AGH

AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

DZIEDZINA: NAUKI INŻYNIERYJNO-TECHNICZNE

DYSCYPLINA: AUTOMATYKA, ELEKTRONIKA I ELEKTROTECHNIKA

AUTOREFERAT ROZPRAWY DOKTORSKIEJ

Sprzętowa akceleracja wymagających obliczeniowo operacji na potrzeby algorytmów sztucznej inteligencji w układach FPGA

Autor: mgr inż. Michał KARWATOWSKI

Promotor: Prof. dr hab. inż. Kazimierz WIATR

Promotor pomocniczy: dr hab. inż. Maciej WIELGOSZ

Praca wykonana:

Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie,

Wydział Informatyki, Elektroniki i Telekomunikacji, Instytut Elektroniki

Kraków, 2023

Streszczenie

Niniejsza rozprawa dotyczy efektywnej implementacji algorytmów sztucznej inteligencji, w szczególności sieci neuronowych, w układach FPGA. Optymalna akceleracja sprzętowa pozwala poszerzyć zastosowania sztucznej inteligencji. Istotnym aspektem towarzyszącym badaniom prezentowanym w pracy jest optymalizacja energetyczna. Na akcelerację obliczeń składa się nie tylko dopasowana architektura sprzętowa, ale również optymalizacja algorytmów. W rozprawie badane są wpływy pruningu oraz kwantyzacji na algorytmy uczenia maszynowego, w tym sieci neuronowych. Głównymi obszarami zastosowań badanych algorytmów jest przetwarzanie języka naturalnego oraz analiza szeregów czasowych. Opracowane zostało narzędzie DL2HDL, które w istotny sposób ułatwia akcelerację w układach FPGA sieci neuronowych, zaprojektowanych w bibliotekach wysokopoziomowych, takich jak PyTorch. Konstrukcja oraz interface'y narzędzia zostały zaprojektowane w sposób przyjazny dla użytkownika, który nie posiada specjalistycznej wiedzy o projektowaniu akceleratorów FPGA. Ważnym elementem narzędzia DL2HDL jest możliwość symulacji oraz testowania działania algorytmu na każdym kroku implementacji projektu. Dzięki opracowanym optymalizacjom narzędzie generuje wydajny akcelerator zorientowany na niską latencję. Optymalizacje pod kątem latencji są wystarczająco daleko idące, aby umożliwić zastosowanie sieci neuronowych w systemach wspomagających akcelerator cząstek LHC pracujący w CERN. Cel rozprawy został osiągnięty. Badania pozwoliły na określenie optymalnych poziomów pruningu oraz kwantyzacji dla analizowanych algorytmów. Wypracowana została lista wymagań oraz powstała implementacja narzędzia DL2HDL mapującego wysokopoziomowy opis sieci neuronowych do zoptymalizowanych architektur sprzętowych w układach FPGA. Szeroko zakrojone optymalizacje pozwoliły na osiągnięcia najniższej spotkanej dotąd w literaturze latencji w trakcie inferencji rekurencyjnych sieci neuronowych typu LSTM.

Motywacja

Nowoczesne społeczeństwo w dużym stopniu polega na systemach informatycznych. Każdy aspekt życia w mniejszym lub większym stopniu kształtowany jest przez interakcje z technologią. Historycznie dominującą rolę w systemach informatycznych odgrywały algorytmy deterministyczne, wymagają one od programisty zaprojektowania wszystkich działań i interakcji z którymi spotka się dane oprogramowanie.

Pierwsze prace nad sztuczną inteligencją prowadzane były już w latach 50 ubiegłego wieku i aż do początku lat 70 pokładano w niej duże nadzieje. Jadnak ze względu na ograniczoną moc obliczeniową ówczesnych komputerów możliwości opracowywanych algorytmów nie były zadowalające. Dopiero w latach 90 możliwości komputerów były wystarczające, aby pozwolić na praktyczne zastosowania. Opracowana w 1998 roku architektura LeNet pokazała możliwości sztucznych sieci neuronowych w rozpoznawaniu pisma odręcznego. W dalszym ciągu jednak algorytmy były zbyt wymagające obliczeniowo. W 2012 roku zaprezentowana została architektura AlexNet. Jest to głęboka sieć neuronowa wykorzystująca karty graficzne do akceleracji obliczeń. Od tego momentu następuje bardzo dynamiczny rozwój algorytmów uczenia maszynowego i sztucznej inteligencji. Stale zwiększa się też zapotrzebowanie na moc obliczeniową. Najnowocześniejsze używane obecnie architektury, na przykład GPT-3, byłyby niemożliwe do wytrenowania 10 lat temu ze względy na ograniczenia ówczesnego sprzętu.

W wielu aplikacjach głównym czynnikiem, pod względem którego optymalizowane są algorytmy, jest koszt inferencji przez sieć neuronową. Częstym schematem jest pobieranie danych na urządzeniu końcowym, transfer do infrastruktury obliczeniowej w chmurze wyposażanej w klastry kart graficznych, gdzie wykonywane są obliczenia, a następnie odesłanie odpowiedzi z powrotem do urządzenia końcowego. Takie rozwiązanie jest bardzo efektywne, ale wprowadza znaczne opóźnienia w działaniu i z tego powodu jest nieodpowiednie w systemach czasu rzeczywistego, również ze względu na swój niedeterministyczny charakter. Rozwiązaniem może być wyposażenie urządzenia końcowego w kartę graficzną tak, aby uniknąć potrzeby transferu

danych, aczkolwiek efektywne wykorzystanie Graphics Processing Unit (GPU) wymaga jak najpełniejszego użycia dostępnej przepustowości. Można to osiągnąć na przykład przetwarzając wiele obrazów jednocześnie. Takie podejście jednak znacząco wydłuża czas odpowiedzi na pojedynczą daną wejściową, przez co nie jest odpowiednie w systemach czasu rzeczywistego o wyższych wymaganiach. Rozwiązaniem mogą być układy Field-Programmable Gate Array (FPGA). Uzyskanie za ich pomocą przepustowości zbliżonej do kart graficznych jest trudne, jednak ich elastyczna architektura pozwala na silne zoptymalizowanie inferencji dla pojedynczej danej, a tym samym pozwala znacznie skrócić czas odpowiedzi.

Odpowiednio zoptymalizowane algorytmy zaimplementowane w układach FPGA pozwalają wielokrotnie zmniejszyć zużycie energii w porównaniu z tradycyjnymi procesorami. Wydajność energetyczna jest ważnym czynnikiem, wielkie centra obliczeniowe zużywają ogromne ilości energii i stanowi ona znaczą część kosztów obliczeń. Innym obszarem, gdzie wydajność energetyczna jest niezwykle istotna, są urządzenia mobilne i robotyka, gdzie nie ma zasilania z sieci, a energia jest czerpana z baterii o ograniczonej pojemności. Zredukowanie zużycia energii przez kluczowe obliczenia może istotnie wpłynąć na czas pracy baterii danego urządzenia.

Poziomy wymagań stawiane przez systemy czasu rzeczywistego mogą być bardzo różne. W przypadku interakcji z człowiekiem latencja rzędu dziesiątek milisekund jest często akceptowalna. Inaczej sytuacja wygląda w przypadku urządzeń przemysłowych czy samochodów autonomicznych, które poruszając się ze znaczną prędkością muszą być w stanie zareagować w czasie o kilka rzędów wielkości krótszym. Wreszcie są też zastosowania, gdzie wymagania co do latencji są ekstremalnie wysokie. Urządzenie wykorzystywane w eksperymentach fizycznych, takie jak Large Hadron Collider (LHC), pracują ze zjawiskami zachodzącymi z prędkościami zbliżonymi do prędkości światła. Wiele części takich systemów mogłoby skorzystać z algorytmów sztucznej inteligencji, jadnak ich wymagania co do czasu odpowiedzi znacząco przekraczają możliwości kart graficznych. Układy Application-Specific Integrated Circuit (ASIC) pozwoliłyby na uzyskanie żądanych latencji, jednak ich cykl produkcyjny jest bardzo długi i ze względu na dynamicznie zmieniające się algorytmy sztucznej inteligencji ich zastosowanie jest trudne. Dodatkowo, w przypadku najnowszych procesów technologicznych produkcja niskoseryjna jest niezwykle kosztowna. Układy FPGA łączą w sobie zalety klasycznych procesorów i specjalizowanych układów.

Układy FPGA programowane są w językach Hardware Description Language (HDL), które znacząco różnią się od języków skryptowych takich jak Python, najczęściej wykorzystywanych przez naukowców i inżynierów zajmujących się algorytmami uczenia maszynowego. Wyma-

gają od programisty innego sposobu rozumowania oraz specjalistycznej wiedzy o platformie sprzętowej, która często obca jest specjalistom z obszaru data science. Sama implementacja algorytmu w układzie FPGA w sposób optymalny jest również istotnie bardziej czasochłonna. Większość istniejących narzędzi pozwalających na wysokopoziomową implementację w układach FPGA powstały z myślą o inżynierach sprzętowych i nie są zoptymalizowane pod kątem sztucznej inteligencji. Obecnie jest niewiele narzędzi pozwalających na konwersję wysokopoziomowych opisów modeli neuronowych do układów FPGA, i skupiają się na inferencji gotowych, wcześniej przygotowanych architektur. Są to narzędzia głównych producentów układów programowalnych AMD Xilinx oraz Intel (Altera). Alternatywne rozwiązania albo mają wiele braków ze strony inżynieryjnej albo wspierają tylko wąską specyficzną grupę algorytmów. Jedynym interesującym narzędziem jest hls4ml, którego głębsze porównanie i analiza znajduje się w rozdziale 5.3.4. Powstało ono pomiędzy pracami opisanymi w rozdziale 4 oraz 5. Stabilne wydania narzędzi głównych producentów układów FPGA zostały wydane już po zakończeniu prac nad narzędziem Deep Learning to Hardware Description Language (DL2HDL), nie koncentrują się one jednak na osiąganiu ekstremalnie niskich latencji oraz nie implementują najbardziej agresywnych optymalizacji.

Cel i tezy pracy

Algorytmy sztucznej inteligencji stwarzają duże wyzwanie dla implementacji sprzętowej w układach FPGA. Część obliczeń przeprowadza operacje na pojedynczych bitach, co daje szczególną przewagę układom programowalnym, które nie są przywiązane do konkretnych szerokości bitowych. Rozwój technologii sprzyja akceleracji obliczeń przy użyciu układów FPGA. Producenci dostrzegli, iż procesory ogólnego przeznaczenia, mimo iż coraz wydajniejsze, nie są w stanie wykonywać niektórych algorytmów w sposób optymalny. Istnieje potrzeba delegacji części obliczeń do komponentów o architekturze dopasowanej do wymagań algorytmu. Od kilku lat oprócz typowych kart montowanych w serwerach dostępne są na rynku podzespoły integrujące w jednym układzie scalonym procesory ARM i układy FPGA. Są one zorientowane na mniejsze zużycie energii i oferują wydajność obliczeniową ze średniego zakresu. Tak ścisłe połączenie procesora i układu FPGA pozwoli na zwiększenie przepustowości przesyłu danych pomiędzy nimi, dopasowanie rodzaju kanału do indywidualnych potrzeb oraz skrócenie latencji.

Niezwykle istotnym aspektem wykonywania obliczeń w większej skali jest ich wydajność energetyczna. Układy FPGA są bardzo elastyczne pod kątem optymalizacji implementacji. Jednym z celów tej pracy jest zbadanie wpływu konfiguracji optymalizacji obliczeń wykorzystywanych przez algorytmu uczenia maszynowego na końcowe zużycie energii.

Znaczna część badań nad sztuczną inteligencją skupia się na uzyskaniu jak najwyższej skuteczności finalnych predykcji. Jadnak optymalizacja inferencji algorytmów również jest wysoce istotna, jeżeli mają być one praktycznie zastosowane w przemyśle. Część związanych z tym aspektów nie jest jeszcze w pełni przebadana i celem tej pracy jest uzupełnienie stanu wiedzy o elementy niezbędne do przeprowadzenia dalszych badań. Część optymalizacji nie ma wpływu na działanie algorytmu, ale z tego powodu ich zakres jest ograniczony. Najbardziej agresywne optymalizacje mogą spowodować degradację wyników osiąganych przez algorytmy sztucznej inteligencji. Celem tej pracy jest zbadanie jak daleko idące optymalizacje można zastosować,

jednocześnie zapewniając że wynik działania badanego algorytmu pozostał niezmieniony lub utrzymał się w akceptowalnym zakresie.

Sieci neuronowe znajdują zastosowanie są w coraz większej liczbie zadań. Między innymi mogą zostać użyte do sterowania urządzeniami wykorzystywanymi w eksperymentach fizycznych, takich jak LHC. Ze względu na dynamikę badanych procesów wymagania co do latencji modeli neuronalnych są ekstremalnie wysokie. Spełnienie ich wymaga stworzenia dedykowanej architektury sprzętowej dla maksymalnie zoptymalizowanych algorytmów. Celem tej pracy jest opracowanie uniwersalnej architektury obliczeń w układach programowalnych pozwalającej na osiąganie tak skrajnie niskich latencji.

Sprzętowa akceleracja algorytmów sztucznej inteligencji otwiera wiele możliwości, aby jednak była dostępna dla szerokiego grona odbiorców, inżynierów, naukowców, powinna płynnie integrować się z typowymi narzędziami używanymi przy opracowywaniu tych algorytmów. W tej pracy wiele uwagi jest poświęcone znalezieniu oraz nakreśleniu wymagań, jakie powinno spełniać narzędzie do konwersji wysokopoziomowego opisu algorytmu uczenia maszynowego do niskopoziomowej i wysoce zoptymalizowanej implementacji w układzie FPGA. Wreszcie celem jest również opracowanie takiego narzędzia.

W wyniku powyższych założeń sformułowano następujące tezy:

"Delegacja części obliczeń algorytmów sztucznej inteligencji do akceleratorów sprzętowych w układach FPGA pozwoli na optymalizację energetyczną obliczeń oraz skrócenie czasu wykonywania algorytmów co pozwoli na poszerzenie pola zastosowań algorytmów sztucznej inteligencji."

"Możliwa jest agresywna optymalizacja inferencji algorytmów uczenia maszynowego oraz efektywna ich implementacja w układach FPGA pozwalająca na osiąganie ekstremalnie niskich latencji przy jednoczesnym zachowaniu wysokiej jakości wyników predykcji."

"Zaprojektowanie narzędzia pozwalającego na konwersję wysokopoziomowego opisu sieci neuronowych do silnie zoptymalizowanych implementacji w układach FPGA pozwoli na łatwą delegację obliczeń do akceleratorów sprzętowych."

Podsumowanie

Celem prac badawczych przedstawionych w niniejszej rozprawie była akceleracja algorytmów sztucznej inteligencji. Wykorzystane zostały w tym celu platformy sprzętowe oparte na układach FPGA. Szczególny nacisk został położony na najbardziej wymagające obliczeniowo operacje. Istotnym aspektem była również integracja narzędzi akcelerujących z istniejącą bazą systemów do projektowania modeli neuronalnych.

Pierwsze badania skupiły się na operacjach w przestrzeni wektorowej. Akceleracja dotyczyła obliczania miary podobieństwa kosinusowego na danych przetworzonych przez schemat ważenia TF-IDF. Algorytm ten pozwala na określenie podobieństwa między dokumentami na podstawie ich zawartości. Powtarzające się wyrazy wskazują większy stopień podobieństwa, jednak są one wcześniej ważone tak, aby generalnie częściej powtarzającym się wyrazom przypisać mniejszą istotność. Powstał projekt akceleratora sprzętowego, a następnie napisane zostało oprogramowanie w języku Very High Speed Integrated Circuit Hardware Description Language (VHDL) które zrealizowało jego działanie. Bazuje on na heterogenicznym połączeniu procesora Central Processing Unit (CPU) z FPGA, nie jest przywiązany do konkretnej platformy sprzętowej, z powodzeniem został uruchomiony na energooszczędnej platformie ZedBoard oraz zestawie serwerowym wyposażonym w kartę PCI-e z układem FPGA Virtex-7. Zbadana została wydajność energetyczna całego zestawu a następnie porównana z analogicznym zestawem bez akceleratora sprzętowego. Oprócz przyspieszenia obliczeń układ FPGA pozwolił na ponad 10-krotne zmniejszenie zapotrzebowania na energię (w zależności od konfiguracji sprzętu, od 10,8 do 12,9). Akcelerator był częścią większego projektu wyszukującego dokumenty podobne w badzie danych.

Kolejnym etapem było zbadanie możliwości dalszej akceleracji poprzez redukcję precyzji obliczeń. Algorytmy sztucznej inteligencji często bazują na prawdopodobieństwie i absolutne wartości są mniej istotne. Przykładowo podczas klasyfikacji najbardziej istotne jest, aby prawidłowy wynik uzyskał najwyższe prawdopodobieństwo, a nie żeby miało ono jakąś konkretną

wartość. Jednak w przypadku bardziej złożonych algorytmów, takich jak głębokie sieci neuronowe, drobne zmiany na początku obliczeń mogą przepropagować się i wpłynąć na wynik końcowy. Przed wprowadzeniem modyfikacji do algorytmów liczących należało sprawdzić jak daleko idące zmiany można wprowadzić tak, aby zachować wymaganą skuteczność całego modelu. Zbadane zostały mechanizmy pruningu oraz kwantyzacji. Wyniki różnią się w zależności od rozwiązywanego problemu, algorytmu oraz danych uczących. Zbadane zostały algorytmy obliczania metryki kosinusowej, K-Nearest Neighbours, konwolucyjne oraz rekurencyjne sieci neuronowe. W większości wypadków osiągnięto szerokości bitowe znacznie poniżej standardowych 32 bitów w reprezentacji zmiennoprzecinkowej, często poniżej 8 bitów reprezentacji stałoprzecinkowej. Możliwości stosowania pruningu w większym stopniu zależą od danych, w wielu przypadkach można uzyskać wyniki poniżej 10% obliczeń, jednak nie jest to reguła. Wypracowane zostały również metody stosowania redukcji precyzji, tak aby uzyskać najlepsze wyniki. Odpowiednie połączenie platformy sprzętowej oraz algorytmów redukcji precyzji jest niezwykle istotne, szczególnie w przypadku mniej elastycznych układów takich jak GPU. W rozprawie badane były przede wszystkim układy FPGA, ponieważ pozwalają na największą swobodę w doborze szerokości bitowych oraz implementacji pruningu. Badane algorytmy zostały z powodzeniem zaimplementowane w układach FPGA, Wyniki eksperymentalne potwierdziły ich działanie.

Ważnym elementem prac wykonanych na potrzeby rozprawy było zaprojektowanie narzędzi pozwalających na łatwą integrację akceleratorów sprzętowych z istniejącym systemem algorytmów sztucznej inteligencji. Implementacja wysoce zoptymalizowanych obliczeń w układach FPGA jest czasochłonna i wymaga specjalistycznej wiedzy. Sprawdzone zostało szereg rozwiązań do implementacji wysokopoziomowej w układach FPGA, co pozwoliło na ukształtowanie serii wymagań jakie powinno spełnić kompletne narzędzie. Stosując te wymagania zaprojektowane oraz zaimplementowane zostało narzędzie DL2HDL. Potrafi ono wygenerować zoptymalizowany kod HDL razem z interface'ami komunikacyjnymi bezpośrednio z modelu neuronalnego wytrenowanego w wysokopoziomowej bibliotece takiej jak PyTorch. Narzędzie to nie wymaga od użytkownika znajomości programowania układów FPGA. Dzięki czystemu interface'owi użytkownika oraz łatwej konwersji modeli zaprojektowanych w popularnych narzędziach zaimplementowane optymalizacje mogą być szeroko stosowane.

Kolejnym etapem było zaprojektowanie i implementacja w narzędziu DL2HDL rozwiązań i optymalizacji pozwalających na uzyskanie możliwie najniższej latencji w trakcie inferencji przez rekurencyjne sieci neuronowe. Zastosowanie tutaj znalazła wiedza zdobyta w trakcie

badań przedstawionych we wcześniejszych rozdziałach rozprawy. W przeprowadzonych eksperymentach uzyskano najlepsze wyniki na kilku zbiorach, w tym na predykcji zdarzeń w szeregach czasowych na danych z czujników magnesów nadprzewodzących będących częścią LHC. Zastosowana architektura składająca się z 2 warstw Long Short-Term Memory (LSTM) oraz liniowej warstwy klasyfikacyjnej uzyskała latencję w trakcie inferencji wynoszącą $210ns$, zużywając przy tym 2.66% jednostek logicznych oraz 1.36% jednostek Digital Signal Processing (DSP) dostępnych w układzie Xilinx Zynq UltraScale+ MPSoC XCZU15EG.

Przedstawiony rezultat jest najlepszym, uzyskanym dotychczas w dostępnej literaturze, wynikiem latencji w trakcie inferencji rekurencyjnych sieci neuronowych. Jednocześnie zachowując zużycie zasobów logicznych układu FPGA na stosunkowo niskim poziomie.

Do najważniejszych oryginalnych rozwiązań autora zaprezentowanych w tej pracy, należy zaliczyć:

- opracowanie projektu oraz implementacja sprzętowa akceleratora obliczeń w przestrzeni wektorowej dla heterogenicznych platform łączących procesor ogólnego przeznaczenia z układem FPGA poprawiającego istotnie wydajność energetyczną,
- zbadanie wpływu ilości danych oraz redukcji precyzji reprezentacji danych na korelację wyników metryki podobieństwa kosinusowego, projekt oraz implementacja akceleratora sprzętowego opartego na układzie FPGA przy użyciu syntezy wysokopoziomowej,
- zbadanie wpływu redukcji precyzji reprezentacji oraz rozmiaru danych wejściowych na skuteczność działania algorytmu K-Nearest Neighbours, projekt oraz implementacja akceleratora sprzętowego opartego na układzie FPGA przy użyciu syntezy wysokopoziomowej,
- zbadanie wpływu kwantyzacji oraz pruningu na wyniki inferencji liniowych, konwolucyjnych oraz rekurencyjnych sieci neuronowych,
- wypracowanie standardów oraz wymagań dla narzędzi pozwalających na wysokopoziomową implementację modeli neuronalnych w akceleratorach sprzętowych opartych na układach FPGA,
- opracowanie projektu oraz implementacja narzędzia DL2HDL pozwalającego na wysokopoziomową implementację modeli neuronalnych w akceleratorach sprzętowych opartych na układach FPGA,

- opracowanie oraz implementacja optymalizacji i rozwiązań pozwalających na stworzenie architektury rekurencyjnych sieci neuronowych w układach FPGA osiągających ekstremalnie niskie latencje.

Wyniki przedstawione w pracy upoważniają do stwierdzenia, że postawione tezy zostały w pełni udowodnione.

Wyniki badań i eksperymentów przedstawione w tej rozprawie pokazują, że algorytmy sztucznej integracji mogą zostać w znacznym stopniu skompresowane poprzez redukcję precyzji oraz pruning. Platformy sprzętowe oparte na układach FPGA są wydajnymi akceleratorami dla algorytmów sztucznej inteligencji. Potrafią zarówno zmniejszyć zużycie energii jak i przyspieszyć działanie. Ich elastyczne architektury pozwalają na efektywną implementację optymalizacji opartych na pruningu oraz kwantyzacji sieci neuronowych. Przy zastosowani najbardziej agresywnych rozwiązań pozwalają na osiągnięcie latencji nieosiągalnych dla innych platform sprzętowych. Narzędzie DL2HDL daje możliwość prostej konwersji modelu neuronalnego zapisanego w wysokopoziomowych narzędziach do silnie zoptymalizowanego kodu HDL. Rozwiązania przedstawione w niniejszej rozprawie mogą przysłużyć się do poszerzenia zastosowań algorytmów sztucznej inteligencji.

Wybrane publikacje autora

[1] M. Wielgosz and M. Karwatowski, "Mapping neural networks to fpga-based iot devices for ultra-low latency processing," *Sensors*, vol. 19, no. 13, p. 2981, 2019.

[2] M. Karwatowski, M. Wielgosz, M. Pietron, M. Staruchowicz, and K. Wiatr, "Comparison of semantic vectors with reduced precision using the cosine similarity measure," in *2017 Intelligent Systems Conference (IntelliSys)*, pp. 898–904, IEEE, 2017.

[3] M. Karwatowski, P. Russek, M. Wielgosz, S. Koryciak, and K. Wiatr, "Energy efficient calculations of text similarity measure on fpga-accelerated computing platforms," in *International Conference on Parallel Processing and Applied Mathematics*, pp. 31–40, Springer, 2015.

[4] K. Wróbel, M. Karwatowski, M. Wielgosz, M. Pietroń, and K. Wiatr, "Compressing sentiment analysis cnn models for efficient hardware processing," *Computer Science*, vol. 21, 2020.

[5] M. Karwatowski, M. Wielgosz, M. Pietroń, K. Piętak, and D. Żurek, "Nlp semi-supervised pu learning with reduced number of labeled examples," in *Future of Information and Communication Conference*, pp. 799–812, Springer, 2021.

[6] J. Caputa, D. Łukasik, M. Wielgosz, M. Karwatowski, R. Frączek, P. Russek, and K. Wiatr, "Fast pre-diagnosis of neoplastic changes in cytology images using machine learning," *Applied Sciences*, vol. 11, no. 16, p. 7181, 2021.

[7] M. Karwatowski and M. Pietron, "Context based lemmatizer for polish language," *arXiv preprint arXiv:2207.11565*, 2022.

[8] M. Pietron, M. Karwatowski, M. Wielgosz, and J. Duda, "Fast compression and optimization of deep learning models for natural language processing," in *2019 Seventh Interna-*

tional Symposium on Computing and Networking Workshops (CANDARW), pp. 162–168, IEEE, 2019.

[9] K. Wr'obel, M. Wielgosz, M. Pietro'n, M. Karwatowski, and A. Smywi'nski-Pohl, "Improving text classification with vectors of reduced precision," in *ICAART*, 2018.

[10] R. Karwatowski and K. Wiatr, "The versatile hardware accelerator framework for sparse vector calculations," *Measurement Automation Monitoring*, vol. 61, no. 7, pp. 327–329, 2015.

[11] M. Wielgosz, M. Karwatowski, M. Pietron, and K. Wiatr, "Fpga implementation of procedures for video quality assessment," *Computer Science*, vol. 19, Jul. 2018.