuczne sieci neuronowe: podstawy i zastosowania*. Akademicka Oficyna wydawnicza PLJ, 1994, ISBN: 83-7101-197-0.

[4] Kasprzak, W., *Metody sztucznej inteligencji C6.pdf*, materiał dydaktyczny, 2024.

[5] Mieczysława Muraszkiewicza i Roberta Nowaka, praca zbiorowa pod redakcją, *Sztuczna inteligencja dla inżynierów*. Oficyna Wydawnicza Politechniki Warszawskiej, 2023, ISBN: 978-83-8156-584-4.

[6] Osowski, S., *Sieci neuronowe do przetwarzania informacji*. Oficyna Wydawnicza Politechniki Warszawskiej, 2020, ISBN: 978-83-7814-923-1.

[7] Qi, H., „Derivation of Backpropagation in Convolutional Neural Network (CNN)", 2016. adr.: https://api.semanticscholar.org/CorpusID:37819922.

[8] Rasheed, A. F. i Zarkoosh, M., *Unveiling Derivatives in Deep and Convolutional Neural Networks: A Guide to Understanding and Optimization*, working paper or preprint, 2024. DOI: 10.36227/techrxiv.170491744.44652991/v1. adr.:]href%20%7B%20https://hal.science/hal-04409232%7D.

[9] Sarah Guido, A. M., *Machine learning, Python i data science*. Helion S.A., 2021, 2023, ISBN: 978-83-8322-751-1.

[10] Sebastian Raschka, V. M., *Machine learning, Python i data science*. Helion S.A., 2021, ISBN: 978-83-283-7001-2.

[11] Stanisław Osowski, R. S., *Matematyczne modele uczenia maszynowego w językach MATLAB i PYTHON*. Oficyna Wydawnicza Politechniki Warszawskiej, 2023, ISBN: 978-83-8156-597-4.

[12] Stuart Russell, P. N., *Artificial Intelligence: A Modern Aproach, 4th Edition*, Grażyński, T. A., red. Pearson Education, Inc., Polish language by Helion S.A. 2023, 2023, ISBN: 978-83-283-7773-8.

[13] Vincent Dumoulin, F. i †, F. V., *A guide to convolution arithmetic for deep learning*, 2018. adr.: https://arxiv.org/pdf/1603.07285.

[14] Y. Bengio - Université de Montréal, Y. L. .-. N. Y. U., *Convolutional Networks for Images, Speech, and Time-Series*, https://www.researchgate.net, 1997.