УДК 004.75

DOi 10.30837/ bi.2025.2(103).01

Є. В. Бодянський[1], Д. В. Савенков[2]

[1]ХНУРЕ, м. Харків, Україна, yevgeniy.bodyanskiy@nure.ua, ORCID iD: 0000-0001-5418-2143

[2] ХНУРЕ, м. Харків, Україна, denys.savenkov@nure.ua, ORCID iD: 0009-0003-7361-015X

ВПЛИВ ПАРАМЕТРІВ ОПТИМІЗАЦІЇ ІНФЕРЕНЦІЇ НА ЕФЕКТИВНІСТЬ СПАЙКОВИХ НЕЙРОННИХ МЕРЕЖ

Спайкові нейронні мережі (SNN) – це третє покоління штучних нейромереж, яке завдяки своїй енергоефективності та розрідженості ідеально підходить для застосування у ресурсо-обмежених середовищах, як, наприклад, IoT або робототехніка. Однак і вони можуть не зустрічати екстремальних вимог, що призводить до необхідності використання методів оптимізації інференції, зокрема квантизації та прунінг. Сучасні дослідження вже розглядали практичне застосування даних методів для спайкових нейромереж, але вони не зосереджувались на впливі початкових параметрів оптимізації на продуктивність стисненої моделі. Мета цього дослідження полягає у систематизація та емпіричне дослідження впливу параметрів методів квантизації та прунінгу на кінцеву продуктивність спайкових нейронних мереж. Для експериментів було використано архітектуру згорткової SNN (CSNN) на основі нейрона Leaky Integrate-and-Fire (LIF). Модель тестувалась на трьох наборах даних класифікації зображень: MNIST, FMNIST та CIFAR10. Стиснення проводилося методами статичної k-бітної квантизації після навчання та структурованого прунінгу з різними коефіцієнтами, що зустрічаються у практичному використанні. Отримані результати показують, що при невисоких параметрах стиснення SNN демонструють несуттєву втрату точності, одночасно забезпечуючи значне зменшення розміру моделі та енергоспоживання. Однак, для більш складного набору даних, неоптимальної навченої моделі та при екстремальних налаштуваннях стиснення, спостерігається різке та значне погіршення метрик класифікації.

СПАЙКОВІ НЕЙРОННІ МЕРЕЖІ, МАШИННЕ НАВЧАННЯ, ШТУЧНІ НЕЙРОННІ МЕРЕЖІ, ШТУЧНИЙ ІНТЕЛЕКТ, ОПТИМІЗАЦІЯ ІНФЕРЕНЦІЇ, ОПТИМІЗАЦІЯ ВИВЕДЕННЯ, КВАНТИЗАЦІЯ, ПРУНІНГ, НЕЙРОМОРФНЕ ОБЧИСЛЕННЯ

Ye.V. Bodyanskiy, D.V. Savenkov. Inference optimization parameters influence spiking neural network efficiency. Spiking neural networks (SNNs) are the third generation of artificial neural networks, which, thanks to their energy efficiency and sparsity, are ideal for use in resource-constrained environments such as IoT or robotics. However, even they may not meet extreme requirements, leading to the need for inference optimization methods, such as quantization and pruning. Recent studies have already considered the practical application of these methods for spiking neural networks, but they have not focused on the impact of initial optimization parameters on the performance of the compressed model. The goal of this study is to systematize and empirically investigate the impact of quantization and pruning method parameters on the final performance of spiking neural networks. A convolutional SNN (CSNN) architecture based on the Leaky Integrate-and-Fire (LIF) neuron was used for the experiments. The model was tested on three image classification datasets: MNIST, FMNIST, and CIFAR10. Compression was performed using static k-bit quantization methods after training and structured pruning with different coefficients encountered in practical use. The results show that at low compression parameters, SNNs demonstrate insignificant accuracy loss while providing a significant reduction in model size and energy consumption. However, for a more complex dataset, a suboptimal trained model, and extreme compression settings, a sharp and significant deterioration in classification metrics is observed.

SPIKING NEURAL NETWORKS, MACHINE LEARNING, ARTIFICIAL NEURAL NETWORKS, ARTIFICIAL INTELLIGENCE, INFERENCE OPTIMIZATION, QUANTIZATION, PRUNING, NEUROMORPHIC COMPUTATION

Вступ

Спайкові нейронні мережі (Spiking Neural Networks, SNNs) – це третє покоління штучних нейронних мереж (Artificial Neural Networks, ANNs), метою та фокусом яких є відтворення обчислювальних принципів біологічних нейронних систем [1]. Даний тип нейронних мереж імітує такі нейробіологічні процеси, як накопичення і розрядження заряду, мембранний потенціал, рефрактерний період, комунікація через імпульси або «спайки», тощо. На відміну від “класичних” ANNs, які базуються на безперервних значеннях та функціях активації, SNNs виконують дискретні, розрідженні, асинхронні та подія-орієнтовані обчислення основані на подіях. Завдяки цьому SNN мають значні переваги з точки зору енергоефективності та обчислювальної потужності [2], що робить їх придатними для використання на нейроморфному обладнанні з низьким енергоспоживанням, що у свою чергу робить SNN привабливими у завданнях IoT та роботехниці.

Для таких завдань, окрім вимог до продуктивності, також притаманні й обмеження в ресурсах, зокрема пам’яті, часу та електроенергії. Незважаючи на зазначену енергоефективність SNN, практичне впровадження великомасштабних SNN залишається значним викликом. Сам розмір навчених моделей SNN, включаючи велику кількість емульованих синапсів і шарів, може перевищувати обсяг пам'яті цільових апаратних платформ та виконуватись довше зазначеного ліміту. Це особливо актуально для складних

глибоких архітектур, необхідних для досягнення найсучаснішої продуктивності у складних завданнях.

Вирішенням даних проблем займаються задачі та методи оптимізації виведення або інференції (inference optimization) [3], які фокусуються на стисненні та пришвидшенні натренованих моделей машинного навчання без значної втрати продуктивності, з метою подальшою інтеграцію у прикладні та практичні системи. Дані методи можна умовно розбити на апаратні та алгоритмічні рішення. До останніх відносять такі методи, як "квантизація" [4] та "прунінг" [5], що регулярно застосовують у практичних завданнях, зокрема у великомовних моделях (Large Language Models).

Незважаючи на емпірично перевірену ефективність, дані методи дуже чутливі до початкових параметрів, від яких залежить не тільки розмір фінальної моделі, а й втрачена продуктивність. Хоча існують дослідження щодо використання методів оптимізації інференції для SNN, питання залежності цих параметрів та продуктивності залишались поза фокусу.

Об'єктом дослідження є процес оптимізації виводу або інференції (inference optimization) спайкових нейронних мереж для їх ефективного застосування на пристроях з обмеженими обчислювальними ресурсами.

Предметом дослідження є залежність втрати продуктивності (точності) та ступеня стиснення спайкових нейронних мереж від початкових параметрів алгоритмічних методів оптимізації, зокрема квантизації та прунінгу.

Мета дослідження полягає у систематизація та емпіричне дослідження впливу параметрів методів квантизації та прунінгу на кінцеву продуктивність спайкових нейронних мереж, а також розробка практичних рекомендацій щодо вибору цих параметрів для досягнення оптимального балансу між розміром моделі та її точністю.

1. Постановка задачи

Для досягнення поставленої мети, необхідно виконати наступні задачі:

Сформувати теоретичну базу: провести аналіз існуючих підходів до квантизації та прунінгу для SNNs.

Розробити експериментальний стенд: імплементувати тренувальний пайплайн, що дозволяє застосовувати різні комбінації параметрів квантизації та прунінгу до тренованих моделей SNN.

Провести серію експериментів: дослідити вплив ключових параметрів, як-от ступінь стиснення (p) для прунінгу та бітність (k) для квантизації, на фінальну продуктивність моделі.

Проаналізувати результати: порівняти метрики стиснення та втрати точності для кожної комбінації параметрів.

У результаті, буде проведено серію експериментів з різними конфігураціями параметрів оптимізації, а їхні метрики будуть проаналізовані та порівняні. Це дозволить надати чіткі рекомендації для практичного застосування SNN на пристроях з обмеженими ресурсами.

2. Огляд теоретичної бази

Як було зазначено у попередніх розділах, SNN працюють дещо відміно від класичних ANN. Замість безперервних функцій активації, SNN обробляють та передають інформацію за допомогою дискретних асинхронних імпульсів. Ці бінарні активаційні події відбуваються, коли мембранний потенціал нейрона перевищує поріг напруги, імітуючи вивільнення нейромедіаторів. Як і їх біологічні аналоги, SNN демонструють часову динаміку та еволюцію через накопичення та розрядження заряду, і включають рефрактерний період після активації.

Найпростішою і найпоширенішою моделлю SNN є нейрон Leaky Integrate-and-Fire (LIF) [6], який абстрагує накопичення мембранного напруги нейрона як резистор-конденсаторну електросхему. Ця простота робить його обчислювально ефективним і придатним для інтеграції в апаратне забезпечення. Модель LIF можна описати наступним рівнянням:

$$V(t)=\frac{1}{\tau_m}\left(V_{rest}+I(t)R\right), \tag{1}$$

де $V(t)$ — напруга мембрани, V_{rest} — напруга стану "спокою", τ_m — постійна часу мембрани, R — опір мембрани, та $I(t)$ — вхідний струм.

Коли $V(t)\geq V_{threshold}$, нейрон «вистрілює» (генерує імпульс), передає напругу до підключених нейронів, скидає свій мембранний потенціал до V_{reset} і входить у рефрактерний період, під час якого він має меншу ймовірність активації. Цей процес можна побачити на рис. 1.

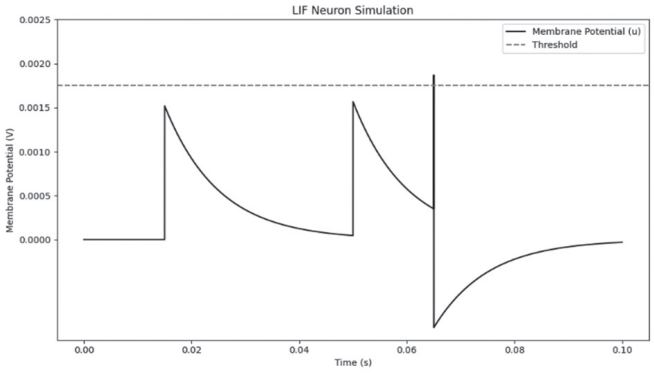

Рис. 1. Процес функціонування LIF нейрону

Незважаючи на свою простоту, моделі LIF є універсальними та обчислювально ефективними в порівнянні з більш складними моделями, такими як моделі Іжакевича [7] або Ходжкіна-Хакслі [8], що моделюють комплексні біохімічні процеси, як іонні

канали. Модель LIF також, у порівнянні з переліченими моделями, демонструє кращу сумісність з класичними методами машинного навчання: вона краще підтримує зворотне поширення та навчання з вчителем; але одночасно ефективно інтегрується з біологічно натхненними парадигмами, як залежної від часу спайку пластичністю (Spike-timing-dependent plasticity, STDP) [9].

Біологічна правдоподібність і низьке енергоспоживання LIF роблять їх перспективною альтернативою ANN, особливо для периферійних і нейроморфних обчислень. Однак у край ресурсно обмежених ситуаціях LIF також може потребувати оптимізації інференції. Хоча оптимізація інференції є широкою темою, найбільш практично використаними методами є квантизації та прунінг (обрізання).

Квантизація [4] - це методика, яка зменшує бітову точність параметрів мережі та активацій, тим самим зменшуючи вимоги до пам'яті та обчислювальних потужностей. У SNN це передбачає представлення безперервного потенціалу мембрани, синаптичних ваг та змінних, пов'язаних з часом, за допомогою меншої кількості бітів. Динамічний діапазон цих змінних може бути великим, а зменшення їх точності може призвести до значної економії в апаратних реалізаціях. Ключовою перевагою квантизації є те, що воно дозволяє виконувати арифметичні операції за допомогою цілочисельних операцій з низькою кількістю бітів, які є швидшими та енергоефективнішими, ніж операції з плаваючою комою. Наприклад, 32-бітне множення з плаваючою комою можна замінити 8-бітним цілочисельним множенням, що призведе до значного зменшення площі апаратного забезпечення та споживання енергії. Загальні методи квантизації розділяють на дві категорії: квантизація після навчання (Post-Training Quantization) [10], яку також розділяють на динамічну та статичну, а також навчання з урахуванням квантизації (Quantization-Aware Training) [11]. Проблема квантизації SNN полягає у збереженні їхньої часової динаміки та потоку інформації, які чутливі до змін точності. Ця чутливість часто вимагає нових схем квантизації, таких як ті, що враховують спайк-орієнтовану природу SNN [12].

Прунінг [5] - це інша загальна методика оптимізації інференції, яка використовується для зменшення розміру нейронної мережі шляхом видалення зайвих або менш важливих зв'язків (ваг) або нейронів. Це призводить до створення більш розрідженої мережі, яка вимагає менше обчислень під час інференції. Мета полягає в досягненні значного стиснення та прискорення моделі без істотної втрати продуктивності. У SNN, як й у ANN, прунінг може застосовуватися як до синаптичних зв'язків, так і до нейронів. Розрідженість, що виникає в результаті прунінгу, є особливо корисною для SNN, які природно працюють з розрідженими, керованими подіями даними. Існує два основних типи прунінгу: неструктурований прунінг та структурований прунінг [3]. Неструктурована обрізка видаляє окремі ваги, що призводить до нерегулярної розрідженості, для використання якої потрібне спеціальне обладнання або програмне забезпечення. На відміну від цього, структурований прунінг видаляє цілі нейрони або канали, що призводить до регулярної розрідженості, яку легше прискорити на стандартному апаратному забезпеченні. Прунінг можна здійснювати двома основними способами: прунінг на основі величини, яке видаляє ваги з найменшими абсолютними значеннями, та прунінг на основі градієнта, яке використовує інформацію з градієнтів мережі для ідентифікації та видалення менш важливих ваг. Ефективність прунінгу в SNN залежить від їхнього навчання та конкретного методу прунінгу, оскільки погано обрізана мережа може втратити здатність кодувати та обробляти часову інформацію.

Підсумовуючи, оптимізація інференції є дуже важливим аспектом прикладних систем штучного інтелекту в умовах обмежених ресурсів. Дані методики активно та ефективно застосовуються у задачах використання великомовних моделей (Large Language Models, LLMs) та IoT, а також разом із SNN. Але треба враховувати, що разом із пришвидшенням моделі дані методики можуть погіршивати їхню точність у залежності від ступеня стиснення. Наступні розділи фокусуються на дослідженні впливу налаштувань методів стиснення SNN моделей на їх продуктивність.

3. Матеріали та методи

Як було зазначено, наша робота фокусується на впливі налаштувань методів стиснення на зменшення продуктивності SNN моделей. Щоб провести експериментальне дослідження, було побудовано наступний пайплайн (його також візуалізовано на рис. 2):

– Сформувати архітектуру SNN моделі M;

– Натренувати її на обраному тренувальному наборі даних D_{train};

– Провести операцію стиснення моделі M зі встановленими параметрами H та отримати стиснену модель M';

– На тестовій виборці Dtest провести тестування;
  – звичайної моделі M;
  – стисненої моделі M';

– Задокументувати результати.

Обрана експериментальна архітектура SNN спеціалізується на виконанні завдань класифікація зображень та складається з послідовних шарів згорток, пулінгу та LIF-нейронів. Повна архітектура зображена на рис. 3. Саму модель було навчено парадигмою навчання з вчителем з використанням алгоритму

backpropagation. Також, в якості операцій стиснення було використано статичну квантизацію після навчання та прунінг.

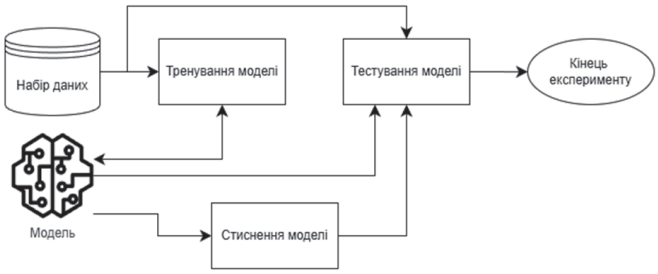

Рис. 2. Тренувальний пайплайн

Для досягнення поставленої мети, необхідно виконати наступні задачі:

– Сформувати теоретичну базу: провести аналіз існуючих підходів до квантизації та прунінгу для SNNs.

– Розробити експериментальний стенд: імплементувати тренувальний пайплайн, що дозволяє застосовувати різні комбінації параметрів квантизації та прунінгу до тренованих моделей SNN.

– Провести серію експериментів: дослідити вплив ключових параметрів, як-от ступінь стиснення (p) для прунінгу та бітність (k) для квантизації, на фінальну продуктивність моделі.

– Проаналізувати результати: порівняти метрики стиснення та втрати точності для кожної комбінації параметрів.

Обрана експериментальна архітектура SNN спеціалізується на виконанні завдань класифікація зображень та складається з послідовних шарів згорток, пулінгу та LIF-нейронів. Повна архітектура зображена на рис. 3. Саму модель було навчено парадигмою навчання з вчителем з використанням алгоритму backpropagation. Також, в якості операцій стиснення було використано статичну квантизацію після навчання та прунінг.

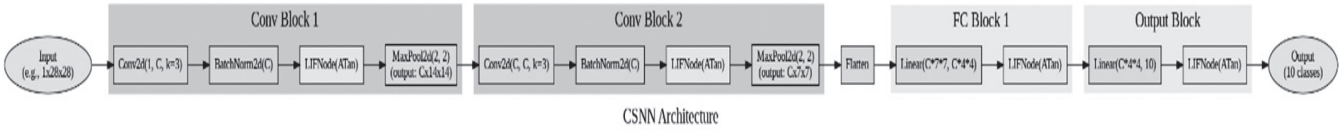

Рис. 3. Використана архітектура SNN

4. Експерименти

Для проведення експериментів з розробленим пайплайном було обрано декілька наборів даних класифікації зображень зі зростаючою складністю: набір MNIST [13], що містить 70 000 зображення рукописних цифр, FMNIST [14], що містить 70 000 зображень одягу, та CIFAR10 [15], який містить 60 000 зображень різних повсякденних об'єктів. Кожен з цих наборів даних має 10 унікальних класів.

З кожним набором даних було побудовано два експериментальних пайплайни, описаних у розділі 3. Також дані експерименти мали таке конфігурацію:

LIF-нейрони мають часову константу мембрани τ встановлено 2.0, для балансу процесу накопичення та розрядження заряду;

Кожен екземпляр даних подається на вхід нейронам T=20ms, щоб імітувати часовий проміжок людського нейрону необхідний для розпізнання інформації;

– Початкові ваги згенеровані випадковим чином, SNN не підлягав оптимізації гіперпараметрів;

– Операції стиснення, як зазначено у розділі 3, виконують після навчання;

– Операції квантизації проводяться з такими значеннями K-бітності: 16, 8, 4;

– Операції прунінгу проводяться з такими значеннями коефіцієнту: 0.1, 0.2, 0.3.

Для проведення експерименту використовуються мова програмування Python, модуль PyTorch [16] та модуль емуляції нейроморфних обчислень SpikingJelly [17]. Також було використано модуль syops [18] для вимірювання енергоспоживання нейромереж до та після прунінгу в умовах емуляції. Значення енергоспоживання базуються на технології емульованого 45 нм процесору, де «арифметичні обчислювальні кроки» (Arithmetic Compute Steps, ACs) коштують 0,9 пДж, а «кроки множення-накопичення» (Multiply-Accumulate Compute Steps, MACs) — 4,6 пДж. Тобто споживання розробленої нейромережі можна апроксимувати до наступної формули:

$$E = 0.9 * ACs + 4.6 * MACs, \quad (2)$$

Результати експериментів описані у розділі 5.

5. Результати

Як було зазначено у попередньому розіділ, кожен експеримент був проведений зі заздалегідь навченими моделями. Кожну з них було виміряно використовуючи класичні метрики задач класифікації: accuracy, precision, recall та f1-score. Результат даних вимірювань описаний у табл. 1. Всі перелічені метрики є зваженими по кожному класу.

Таблиця 1

Метрики натренованих нестиснених моделей(%)

Модель	Набір даних	Метрики на тестовій вибірці			
		Accuracy	Precision	Recall	F1-score
CSNN	MNIST	0.9872	0.9871	0.9871	0.9871
CSNN	FMNIST	0.8582	0.8562	0.8581	0.8544
CSNN	CIFAR10	0.5083	0.5149	0.5083	0.5033

Дана проста модель перед стисненням має гарні показники на простих наборах даних, хоча з більш складним CIFAR10 метрики далекі від оптимальних. Кожна стиснена модель також була протестована на ідентичних тестових підвибірках. Їхні вимірювання перелічені у табл. 2 (де перелічені квантизовані моделі qCSNN(k=n), де k – бітність квантизації) та у табл. 3 (де перелічені моделі після прунінгу pCSNN(p=m), де p – коефіцієнт прунінгу).

Таблиця 2

Метрики стиснених моделей методом квантизації (%)

Модель	Набір даних	Метрики на тестовій вибірці			
		Accuracy	Precision	Recall	F1-score
qCSNN(k=16)	MNIST	0.9854	0.9853	0.9853	0.9852
qCSNN(k=8)	MNIST	0.9851	0.985	0.9851	0.985
qCSNN(k=4)	MNIST	0.9793	0.9793	0.9793	0.979
qCSNN(k=16)	FMNIST	0.8551	0.853	0.8551	0.8514
qCSNN(k=8)	FMNIST	0.8538	0.8519	0.8538	0.8503
qCSNN(k=4)	FMNIST	0.7995	0.8315	0.7995	0.7933
qCSNN(k=16)	CIFAR10	0.5066	0.5115	0.5066	0.5015
qCSNN(k=8)	CIFAR10	0.4999	0.5066	0.4998	0.4958
qCSNN(k=4)	CIFAR10	0.2821	0.5239	0.282	0.2257

Таблиця 3

Метрики стиснених моделей методом прунінгу (%)

Модель	Набір даних	Метрики на тестовій вибірці			
		Accuracy	Precision	Recall	F1-score
pCSNN(p=0.1)	MNIST	0.9865	0.9865	0.9864	0.9864
pCSNN(p=0.2)	MNIST	0.9861	0.986	0.986	0.9859
pCSNN(p=0.3)	MNIST	0.981	0.981	0.9808	0.9808
pCSNN(p=0.1)	FMNIST	0.8574	0.8554	0.8574	0.8538
pCSNN(p=0.2)	FMNIST	0.8518	0.8502	0.8518	0.8474
pCSNN(p=0.3)	FMNIST	0.8406	0.8446	0.8406	0.8348
pCSNN(p=0.1)	CIFAR10	0.4983	0.4998	0.4982	0.4934
pCSNN(p=0.2)	CIFAR10	0.4962	0.5065	0.4962	0.4914
pCSNN(p=0.3)	CIFAR10	0.4206	0.509	0.4206	0.4129

Ефективність на простих наборах MNIST та FMNIST не зазнала сильного впливу. З більш складним CIFAR10, — ефективність класифікації якого була доволі слабкою й до стиснення, — ситуація дещо інша: при низьких параметрах стиснення модель не зазнала значного падіння метрик, але після перетину певного порогу, метрики класифікації CIFAR10 зазнають значного та різкого погіршення.

Погіршення метрик — очікуваний аспект стиснення, але разом із цим очікується й покращення інших характеристик моделі: зменшення розмірності моделі та/або енергоспоживання. Зміна даних характеристик також залежить і від інших факторів, як оптимізація кодової та апаратної імплементації, або використаних технологій емуляція. Але загально їх можна апроксимувати до результатів наведених у наступних двох графах:

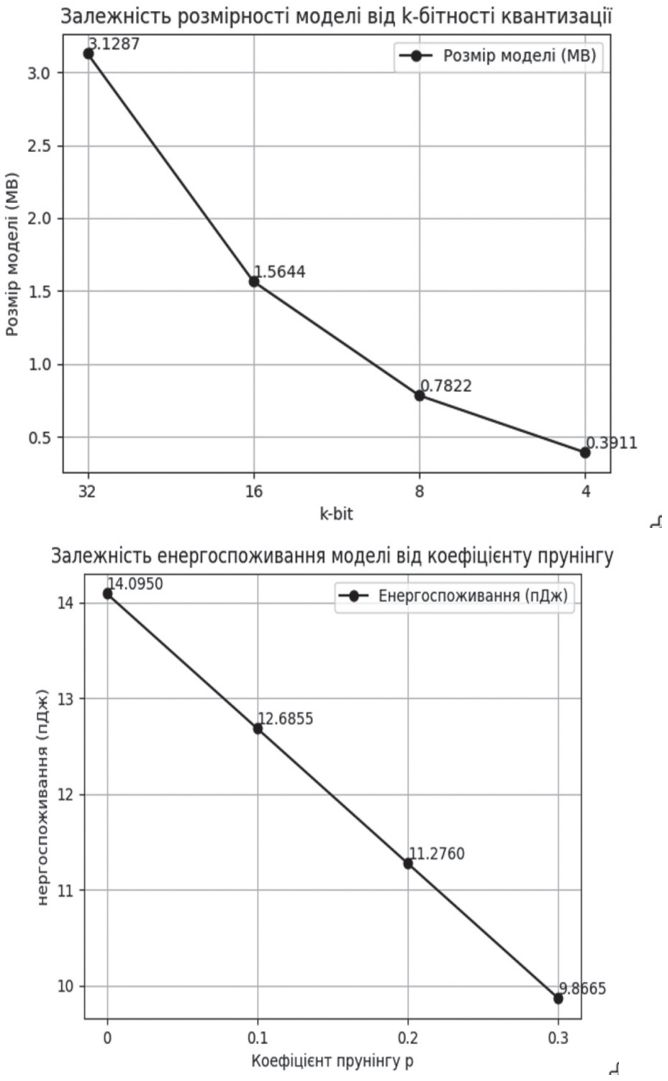

Рис. 4. Графи залежності розмірності моделі від k-бітності квантизації та енергоспоживання моделі від коефіцієнту прунінгу

6. Обговорення

Отриманні результати вказують на несуттєву втрату продуктивності розробленої SNN моделі при невисоких параметрах стиснення, при цьому значно зменшуючи необхідність у пам'яті та енергоспоживанні. Але варто зазначити, що більш "екстремальні" налаштування видають гірші результати для більш комплексних наборів даних. При використанні даних та інших методів стиснення моделей треба враховувати особливості вхідних даних, архітектури та оптимальності самої моделі, зокрема при більш екстремальних налаштуваннях.

Враховуючи інші особливості SNN, зокрема енергоефективність та розріджені обчислення, та результати дослідження, методи стиснення SNN можуть бути ефективно використанні у системах IoT, робототехніці або інших ресурсно-обмежених середовищах, в ситуаціях оптимальності нестиснених моделей. Але саме рішення та використання цих методів все одно підвладні сучасним вразливостям та недолікам SNN.

Висновки

У цій роботі було проведено теоретичне та практичне дослідження впливу на продуктивність SNN таких методів стиснення, як квантизація та прунінг. Для цього були проведені експерименти з типовими наборами даних зростаючої складності для вирішення задач аналізу даних, під час яких було протестовано стиснену за різними параметрами модель на втрату продуктивності за класичними метриками задач навчання з вчителем та класифікації. Також було проведено аналіз оптимізації розмірності та енергоспоживання даних моделей. Практичне значення даної роботи та отриманих результатів полягає у розумінні потенційно очікуваних переваг та вартості застосування стиснення SNN моделей. Дані моделі мають кращу енергоефективність та можливість виконувати розріджені обчислення, що, у поєднанням зі стисненням, робить їх більш привабливими у вирішенні задач інтелектуального аналізу даних у ресурсообмежених системах, як системи IoT або навчання парадигмою онлайн-навчання

Список літератури

[1] Gerstner W. Spiking neuron models: Single neurons, populations, plasticity. Cambridge, U.K: Cambridge University Press, 2002. 480 с.

[2] Davidson S., Furber S. B. Comparison of Artificial and Spiking Neural Networks on Digital Hardware. Frontiers in Neuroscience. 2021. Т. 15. URL: https://doi.org/10.3389/fnins.2021.651141 (дата звернення: 26.11.2025).

[3] Optimization Methods, Challenges, and Opportunities for Edge Inference: A Comprehensive Survey / R. Zhang та ін. Electronics. 2025. Т. 14, № 7. С. 1345. URL: https://doi.org/10.3390/electronics14071345 (дата звернення: 26.11.2025).

[4] Pruning Parameterization with Bi-level Optimization for Efficient Semantic Segmentation on the Edge / C. Yang та ін. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), м. Vancouver, BC, Canada, 17–24 черв. 2023 р. 2023. URL: https://doi.org/10.1109/cvpr52729.2023.01478 (дата звернення: 26.11.2025).

[5] FlatQuant: Flatness Matters for LLM Quantization / Y. Sun та ін. arXiv. 2025.

[6] Burkitt A. N. A Review of the Integrate-and-fire Neuron Model: I. Homogeneous Synaptic Input. Biological Cybernetics. 2006. Т. 95, № 1. С. 1–19. URL: https://doi.org/10.1007/s00422-006-0068-6 (дата звернення: 26.11.2025).

[7] Izhikevich E. M. Simple model of spiking neurons. IEEE Transactions on Neural Networks. 2003. Т. 14, № 6. С. 1569–1572. URL: https://doi.org/10.1109/tnn.2003.820440 (дата звернення: 26.11.2025).

[8] Hodgkin A. L., Huxley A. F. A quantitative description of membrane current and its application to conduction and excitation in nerve. The Journal of Physiology. 1952. Т. 117, № 4. С. 500–544. URL: https://doi.org/10.1113/jphysiol.1952.sp004764 (дата звернення: 26.11.2025).

[9] A neuronal learning rule for sub-millisecond temporal coding / W. Gerstner та ін. Nature. 1996. Т. 383, № 6595. С. 76–78. URL: https://doi.org/10.1038/383076a0 (дата звернення: 26.11.2025).

[10] Post-Training Quantization for Vision Transformer / Z. Liu та ін. Advances in Neural Information Processing Systems. 2021. Т. 34. С. 28092–28103.

[11] QuantNAS: Quantization-aware Neural Architecture Search For Efficient Deployment On Mobile Device / T. Gao та ін. 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), м. Seattle, WA, USA, 17–18 черв. 2024 р. 2024. С. 1704–1713. URL: https://doi.org/10.1109/cvprw63382.2024.00177 (дата звернення: 26.11.2025).

[12] Li C., Ma L., Furber S. Quantization Framework for Fast Spiking Neural Networks. Frontiers in Neuroscience. 2022. Т. 16. URL: https://doi.org/10.3389/fnins.2022.918793 (дата звернення: 26.11.2025).

[13] Li Deng. The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. IEEE Signal Processing Magazine. 2012. Т. 29, № 6. С. 141–142. URL: https://doi.org/10.1109/msp.2012.2211477 (дата звернення: 26.11.2025).

[14] Xiao H., Rasul K., Vollgraf R. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithm. arXiv. 2017. URL: https://arxiv.org/abs/1708.07747 (дата звернення: 26.11.2025).

[15] Krizhevsky A. Learning Multiple Layers of Features from Tiny Images. 2009. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf (дата звернення: 26.11.2025).

[16] PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation / J. Ansel та ін. ASPLOS '24: 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2, м. La Jolla CA USA. New York, NY, USA, 2024. URL: https://doi.org/10.1145/3620665.3640366 (дата звернення: 26.11.2025).

[17] SpikingJelly: An open-source machine learning infrastructure platform for spike-based intelligence / W. Fang та ін. Science Advances. 2023. Т. 9, № 40. URL: https://doi.org/10.1126/sciadv.adi1480 (дата звернення: 26.11.2025).

[18] Training Full Spike Neural Networks via Auxiliary Accumulation Pathway / G. Chen та ін. arXiv. 2023. URL: https://doi.org/10.48550/arXiv.2301.11929 (дата звернення: 26.11.2025).

Надійшла до редколегії 11.09.2025