УДК 004.8

К.т.н., доцент Клятченко Я. М., аспірант Кубай О.Ф.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

СПАЙКОВІ НЕЙРОННІ МЕРЕЖІ ДЛЯ ЕФЕКТИВНОЇ РЕАЛІЗАЦІЇ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ НА ПЛІС

Abstract

Yaroslav Klyatchenko, associate professor, PhD;Oleh Kubai, PhD student

Spiking neural networks for efficient implementation of machine learning algorithms on FPGA

This paper discusses the problems of the popular artificial neural network (ANN) model which is widespread nowadays and a promising alternative to them called spiking neural network (SNN). The 3 main approaches for training and running them on FPGA are discussed. It may be helpful for companies searching for faster and more energy-efficient ways to deploy their neural networks.

Вступ

Протягом останніх десятиліть штучні нейронні мережі (Artificial Neural Networks, ANN) стали основою машинного навчання. Надзвичайних результатів було досягнуто у сферах комп'ютерного зору, обробки мов та робототехніки. Однак, одним зі значних недоліків цієї моделі є велике енергоспоживання та потреба у додатковій пам'яті для навчання, що робить її неефективною для використання у вбудованих системах. Ці обмеження стимулювали розвиток альтернативних підходів, серед яких особливе місце займають спайкові нейронні мережі (Spiking Neural Networks, SNN).

Постановка задачі

Метою роботи є огляд спайкових нейронних мереж та алгоритмів їх навчання для імплементації алгоритмі машинного навчання на ПЛІС, порівняльна характеристика цих алгоритмів, а також висвітлення питань що досі потребують дослідження.

Порівняння ANN і SNN

SNN обробляють інформацію у спосіб схожий до біологічного мозку, використовуючи окремі імпульси (спайки) замість безперервних значень для активації нейронів. Це дозволяє запускати такі моделі з використанням значно простіших електричних схем. Розглянемо компоненти найпростішої електричної схеми придатної для запуску ANN:

1. Регістри для зберігання ваг з'єднань нейронів і результатів проміжних досліджень. ANN, як правило, використовують 32-бітні дробові числа. В залежності від того який тип пам'яті використовується, static чи dynamic RAM, для зберігання одного біту необхідно від одного до п'яти транзисторів.
2. Arithmetic Logic Unit (ALU) для виконання операцій над цими числами.
3. Control Unit (CU) для оркестрації виконання моделі.

Тобто необхідна Фон-Нейманівська машина, або аналогічна їй. Можлива оптимізація ANN за рахунок представлення ваг з'єднань у вигляді резисторів, вхідні сигнали в такому разі будуть у вигляді струмів. Їх додавання можна виконати завдяки першому правилу Кірхгофа (з'єднання провідників у вузол), нелінійна функція активації може бути імітована, наприклад, за допомогою операційного підсилювача. Схожим чином працюють деякі аналогові підсилювачі нейронних мереж, проте цей підхід має кілька недоліків, як-от втрата точності у порівнянні з цифровою реалізацією, а також потенційно велике енергоспоживання.

Найпоширенішою моделлю нейрона SNN є Leaky Integrate-and-Fire (LIF), яка схожа на апаратну реалізацію ANN описану у попередньому абзаці, проте в якості функції активації використовується threshold-функція яка генерує спайк на виході нейрона після накопичення достатнього заряду на його мембрані (за рахунок вхідних струмів). Ця модель має багато переваг порівняно з ANN, основні:

1. Нижче енергоспоживання за рахунок того що спайки можуть мати однакові невеликі значення напруги.
2. Стійкість до завад в середовищі (залежить від кодування сигналів).
3. Вбудований механізм пам'яті (заряд зберігається на мембрані нейрона до моменту спайку).

Було запропоновано кілька способів апаратної реалізації (RC-ланцюги, мемристори), в рамках даної роботи розглянемо компоненти однієї з можливих реалізацій SNN на програмованих логічних інтегральних схемах (ПЛІС):

1. Регістри – для зберігання ваг нейронів.
2. Суматори – для виконання додавання/віднімання.
3. Лічильник – у якості мембрани нейрона.

4. Компаратор – для порівняння значення на лічильнику з максимально допустимими і генерування вихідного спайку.

Реалізація SNN на ПЛІС теоретично не є найефективнішою, проте вона має всі переваги описані раніше і дозволяє з достатньою гнучкістю реалізувати досить складні алгоритми на відміну, наприклад, від мемристорів.

Навчання SNN

Найефективніший алгоритм навчання ANN – зворотнє поширення помилки, обчислює часткові похідні функції помилки (loss function) по кожній з ваг у нейронній мережі правила ланцюга для диференціювання складених функцій. Це означає що необхідно обчислити в тому числі і похідну функції активації нейрона, що неможливо у випадку SNN оскільки степ-функція не безперервна, її похідна дорівнює нескінченності в точці «0» і нулю в інших точках. Через це увагу дослідників привертають альтернативні алгоритми, які можна поділити на 3 групи:

1. Біологічно натхненні алгоритми – базується на біологічних маханізмах адаптації синапсів. Зміна ваг, як правило, відбувається локально, залежно від відносного часу активації синапсів (наприклад, Spike-Timing-Dependent Plasticity, STDP[7]). Також є реалізації з використанням глобального сигналу помилки (Reward-Modulated STDP [6], PEPITA [3]). Алгоритми цієї групи працюють досить повільно, тому підходять тільки для онлайн навчання, хоча можуть бути просто реалізовані на ПЛІС. Тим не менш, вони залишаються під прискіпливою увагою дослідників, адже біологічний мозок є результатом мільйонів років еволюції і є значно ефективнішим за будь-який з алгоритмів цієї групи, що зберігає простір для подальших досліджень.
2. Конвертування ANN в SNN – спочатку тренується класична нейронна мережа, після цього її параметри конвертуються у спайкову модель. Це дає можливість використовувати існуючі фреймворки і обчислювальні потужності, які оптимізовані під ANN. Найпростіший приклад застосування цього способу – тренування мережі з функцією активації ReLU і конвертація її в LIF модель [5]. Серед відкритих питань:
    a. Конвертування функцій активації відмінних від ReLU.
    b. Оптимізація алгоритмів навчання для збільшення точності SNN після конвертуванні
3. Донедавна конвертування ANN в SNN вважалося найкращим підходом до навчання SNN, проте точки глобальних мінімумів в ANN і результуючої SNN, як правило, відрізняються, що призводить

до деградації точності (5-10%). Значним кроком вперед є використання сурогатних градієнтів де при прямому поширенні (forward pass) використовується SNN, а при зворотному (error backpropagation) – аналогічна SNN. Цей спосіб значно зменшує втрати точності оскільки глобальна помилка обчислюється з використанням SNN, а не ANN. Найефективнішою функцією для сурогатного градієнту є арктангенс (опираючись на емпіричні дослідження [4]). Відкриті питання:

a. Чому цей метод працює так добре? [2] надає аргументацію цьому, проте вона опирається на апроксимацію «глобальної» функції активації мережі за допомогою диференційовної функції
b. Оптимізація для роботи з стохастичним градієнтним спуском (Stochastic Gradient Descent, SGD) – цей метод, як правило, використовується з алгоритмом оптимізації Adam. SGD працює на невеликих мережах, проте його ефективність значно падає при збільшенні кількості шарів [1]. Adam потребує в 3 рази більше параметрів ніж найпростіша реалізація SGD, що ускладнює його використання для онлайн навчання у вбудованих пристроях.
c. Дослідження алгоритмів для ефективного навчання цим способом на ПЛІС на інших апаратних системах. Цей пункт також включає дослідження алгоритмів які були раніше запропоновані для ANN у контексті SNN, як от Feedback Alignment.

Висновки

Спайкові нейронні мережі є перспективним напрямом розвитку енергоефективного глибинного навчання, а ПЛІС — гнучкою платформою для їх реалізації. Серед трьох основних підходів до навчання SNN найбільш збалансованим за точністю, стабільністю та складністю апаратної реалізації є метод сурогатних градієнтів. Він дозволяє досягнути точності співставної з класичними нейронними мережами. Попри згадані переваги, спайкові нейронні мережі використовуються рідше ніж класичні через недостатню теоретичну базу а також через значний успіх останніх, що робить SNN непривабливою технологією для інвесторів. Проте в останні роки розголосу набули проблеми енергоефективності ANN, що привернуло увагу більшої кількості дослідників до SNN.

Література

1. Li Y., Guo Y., Zhang S., Deng S., Hai Y., Gu S. Differentiable Spike: Rethinking Gradient-Descent for Training Spiking Neural Networks. Ред. Ranzato M., Beygelzimer A., Dauphin Y., та ін. Curran Associates, Inc., 2021.
2. Gygax J., Zenke F. Elucidating the theoretical underpinnings of surrogate gradient learning in spiking neural networks. 2024.
3. Dellaferrera G., Kreiman G. Error-driven Input Modulation: Solving the Credit Assignment Problem without a Backward Pass. 2023.
4. Stan M.-I., Rhodes O. Learning long sequences in spiking neural networks. *Scientific Reports*. 2024. Вип. 14, № 1. С. 21957.
5. Lu S., Xu F. Linear Leaky-Integrate-and-Fire Neuron Model Based Spiking Neural Networks and Its Mapping Relationship to Deep Neural Networks. 2022.
6. Khoee A. G., Javaheri A., Kheradpisheh S. R., Ganjtabesh M. Meta-Learning in Spiking Neural Networks with Reward-Modulated STDP. 2023.
7. Bi G., Poo M. Synaptic Modifications in Cultured Hippocampal Neurons: Dependence on Spike Timing, Synaptic Strength, and Postsynaptic Cell Type. *The Journal of Neuroscience*. 1998. Вип. 18, № 24. С. 10464–10472.