https://doi.org/10.31891/2307-5732-2025-351-51
УДК 004.89:004.04:004.09

ПІХ ВОЛОДИМИР
Івано-Франківський національний технічний університет нафти і газу
https://orcid.org/0000-0001-9420-5522
e-mail: v.pikh@nung.edu.ua
ПІХ МАРІЯ
Івано-Франківський національний технічний університет нафти і газу
https://orcid.org/ 0009-0005-4204-1905
e-mail: mariia.pikh@nung.edu.ua
СЛАБІНОГА МАРЯН
Івано-Франківський національний технічний університет нафти і газу
https://orcid.org/ 0000-0002-7296-0356
e-mail: marian.slabinoha@nung.edu.ua
СТРІЛЕЦЬКИЙ ЮРІЙ
Івано-Франківський національний технічний університет нафти і газу
https://orcid.org/ 0000-0002-0105-8306
e-mail: momental@ukr.net
ШЕКЕТА ВАСИЛЬ
Івано-Франківський національний технічний університет нафти і газу
https://orcid.org/ 0000-0002-1318-4895
e-mail: vasyl.sheketa@nung.edu.ua

МЕТОДОЛОГІЯ ОПТИМІЗАЦІЇ НЕЙРОННОЇ МЕРЕЖІ З ІНТЕГРАЦІЄЮ ПЕРЦЕПТРОННИХ КОМПОНЕНТІВ РЕАЛІЗОВАНИХ НА ПЛІС

Запропонована робота присвячена розробці нової методики оптимізації нейронної мережі, що забезпечує висновок у реальному часі на програмованих логічних інтегральних схемах. Основна ідея дослідження полягає у перенесенні частини обчислень з онлайн-висновку на стадію офлайн-навчання за допомогою методів пояснюваного ШІ (ПШІ). Це дозволяє оцінити важливість кожної ознаки, забезпечуючи розрідженість даних та зменшуючи накладні витрати на локальне обчислення. Завдяки цьому локальна нейромережа може бути значно легшою, що суттєво знижує вимоги до пам'яті, енерговитрат та обчислювальних ресурсів, а також дозволяє зберегти високий рівень точності висновку. Методика враховує неоднорідність вхідних даних, що дозволяє відокремити ключові ознаки, які зберігаються локально, від менш важливих, що передаються на віддалений сервер для подальшої обробки. Інтеграція локальних та віддалених нейромереж відбувається через уніфіковану функцію втрат, яка включає як критерій точності, так і вимогу до асиметрії важливості ознак. Використання попередньої обробки екстрактора ознак, а також адаптивне розбиття мережі сприяють досягненню оптимального компромісу між точністю висновку та витратами на обчислення. Дослідження демонструє, що запропонований підхід дозволяє зменшити затримки висновку до десятків мілісекунд, що є критичним для застосувань у IoT, автономних роботах, медичних пристроях та інших системах з обмеженими ресурсами. Також, акцентується увага на аспектах залучення одношарових та багатошарових перцептронів, як первинних компонентів нейронних мереж, що можуть використовуватись на етапах попередньої обробки сигналів та даних, зокрема для оцінювання неоднорідності даних, а також на етапі формування локальних висновків. Окрім того, підвищена енергоефективність і оптимізація використання локальної пам'яті сприяють розширенню можливостей впровадження передових технологій штучного інтелекту на рівні ПЛІС. Методика забезпечує більш точну атрибуцію ознак за допомогою інтегрованих градієнтів, що дозволяє адаптивно налаштовувати параметри як локальної, так і віддаленої мереж, мінімізуючи затримки передачі даних та знижуючи енергоспоживання. Загальний внесок роботи полягає у створенні гнучкої архітектури нейронних мереж, здатної ефективно балансувати між точністю висновку та обчислювальними витратами, що відкриває нові перспективи для практичного використання ШІ у різних галузях сучасної ІТ-індустрії.

Ключові слова: *нейронна мережа, перцептрон, цифрові компоненти, імовірнісні оцінки, програмовані логічні інтегральні схеми (ПЛІС).*

PIKH VOLODYMYR, PIKH MARIYA, SLABINOHA MARIAN,
STRILETSKYI YURII, SHEKETA VASYL
Ivano-Frankivsk National Technical University of Oil and Gas

METHODOLOGY OF OPTIMIZATION OF A NEURAL NETWORK WITH INTEGRATION OF PERCEPTRON COMPONENTS IMPLEMENTED ON FPGA

The proposed work is dedicated to developing a new methodology for optimizing a neural network that enables real-time inference on programmable logic integrated circuits. The main idea of the study is to transfer part of the computations from online inference to the offline training stage using explainable methods. This approach allows for evaluating the importance of each feature, ensuring data sparsity and reducing the overhead of local computations. Consequently, the local neural network can be significantly lighter, which substantially lowers memory requirements, energy consumption, and computational resources, while still maintaining a high level of inference accuracy. The methodology takes into account the heterogeneity of the input data, allowing for the separation of key features that are retained locally from less important ones that are transmitted to a remote server for further processing. Integration of local and remote neural networks is achieved through a unified loss function that includes both the accuracy criterion and the requirement for asymmetry in feature importance. The use of preliminary processing by the feature extractor, along with adaptive network partitioning, helps achieve an optimal balance between inference accuracy and computational costs. The study demonstrates that the proposed approach can reduce inference delays to tens of milliseconds, which is critical for applications in IoT, autonomous robotics, medical devices, and other systems with limited resources. Additionally, attention is focused on the aspects of incorporating both single-

layer and multi-layer perceptrons as primary components of neural networks. These can be used in the stages of preliminary signal and data processing—particularly for evaluating data heterogeneity—as well as in forming local inferences. Furthermore, improved energy efficiency and optimized use of local memory expand the possibilities for implementing advanced artificial intelligence technologies at the FPGA level. The methodology enables more precise feature attribution using integrated gradients, which allows for adaptive tuning of parameters in both local and remote networks, minimizing data transmission delays and reducing energy consumption. Overall, the contribution of this work lies in the creation of a flexible neural network architecture that effectively balances inference accuracy with computational costs, opening up new prospects for the practical application of AI in various sectors of the modern IT industry.

Keywords: *neural network, perceptron, digital components, probabilistic estimates, programmable logic integrated circuits (PLIS).*

Стаття надійшла до редакції / Received 11.04.2025
Прийнята до друку / Accepted 26.04.2025

Постановка задачі

Сучасні нейронні мережі, зокрема багатошарові архітектури, базуються на ключових ідеях, які були запропоновані на етапах формування та дослідженнях структури перцептрону, який став основою для розвитку сучасних підходів у цій сфері. Цифровий *перцептрон* зазвичай вважають простою моделлю, що реалізує механізми кореляційної обробки, однак використання інноваційних підходів, як-от для фрактальної структури даних чи імовірнісних, зокрема ентропійних оцінок даних, може дозволити краще зрозуміти, як структура зв'язків і характеристики вхідних даних впливають на швидкість, ефективність функціонування нейронної мережі.

Зменшення накладних витрат на навчання нейромережі є важливим та актуальним завданням сучасного етапу розвитку ІТ. Використання штучного інтелекту для оцінки важливості конкретної функції є дорогим з точки зору обчислень через часте обчислення градієнтів у кожній ітерації навчання. Прямим пом'якшенням є зменшення кількості таких градієнтних обчислень, але це може вплинути на якість маніпулювання асиметрією. Альтернативно, оскільки стандартне навчання НМ також передбачає операції з градієнтами, можна повторно використовувати ці існуючі градієнти для прискорення побудови оцінки. Гнучкі нейромережі перевершують існуючі схеми, коли доступна пропускна здатність мережі низька. Якщо мережа недоступна або стикається з сильними перешкодами, то все ще може покладатися на локальний прогнозатор для прийняття основних рішень. Оскільки найважливіші функції виконуються локальним предиктором, мережа докладає всіх зусиль, щоб підтримувати точність висновку. Також життєздатним є розгортання більш складних локальних предикторів для підвищення точності в таких екстремальних умовах.

Таким чином *проблема висновку штучного інтелекту на слабких пристроях*, полягає в тому, що статичним моделям важко адаптуватися до нових даних і різних сценаріїв застосування. Замість цього, модель НМ повинна бути негайно перенавчена під час виконання з новими вхідними даними, несучи при цьому мінімальні витрати на обчислення.

Аналіз останніх джерел

Нейронні мережі [1-5] були використані для забезпечення рішення багатьох нових задач, таких як розпізнавання обличчя та мови, відстеження об'єктів, побудови особистих помічників для бізнесу та медичних застосунків. З появою цих програм виникає нагальна потреба застосування висновку нейромережі у реальному часі на невеликих вбудованих пристроях [6-7], щоб забезпечити більш інтелектуальне та оперативне прийняття рішень на цих слабких пристроях. Наприклад, обробка даних на пристроях датчиків безпеки та промислових актуаторах дозволить оперативно реагувати на системні події, а аналіз даних про діяльність людини в режимі реального часу на переносних пристроях може вчасно визначити потенційні ризики для здоров'я. Розгортання моделей НМ на невеликих дронах і роботах є технічною основою автономної навігації цих пристроїв, яка корисна в багатьох сценаріях спостереження за навколишнім середовищем, аварійно-рятувальних робіт і військових сценаріїв. Крім того, висновок НМ у режимі реального часу, якщо він стане можливим на датчиках, що працюють із енергоспоживанням, і пристроях із живленням від радіочастот, може розширити поточний горизонт ШІ до небувалих нових висот.

Однак розгортання НМ на цих невеликих пристроях є дуже складним через невідповідність між слабкими можливостями цих пристроїв і високими обчислювальними вимогами НМ. Наприклад, типові моделі нейромереж містять мільйони параметрів і десятки згорткових шарів і вимагають відповідно сотень мегабайт пам'яті та процесори ГГц-ними характеристиками, щоб досягти затримки висновку рівня мс. Однак такий обсяг обчислювальних ресурсів у понад десятки разів перевищує той, який доступний на мікроконтролерах зараз.

Щоб усунути таку невідповідність, необхідно зменшити складність НМ за допомогою стиснень [8-10] які видаляють зайві ваги та структури мережі.

Однак існуючі схеми в основному націлені на потужні мобільні пристрої (наприклад, смартфони), де достатньо помірного зменшення складності НМ. При адаптації до надзвичайних обмежень ресурсів слабких вбудованих пристроїв надто спрощені НМ зазнають значного зниження точності висновку. Наприклад, коли розмір моделі зменшується в сотні разів, її точність висновку може впасти на десятки відсотків. Навіть із застосуванням технік типу NeuralArchitectureSearch (NAS), яка

знаходить найкращу структуру НМ із заданим обмеженням складності [11-12], втрата точності висновку все ще може становити ті самі десятки відсотків .

Натомість кращим рішенням для уникнення втрати точності виведення є перевантаження обчислень НМ на хмарний сервер. Щоб мінімізувати витрати на розвантаження зв'язку, можна стиснути вхідні дані НМ перед передачею, але ступінь стиснення може бути обмеженим і призвести до високої затримки передачі даних, за допомогою низькошвидкісних бездротових радіостанцій, які використовуються на вбудованих пристроях з метою економії енергії. Також існують підходи щодо розділення НМ [13-14] і використання сегменту локальної нейронної мережі для перетворення вхідних даних у більш стиснуту форму подання ознак перед передачею.

Проте, існуючі схеми розділення НМ, однак, потребують використання дорогої локальної НМ для забезпечення розрідженості функцій і спричинятимуть неприйнятні затримки обчислень на локальному пристрої.

Виклад основного матеріалу

Висновок НМ на пристрої, як частина задачі створення гнучких НМ пов'язана з існуючими зусиллями зі створення легких моделей НМ. Стиснення НМ і скорочення адаптують складні НМ шляхом видалення зайвих ваг і структур. Пошук нейронної архітектури зводиться до формування теоретичної межі шляхом пошуку оптимальної структури НМ за обмеженням складності НМ. За певних обставин, коли бездротове з'єднання недоступне на локальному вбудованому пристрої, і локальний висновок, є єдиним варіантом, ці методи можуть бути корисними для підтримки деяких простих завдань логічного висновку НМ із низькими вимогами до продуктивності. Однак через надзвичайні обмеження ресурсів слабких вбудованих пристроїв ці методи мають обмежені можливості для підтримки більш складних висновків НМ або досягнення висновків НМ у реальному часі.

Задача створення гнучкої НМ також пов'язана з проблемою розвантаження НМ. Ранні зусилля передають стиснуті необроблені дані на сервер. Щоб покращити стисливість даних, у пізніших роботах використовується локальна НМ, яка перетворює необроблені дані в розріджені функції, але локальна НМ повинна бути складною, щоб забезпечити розрідженість функцій. Будучи ортогональною, гнучка НМ, потребує розвантаження даних на кілька серверів, щоб дослідити неоднорідність обчислювальної потужності серверів.

У зв'язку з широким розповсюдженням додатків штучного інтелекту існує нагальна потреба оптимізувати висновок нейронної мережі у реальному часі на невеликих вбудованих пристроях, проте розгортання мережі та досягнення високої продуктивності висновку на таких невеликих пристроях є складним завданням через їхні надзвичайно обмежені можливості. Хоча розділення та розвантаження мережі можуть сприяти такому розгортанню, вони не здатні мінімізувати локальні витрати на вбудованих пристроях. Замість цього ми пропонуємо вирішити цю проблему за допомогою гнучкого розвантаження мережі, яке переносить необхідні обчислення в розвантаженні сегменти мережі з рівня онлайн-висновків до рівня офлайн-навчання. В даному дослідженні представлено нову техніку розвантаження НМ, яка забезпечує висновок у режимі реального часу на слабких вбудованих пристроях за допомогою методів штучного інтелекту, що дає можливість забезпечити розрідженість функцій на етапі навчання та мінімізувати витрати на онлайн-обчислення та зв'язок. Отримані результати показують, що затримка логічного висновку в таких випадках менша, ніж у існуючих схем, що гарантує своєчасне використання сенсорних даних на вбудованих пристроях. Це також зменшує споживання ресурсів локального пристрою, не погіршуючи точність логічного висновку загалом.

Таблиця 1

Порівняльний аналіз підходів до нейромережевого висновку на ПЛІС

		Локальна обчислювальна складність	**Витрати пам'яті**	**Затрати на перетворення даних**	**Втрати точності висновку**	**Затрати на процес навчання**
Локальний висновок	Стиснення	дуже висока	висока	немає	висока	висока
	Відтинання	дуже висока	висока	немає	висока	висока
	Нейро-архітектура	висока	середня	немає	середня	дуже висока
Віддалений висновок	JPEG \ MPEG	низька	низька	висока	низька	низька
	Метод стиснення	середня	низька	середня	середня	середня
Розбиття гнучкої нейромережі		висока	середня	низька	низька	середня
Гнучка нейромережа		дуже низька	низька	дуже низька	дуже низька	середня

Ключова причина виділених обмежень полягає в тому, що ці схеми незалежно застосовують однаковий підхід до навчання до всіх вхідних даних, і, отже, потребують достатньої потужності представлення в локальній НМ для найгіршого випадку вхідних даних.

Щоб усунути це обмеження та практично ввімкнути висновок НМ на надзвичайно слабких пристроях з мінімальною затримкою, ми прагнемо побудувати нову техніку, яка змінює обґрунтування розділення та розвантаження НМ із фіксованого на гнучкий та орієнтований на дані. Наша основна ідея полягає в тому, щоб включити знання про неоднорідність різних вхідних даних у навчання мережі, щоб необхідні обчислення для забезпечення розрідженості функцій перемістилися з онлайн-виводу на офлайн-навчання.

Одним з варіантів оцінювання такої неоднорідності може бути використання одношарового чи багатошарового перцептрону як окремого цифрового компоненту на платформі з обмеженими обчислювальними ресурсами. Функціонально такий перцептрон реалізує операцію кореляції чи згортки, що дозволяє оцінити наявність чи відсутність періодичних складових, зміну динаміки процесів, зміщення фази тощо. Іншим варіантом є залучення однієї з функцій ймовірнісного оцінювання вхідних даних як основної функції агрегації перцептрону замість суми. Зокрема, якщо використати оцінку інформаційної ентропії, то можна отриманим міру невизначеності чи хаотичності даних. Таким чином, згадані підходи дають можливість попереднього оцінювання неоднорідності даних відносно нескладними обчислювальними ресурсами.

Реалізації простої НМ, як комбінації перцептронів на платформі програмованих логічних інтегральних схем (ПЛІС) передбачає створення апаратної моделі, яка забезпечує паралельну обробку даних і оптимізацію обчислювальних ресурсів, тобто використати можливостей ПЛІС для досягнення високої швидкості обчислень та енергоефективності. В такій структурі, кожен нейрон цифрового перцептрону може бути реалізований як окремий модуль на ПЛІС, що дозволить паралельно обчислювати вихідні сигнали, використовуючи окремі обчислювальні блоки. Крім того, для кожного нейрона необхідно передбачити блоки зберігання вагових коефіцієнтів і зміщень, які можуть оновлюватися під час навчання або завантажуватися як готові параметри. Слід зауважити, що завдяки можливості перепрограмування ПЛІС, архітектуру перцептрону, як окремого цифрового компоненту, можна адаптувати до різних задач, змінюючи кількість шарів або кількість нейронів у них без потреби втручання у готові схемні рішення.

Більш конкретно, ми інтерпретуємо таку неоднорідність як важливість різних характеристик даних для логічного висновку НМ і використовуємо методи eXplainable AI (ПШІ), щоб чітко оцінити таку важливість під час навчання. Таким чином, як показано на рисунку, онлайновий висновок може забезпечити розрідженість функцій, лише стискаючи та передаючи менш важливі функції, без залучення дорогих обчислень НМ. Важливі характеристики, з іншого боку, зберігаються на локальному пристрої та можуть бути сприйняті легкою НМ з низькою складністю. Прогнози від локальної НМ і віддаленої НМ, зрештою, об'єднуються на локальному пристрої для висновку

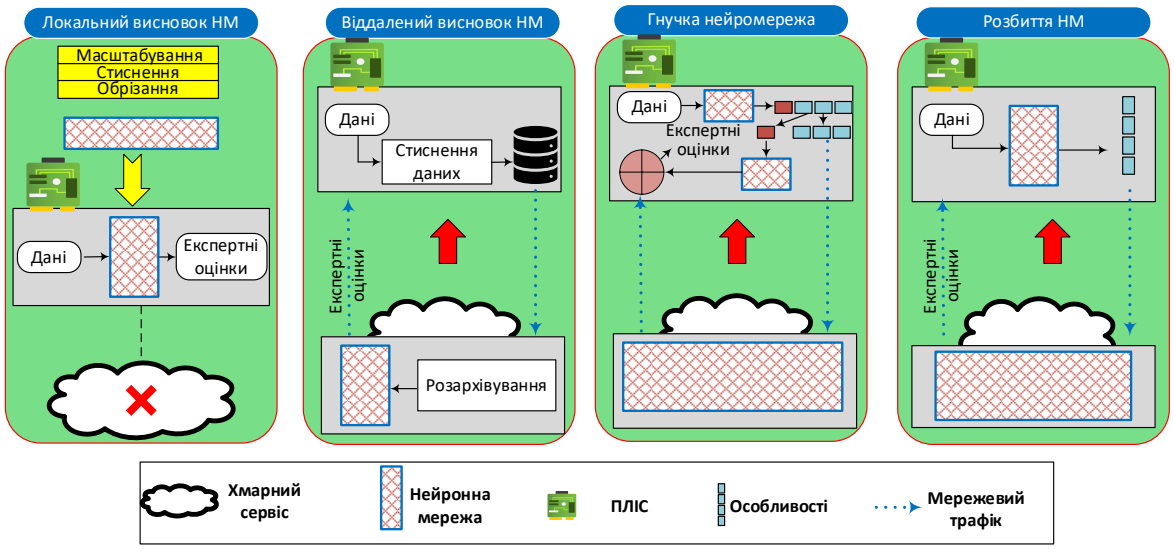

Рисунок 1 - Загальний контекст побудови та впровадження гнучких нейромереж

Основна проблема використання гнучких НМ на практиці, однак, полягає в тому, що різні характеристики даних можуть мати подібне значення для виведення НМ. У цьому випадку розрідженість менш важливих функцій буде зменшена, що призведе до нижчої стисливості даних, і більше функцій також потрібно зберегти на локальному пристрої, спричиняючи додаткову затримку обчислення. Щоб вирішити цю проблему та одночасно мінімізувати витрати локального вбудованого пристрою на обчислення та зв'язок, основний підхід полягає в цілеспрямованому маніпулюванні важливістю функцій даних за допомогою нелінійного перетворення у просторі високовимірних

функцій, щоб гарантувати спотворення такого розподілу важливості між різними функціями. Іншими словами, лише кілька ознак роблять більший внесок у висновок НМ. Доцільним є оперування асиметрією за допомогою надзвичайно легкого екстрактора ознак і спільного навчання екстрактора ознак із локальними та віддаленими НМ, щоб забезпечити точність висновку.

Важливо, що саме методика гнучких нейромереж є першою технікою, яка забезпечує висновок НМ у реальному часі на вбудованих пристроях із надзвичайно слабкими можливостями в обчисленнях і зв'язку.

Виконується ефективне перенесення необхідних обчислень в розвантаженні НМ з онлайн-виводу на офлайн-навчання, використовуючи методи ПШІ, які дозволяють полегшити контроль розрідженості функцій під час виконання. Нові методи штучного інтелекту, які використовують ПШІ для явного маніпулювання важливістю різних характеристик даних у висновках НМ, дозволяють забезпечити ефективність розділення та розвантаження НМ. Забезпечуючи нерівномірність розподілу важливості між різними функціями, дозволяються гнучкі компроміси між точністю та вартістю виведення НМ на вбудованих пристроях, не зазнаючи додаткових витрат на обчислення чи зберігання. Виконання реалізації гнучкої НМ на відповідній платі і сервері з графічним процесором дозволить оцінити продуктивність НМ на різних популярних наборах даних у різних системних умовах. За результатами таких імплементацій можна зробити наступні висновки: 1) гнучка НМ працює в режимі реального часу, і в порівнянні з існуючими схемами дозволяє зменшити затримку виведення НМ і обмежує таку затримку в заданих межах для більшості наборів даних. Таким чином, підтримується висновок НМ у реальному часі на слабких вбудованих пристроях, гарантуючи, що сенсорні дані завжди можуть бути використані вчасно; 2) порівняно з існуючими схемами поділу НМ, гнучка НМ забезпечує подібну точність висновку, але досягає значно більшої розрідженості функцій і таким чином, така висока розрідженість зменшує обсяг передачі даних у розвантаженні НМ; 3) порівняно з поточними схемами, виведення НМ на вбудованих пристроях, гнучка НМ зменшує споживання локальної енергії в рази, споживаючи відповідно менше пам'яті та в рази менше місця для зберігання даних; 4) гнучка НМ є адаптивною, що суттєво мінімізує зниження продуктивності висновку НМ у різних налаштуваннях вбудованих пристроїв і системних умовах, навіть із надзвичайно низькою обчислювальною потужністю та пропускною здатністю бездротового зв'язку.

Щоб зменшити витрати зв'язку на розвантаження НМ, інтуїтивно зрозумілим методом є стиснення необроблених даних перед передачею, але сильне стиснення спотворить важливу інформацію в даних і, отже, вплине на точність висновку НМ. Натомість поточні підходи до розділення НМ покращують стисливість даних шляхом вилучення більш стислих форм представлень ознак із необроблених вхідних даних. Однак, як показано на рисунку з двома репрезентативними підходами до розділення, хоча ці схеми можуть досягти подібних ступенів стиснення з мінімальним впливом на точність висновку НМ, їх виділення ознак є дуже дорогим з точки зору обчислень. Вищенаведені обмеження визначають весь дизайн рішення, який забезпечує кращу стисливість даних шляхом оцінки важливості різних функцій даних під час офлайн-навчання. Грунтуючись на таких знаннях про важливість функцій, під час онлайнового висновку ми можемо явно забезпечити розрідженість менш важливих функцій з мінімальними локальними обчислювальними витратами. Щоб оцінити таку важливість ознак, класичні підходи на основі збурень вимірюють, як змінюється точність висновку НМ після введення шуму в функції в усіх навчальних даних, але не можуть точно оцінити важливість ознак для окремих зразків даних. Механізми на основі уваги підтримують таку індивідуальну оцінку шляхом додавання додаткового генератора ваги в тренування НМ, але потребують адаптації структури генератора ваги до кожної моделі НМ і, отже, можуть бути неточними в деяких моделях НМ. Нещодавні дослідження eXplainable AI (ПШІ) покращують точність оцінки важливості ознак, пропонуючи інструменти атрибуції, які кількісно співвідносять кожну вхідну змінну з виходами НМ під час навчання. Наприклад, типові інструменти ПШІ, такі як інтегровані градієнти (IG), як показано на рисунку, передають ряд лінійних інтерполяцій між вхідними змінними та простою базовою лінією в мережу. Потім для вхідної змінної вони обчислюють кожен градієнт її інтерполяції щодо виходу НМ (наприклад, оцінки достовірності) і накопичують ці градієнти для вимірювання важливості цієї вхідної змінної. Таким чином, ці інструменти ПШІ є надійними та застосовними до будь-якої моделі ШІ без додаткових модифікацій. Однак одним із ключових обмежень існуючих інструментів ПШІ є те, що його точність оцінки важливості ознак будується на попередньому точному висновку НМ. Якщо вихідні дані мережі неоднозначні наприклад, через неадекватне навчання, ПШІ може дати оманливі оцінки, оскільки градієнти, обчислені на основі вихідних даних мережі, є дуже випадковими. У гіршому випадку така випадковість може спричинити неправильне оцінювання всіх ознак за їхньою важливістю. Це обмеження спонукає використовувати попередньо підготовлену еталонну модель НМ у навчанні для випадку гнучкої мережі, щоб забезпечити правильну оцінку ПШІ щодо важливості її функцій.

Виходячи з важливості функцій, оцінених в ПШІ, ефективність розвантаження гнучкої НМ залежить від нерівності розподілу такої важливості між різними функціями. Чим вищою є така асиметрія, тим менше ознак відіграє домінуючу роль у виведенні НМ, і, отже, ми можемо забезпечити більшу розрідженість менш важливих ознак без погіршення точності висновку НМ. Однак, як показано

на рисунку, який ілюструє такий розподіл важливості різних вибірок даних у наборі даних, нерівність може не завжди існувати в усіх вхідних даних. Крім того, як показано на рисунку, коли ми вимірюємо асиметрію як відношення нормалізованої важливості найкращих характеристик, така класифікація вибірок даних у наборах даних становить прийнятне значення. Така низька асиметрія у вхідних даних спонукає розробляти нові структури НМ, які цілеспрямовано маніпулюють і посилюють таку асиметрію у вилученні ознак, мінімізуючи при цьому вплив такої асиметрії на точність висновку НМ.

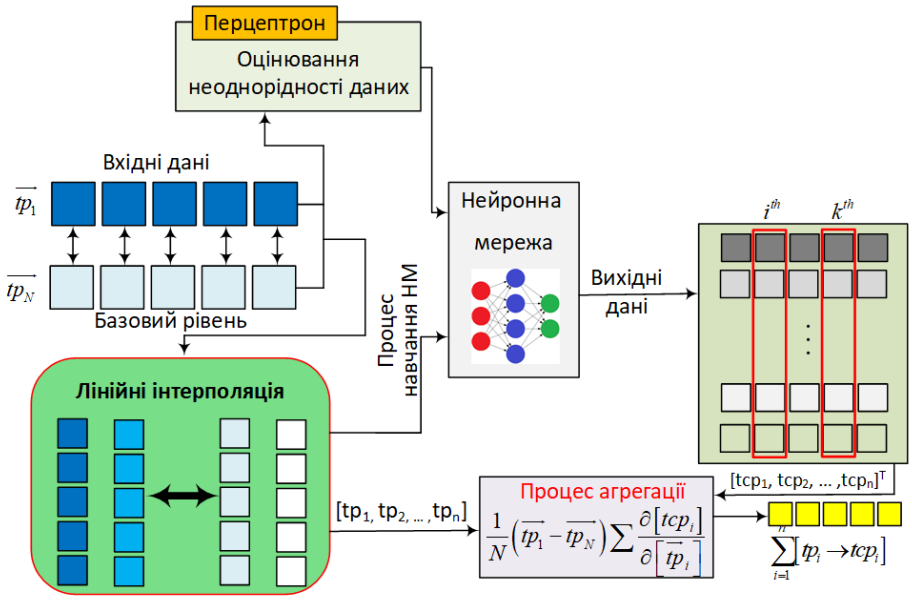

Рисунок 2 -Схема інтеграції градієнтів

Як показано на рисунку гнучка НМ розділяє нейронну мережу на локальну та віддалену мережу. В режимі онлайн-виведення середовище виконання гнучкої НМ використовує легкий екстрактор функцій на локальному вбудованому пристрої для надання вхідних даних функцій: локальна НМ зберігає основні особливості, які мають високу важливість, щоб зробити локальний прогноз, який потім поєднується з прогнозом віддаленої НМ на основі інших менш важливих функцій для остаточного висновку. Таким чином, складність локальної НМ може бути зведена до мінімуму без погіршення точності виведення, а висока розрідженість може бути забезпечена під час стиснення та передачі менш важливих функцій на сервер. Під час офлайн-навчання гнучка НМ спільно навчає екстрактор функцій, локальну та віддалену мережу з уніфікованою функцією втрати. Зокрема, екстрактор ознак навчається відповідати вимогам користувача щодо нерівності важливих ознак, так що нормалізована важливість ознак верхнього рівня має перевищувати задане порогове значення. Під час процесу навчання дотримання цієї вимоги еквівалентно застосуванню нелінійних перетворень до вихідного вектора ознак у просторі ознак високої розмірності.

На основі такого дизайну гнучка НМ може балансувати між точністю та вартістю висновку НМ, регулюючи необхідну асиметрію важливості функцій. Чим вище асиметрія відповідно, тим менше споживання ресурсів буде локальним пристроєм через більшу стисливість менш важливих функцій, що передаються, але на висновок НМ це більше впливає через нелінійне перетворення екстрактора функцій у просторі ознак. На практиці, коли те саме середовище виконання для гнучкої НМ навчається для конкретного вбудованого пристрою, користувач може адаптивно вибирати різні компроміси відповідно до сценаріїв застосування та умов локальних ресурсів, не витрачаючи додаткові локальні обчислювальні ресурси чи ресурси зберігання для підтримки кількох моделей НМ або прийняття різних стратегій навчання.

Для того, щоб маніпулювати важливістю виділених ознак і відповідати вимогам асиметрії, основний підхід гнучких НМ полягає в тому, щоб включити як точність висновку, так і поточну асиметрію важливості ознаки в уніфіковану функцію втрат під час навчання. Якщо говорити точніше, на кожному етапі навчання гнучкої НМ виконується передавання поточного набору функцій, витягнутих за допомогою екстрактора функцій, до модуля інструментів ПШІ, який оцінює та виводить важливість кожної функції для висновку НМ. Таким чином, асиметрія важливості ознаки включається у функцію втрат.

Таке впорядкування має важливе значення для онлайнового висновку, де інструмент ПШІ недоступний, і наявні першочергові k особливостей з високою значущістю, а отже, вони завжди повинні знаходитися у фіксованих каналах вилученого вектора ознак. Завдяки такому впорядкуванню функцій ми можемо далі вказати екстрактору функцій підвищити важливість заданих функцій і, отже, забезпечити необхідну асиметрію. З іншого боку, знання про фіксоване розміщення ознак у заданому

векторі ознак також уможливить оперативне розділення функцій для локальних і віддалених висновків, не залучаючи жодних додаткових обчислень чи ручних зусиль під час виконання.

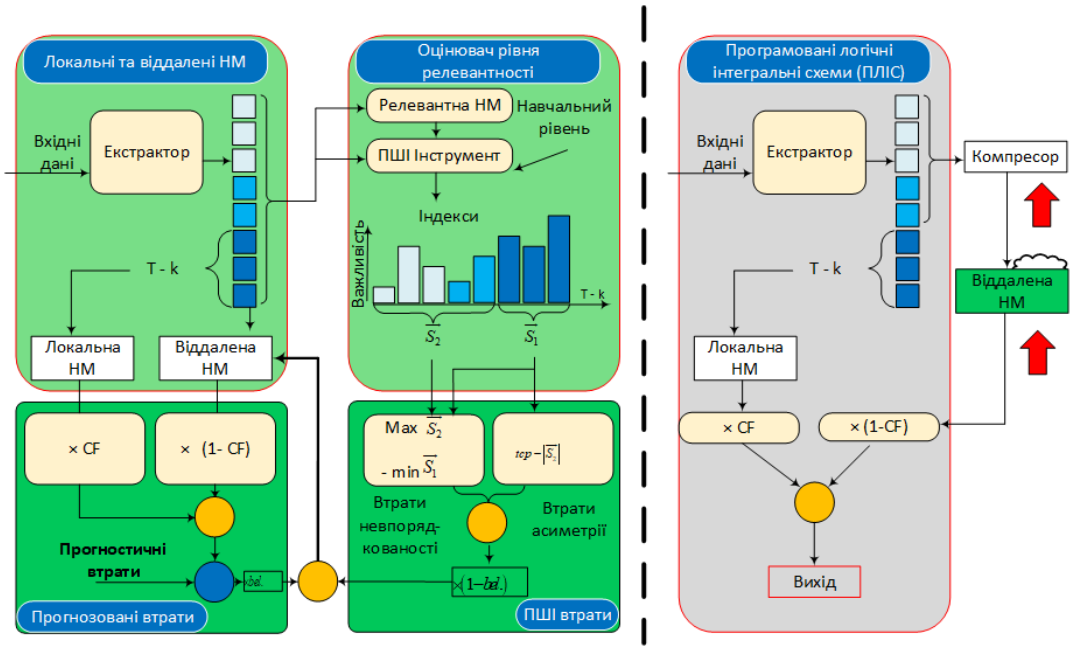

Рисунок 3 –Імплементація процесу вбудовування

Оцінка важливості функції ПШІ вимагає заздалегідь добре навченої НМ, щоб надати правильні мітки висновку. Отже, щоб забезпечити якість навчання, слід запровадити еталонну модель НМ, яка була попередньо навчена для того самого навчального завдання з достатньою потужністю представлення, щоб надати вихідні дані інструменту ПШІ за допомогою витягнутих функцій із екстрактора функцій. Щоб надалі уникнути будь-якої можливої неоднозначності, ми порівнюємо кожен висновок, зроблений еталонною НМ, із навчальною міткою та використовуємо її в оцінці ПШІ, лише якщо еталонна НМ робить правильні передбачення. Виконується спільне навчає локальної та віддаленої мережі із інструментом виділення ознак, щоб гарантувати, що вони можуть надавати точні прогнози на основі виділених функцій із спотвореним розподілом важливості. Однак, оскільки екстрактор функцій потрібно розгортати на локальному пристрої і, отже, він має бути дуже легким, він може не мати достатньої потужності представлення, щоб досягти мети навчання на початковому етапі навчання, і спільне навчання, отже, може зіткнутися з неочікувано високими труднощами або навіть не бути збіжним взагалі. Наприклад, як показано на рисунку, таке спільне навчання на заданому наборі даних, якщо воно починається з нуля, є дуже нестабільним, якщо в екстракторі ознак не використовується достатня кількість згорткових шарів. Щоб уникнути таких труднощів у навчанні, пропонований підхід полягає у попередній обробці засобу вилучення функцій та ініціалізації його мережевих ваг перед спільним навчанням із локальними та віддаленими мережами. Таким чином, спільне навчання не розпочнеться з нуля, а замість цього з більш усталеної стадії з меншою двозначністю, і, отже, матиме менші вимоги до початкової потужності представлення екстрактора ознак.

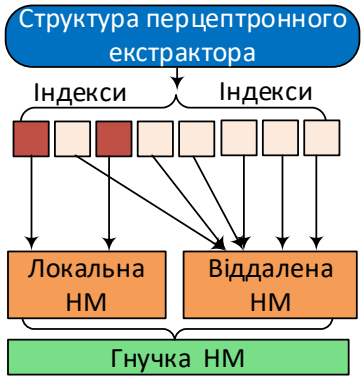

Рисунок 4 - Попередня обробка екстрактора ознак

Точніше, упорядкування функцій, передбачене втратою розладу може бути важко виконати екстрактором функцій на початковому етапі навчання. Замість цього, як показано на рисунку, ми

вибираємо k початкових каналів у вихідному векторі ознак, де, швидше за все, будуть розташовані K перших ознак з високою важливістю. Потім ми використовуємо відповідні k ознак як вхідні дані для локальної НМ. Мережа об'єднує передбачення, зроблені локальними та віддаленими мережами через зважене підсумовування, щоб отримати кінцевий висновок на локальному вбудованому пристрої. Порівняно з іншими альтернативами на основі НМ, такими як додавання додаткового рівня НМ для комбінування, ми використовуємо це рішення оскільки, обчислення таких зважених сум від точки до точки набагато легше, ніж операції НМ, і додає незначні витрати на обчислення в локальному пристрої. По-друге, вихідні дані локальних і віддалених НМ завжди відповідають однаковій кількості вирівняних каналів функцій, і підсумовування точка-точка зберігає таке вирівнювання. З іншого боку, використання рівня НМ (наприклад, повністю пов'язаного або згорткового) для об'єднання цих двох виходів може сплутати їх разом і порушити таке вирівнювання, що погіршить точність кінцевого висновку. Основна складність такої комбінації, однак, полягає в тому, що вихідні дані локальної НМ і віддаленої НМ можуть бути не в тому самому масштабі і, отже, можуть призвести до додаткової втрати точності логічного висновку, оскільки деякі малі, але важливі вихідні значення в одній НМ можуть бути перевищені великими значеннями в іншій НМ. Щоб вирішити цю проблему, пропоноване рішення полягає в тому, щоб включити вагу підсумовування в процедуру спільного навчання. Будучи таким самим, як і інші параметри НМ, даний коефіцієнт також навчається за допомогою градієнтного зворотного зв'язку з використанням алгоритмів стохастичного градієнтного спуску. Однак, через велику різницю між складністю локальної та віддаленої НМ, підготовка, ймовірно, буде спрямована на віддалену НМ та ігноруватиме локальну.

Навчене значення завантажується в середовище виконання гнучкої НМ на локальному пристрої. У реальних налаштуваннях, коли екстрактор функцій неправильно оцінює важливість деяких функцій через можливу неточність у ПШІ, користувач може гнучко налаштувати стратегію НМ щодо розбиття НМ під час виконання, переконфігурувавши відповідне значення, щоб зменшити втрату точності висновку.

Оскільки оцінка важливості функції ПШІ базується на накопиченні градієнтів під час навчання а, отже, недоступна під час онлайн-виведення, гнучка НМ гарантує, що її екстрактор функцій завжди генеруватиме задану кількість особливостей із найвищою важливістю в перших k каналах у вихідному векторі ознак, як показано на рисунку, щоб середовище виконання НМ на локальному вбудованому пристрої могло правильно ідентифікувати їх для кожного із вхідних даних під час виведення.

Таким чином, мінімізація цієї втрати гарантує, що витягнуті функції завжди сортуються в порядку спадання їх важливості. Однак суворе дотримання такого порядку спадання у створеному векторі ознак вимагає високої потужності представлення в екстракторі ознак або додає додаткову плутанину в навчанні, якщо використовуваний екстрактор ознак є надто легким. Щоб продемонструвати це, ми проводимо попередні експерименти за допомогою екстрактора функцій моделі на наборі даних. Як показано на рисунку, застосування такого порядку спадання у вихідному векторі ознак гарантовано знижує точність висновку. Натомість ми пом'якшуємо навчальну мету, зменшуючи кількість змінюваних функцій. Як показано на рисунку, ми не вимагаємо, щоб усі функції були відсортовані в порядку спадання їх важливості, натомість вимагаємо лише, щоб важливість будь-якої функції з перших k була вищою за важливість будь-якої іншої. Якщо під час навчання буде виявлено будь-яке порушення, штраф буде зв'язаний із НМ для оновлення параметрів. Базуючись на цій легкій меті навчання, ми будуємо нашу втрату розладу, яка буде відмінна від нуля, лише якщо будь-яке порушення впорядкування функцій відбувається під час навчання. Теоретично ця функція втрат розладу ознак майже завжди диференційована і може бути легко включена в звичайну процедуру навчання.

Щоб виконати вимогу асиметрії, ми хочемо, щоб кумулятивна нормалізована важливість ознак перевищувала заданий поріг, і, отже, ми зможемо визначити втрату асиметрії. Потім доцільно об'єднати втрату розладу та втрату асиметрії разом, щоб оцінити втрати процесу навчання для маніпуляції асиметрією.

Вибір початкових каналів ознакдосягнення більшої асиметрії може зменшити вплив втрати прогнозу у зворотному зв'язку навчання і, отже, погіршити точність висновку НМ. Навпаки, ми спостерігаємо, що помірне значення коефіцієнта могло б ефективно наблизити вимогу до асиметрії з мінімальним впливом на точність висновку НМ. Крім того, можна також прийняти методи балансування втрат НМ для адаптивного налаштування коефіцієнта під час виконання.

Важливим моментом є попередня обробка екстрактора функцій, де вибираються початкові k каналів функцій як вхідні дані для локальної НМ у спільному навчанні. Інтуїтивно зрозуміло, що ми можемо випадковим чином вибрати k каналів функцій і вказати екстрактору функцій виробляти найважливіші функції в цих каналах, використовуючи втрату розладу. Однак такий довільний вибір принесе серйозні труднощі в навчанні, що призводитиме до низької якості процесу навчання. Як показано на рисунку, НМ матиме труднощі з навчанням вже із початкових етапів, що зрештою призводитиме до поганої конвергенції. Замість цього ми робимо такий вибір каналу на основі ймовірності того, що одна з першокласних k особливостей з високою значущістю знаходиться в каналі, і обчислюємо таку ймовірність на основі навчальних даних. Точніше кажучи, така ймовірність кожного

каналу обчислюється кумулятивно з усіх N вибірок даних у навчальному наборі даних і збільшується на 1/N щоразу, коли в каналі розташовано k найважніших ознак вибірки даних. Як показано на рисунку, така попередня обробка може значно зменшити труднощі процесу навчання та забезпечити відповідну гарантовану якість навчання.

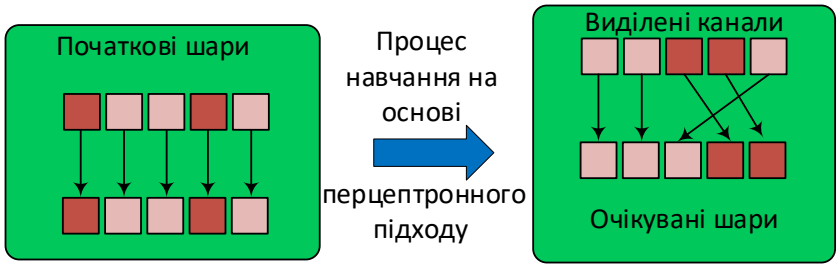

Рисунок 5 - Навчання шару відображення

Після вибору початкових k каналів ми очікуємо, що процес спільного навчання зможе поступово забезпечити необхідне впорядкування функцій, через втрату розладу. Щоб полегшити це завдання, як показано на рисунку, під час навчання гнучкої нейромережі ми додаємо додатковий рівень відображення між екстрактором функцій і локальною мережею, а також інструктуємо процес навчання, щоб гарантувати, що перші k важливих функцій знаходяться в перших k каналах вихідного вектора ознак. Після завершення навчання цей шар відображення буде відкинуто, і для висновку використовуватиметься лише екстрактор ознак.

Як показано на рисунку, ми використовуємо плату як локальний вбудований пристрій, який широко використовується як обчислювальна платформа в поточних дослідженнях на пристрої. Він підтримує гнучке масштабування частоти процесора, щоб забезпечити різну кількість обчислювальної потужності пристрою. Крім того, оскільки нейронна мережа для мікроконтролерів даної серії була офіційно підтримана спільнотою TensorFlow, можна вважати, що використання цих мікроконтролерів для реалізації та оцінки рівня гнучкості НМ може краще виправдати практичні переваги гнучкості НМ порівняно з використанням інших мікроконтролерів.

Як показано на рисунку, наше офлайн-навчання гнучкої НМ реалізовано за допомогою бібліотеки TensorFlowPython, і ми перетворили локальну НМ з а допомогою TensorFlowLiteConverter. Потім ця модель перетворюється на статичний бінарний масив для кращої ефективності пам’яті на локальному пристрої. Ми використовуємо середовище виконання для виконання вибраної моделі на платі. Щоб ще більше зменшити затримку обчислення локальної мережі, ми об’єднуємо оригінальне середовище виконання із розширенням, що забезпечує додаткове прискорення кількох вибраних операцій мережі на пристроях виділеного класу. З іншого боку, НМ зберігає повну точність і виконується середовищем виконання TensorFlowPython на сервері.

На платі ми використовуємо реалізації програмного забезпечення на C++ і налаштування вбудованого обладнання. Щоб стиснути менш важливі функції перед передачею, ми спочатку використовуємо квантування на основі навчання, а потім застосовуємо стандартне стиснення. Стиснуті функції доставляються до модуля WiFi, а модуль WiFi передає ці функції на сервер зі швидкістю порядку Мбіт/с. На сервері ми пишемо спеціальний сценарій Python для зв’язку з платою через API загального сокета та перевіряємо цілісність отриманих функцій, застосовуючи їх до віддаленої НМ.

У наших оцінках, щоб відповідати обмеженням локальних ресурсів вбудованого пристрою, ми створили екстрактор функцій із двома згортковими шарами, кожен із яких має задану кількість каналів виведення. Локальна НМ має мінімальну складність і містить один глобально-середній шар об’єднання та один щільний шар. Віддалена НМ побудована шляхом видалення першого згорткового рівня з моделі. У всіх оцінках розміри екстрактора функцій, локальної НМ і віддаленої НМ залишаються фіксованими, але ми змінюємо ступінь стиснення під час передачі набору менш важливих функцій від локальної до віддаленої. Зокрема, оскільки дизайн мережі приймає різні вхідні роздільні здатності для різних наборів даних, можна гарантувати, що завжди використовується однакова роздільна здатність зображення серед усіх інших підходів, включаючи гнучку НМ, щоб проводити справедливі порівняння між різними підходами.

В наших оцінках усі результати експерименту усереднюються за всіма наборами даних тестування. Ми порівнюємо гнучку НМ з базовою лінією висновку лише по краю та трьох існуючих підходах до висновку НМ, які охоплюють обидві категорії локального висновку та розділення НМ: 1) лише граничний висновок: усі локальні дані стискаються компресором і передаються на сервер для висновку; 2) уся НМ працює на локальному вбудованому пристрої, і система оптимально виявляє структуру НМ відповідно до обмежень ресурсів пристрою щодо складності НМ; 3) кодувальник на основі НМ вбудовано в локальний пристрій для перетворення необроблених даних або функцій у більш стиснуту форму, кодер навчається з обмеженням розрідженості наскрізним способом; 4) окрім розбиття НМ, структури раннього виходу включені в НМ для адаптивного налаштування складності НМ для висновку під час виконання.

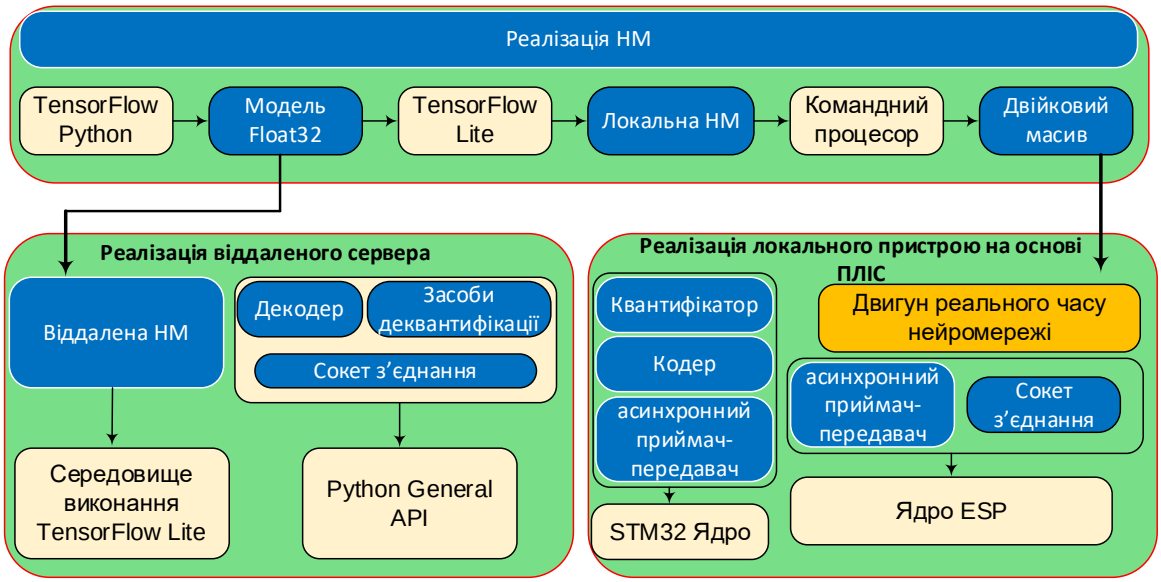

Рисунок 6 - Реалізація гнучкої НМ

Як передумови, ми спочатку оцінюємо якість і вартість навчання гнучкої НМ. Як показано на рисунку, під час процедури навчання гнучка НМ демонструє дуже схожу швидкість збіжності навчання з точки зору точності тесту та втрат порівняно зі звичайним навчанням на непідготовлених наборах даних. Ці результати показують, що, незважаючи на те, що додане впорядкування функцій і маніпулювання нерівністю збільшують складність навчання, гнучка НМ все ще може забезпечити швидку навчальну конвергенцію з відповідним дизайном функції втрат і попередньою обробкою екстрактора ознак. З іншого боку, з додатковими обчисленнями важливості функції з використанням ПШІ і відповідним залученням додаткового зворотного зв'язку для процесу навчання ми спостерігаємо збільшення часу для кожного етапу навчання в гнучкій НМ. Однак, оскільки навчання екстрактора функцій, локальних і віддалених мереж НМ проводиться в автономному режимі, таке збільшення часу не вплине на онлайн-продуктивність мережі на слабких вбудованих пристроях. Скорочення такого часу навчання можна здійснити або за допомогою потужнішого обчислювального обладнання в формі потужніших графічних процесорів, або більш легких інструментів ПШІ відповідно. Загалом, точність виведення НМ можна підвищити, використовуючи більш складні НМ, що, у свою чергу, призводить до довшої затримки висновку. Для спрощення порівнянь ми налаштовуємо складність НМ існуючих підходів так, щоб різниця між їх точністю висновку та точністю висновку гнучкої НМ завжди була в заданих межах. У цих випадках ми порівнюємо затримку наскрізного висновку гнучкої НМ з їхньою. Слід звернути увагу, що для підходів локального висновку, затримка висновку визначатиметься лише часом обчислення локальної НМ.

Для підходів щодо розділення НМ, затримка висновку складається з 1) часу обчислення локальної НМ; 2) часу локального обчислення для стиснення даних; 3) часу передачі по мережі; 4) часу віддаленої НМ для декомпресії та обчислення даних. Результати на рисунку показують, що гнучка НМ здатна зменшити затримку наскрізного висновку в рази порівняно з усіма існуючими підходами, зберігаючи аналогічну точність висновку. Зокрема, таку затримку в більшості наборів даних можна ефективно контролювати в межах десятків мс, що можна порівняти з інтервалом вибірки багатьох вбудованих сенсорних пристроїв. У результаті гнучка НМ може ефективно підтримувати висновок НМ у режимі реального часу на слабких вбудованих пристроях, гарантуючи, що згенеровані сенсорні дані завжди можуть бути використані вчасно. Більш конкретно, на рисунку показано, що зменшення затримки гнучкої НМ в основному відбувається за рахунок нижчого часу обчислення локальної мережі, який можна суттєво скоротити. Порівняно з близькими аналогами, які мають використовувати складну локальну мережу, щоб забезпечити розрідженість функцій, при впровадженні ПШІ, гнучка НМ дозволяє досягти більшої розрідженості функцій за допомогою набагато легшої локальної мережі і відповідного їй екстрактора функцій. Затримка виведення буде набагато вища, ніж у інших підходів, через складну НМ, яка повністю виконується на вбудованому пристрої.

З іншого боку, незважаючи на те, що граничний висновок викликає мінімальну затримку локального обчислення, він матиме низьку швидкість бездротового з'єднання на локальному пристрої, що призводить до значно більшої затримки бездротової передачі через низьку стисливість даних. Таким чином, загальна наскрізна затримка граничного висновку вища, ніж у випадку гнучкої НМ. Така більша розрідженість функцій, з іншого боку, також

призводить до значного скорочення часу передачі по мережі.

Як було показано вище, таке зменшення на деяких наборах даних, може перевищувати порогове значення. Це зниження, навіть менше ніж на десятки відсотків, може бути важливим у деяких

сценаріях IoT, де пристрої IoT бездротово підключені до магістральної мережі 5G і, отже, їм потрібно буде здійснювати платежі на основі використання постачальника послуг 5G.

Вплив ступеню стиснення визначається тим фактом, що оскільки мережа працює найкраще серед трьох існуючих підходів для порівняння, ми додатково порівнюємо його продуктивність з випадком гнучкої НМ, коли застосовуємо різні рівні стиснення для передачі даних на віддалений сервер. Результати на рисунку для виділених наборів даних показують, що гнучка НМ завжди може досягти вищої точності виведення із застосуванням того самого рівня стиснення завдяки більш гнучкому та ефективнішому застосуванню розрідженості функцій, що призводить до кращої стисливості. Зокрема, коли застосовуються дуже високі рівні стиснення, система відчуває значне зниження точності через обмежену потужність представлення кодера, але таке зниження у випадку гнучкої НМ є набагато нижчим.

Вплив перезважування прогнозів при формуванні передбачень, зроблених локальною НМ і віддаленою НМ, об'єднуються для вихідного висновку за допомогою деякого налаштовуваного параметра. Результати на рисунку щодо наборів типових даних показують, що точність висновку НМ значно впаде, якщо використовуються сильно зміщені значення параметра близькі до 0 або 1. Це пояснюється тим, що використання дуже малого значення параметра зменшує внесок важливих характеристик і, отже, може пропустити ключову інформацію для висновку. З іншого боку, збільшення значення виділеного параметра накладає більшу частину завдання логічного висновку на локальну НМ, яка може бути недостатньо складною для досягнення високої точності. Натомість ми робимо висновок, що максимальної точності висновку можна досягти для заданого значення параметра і для виділених наборів даних. Слід зауважити, що оптимальне значення параметра залежить від характеристик даних у навчальному наборі даних. На практиці це значення можна або спільно навчити в автономному режимі за допомогою екстрактора функцій і мережевих представлень, або налаштувати вручну в режимі онлайн на основі конкретних характеристик даних для кращої точності висновку.

Слід оцінити кількість локальних ресурсів на вбудованому пристрої, які споживаються при формуванні висновку гнучкої НМ. Такі локальні ресурси включають 1) локальне живлення акумулятора та 2) локальну пам'ять і флеш-пам'ять. Для коректного порівняння між різними схемами, подібними до типових експериментів, слід зберігати розрив між точністю висновку різних схем у межах кількох відсотків.

Ми вимірюємо обсяг споживання енергії локальним пристроєм на один прогін висновку НМ як середнє значення за задану кількість прогонів висновку. Таке енергоспоживання включає як витрати на обчислення локальної НМ, так і вартість передачі даних через WiFi. Як показано на рисунку, оскільки гнучка НМ використовує дуже легкий екстрактор функцій і локальну мережу, але досягає ще більшої розрідженості функцій за допомогою цих легких структур мережі, її час виконання споживає менше локальної енергії як для обчислень, так і для зв'язку, що призводить до значно вищої енергоефективності, особливо при використанні на менших наборах даних, його енергоефективність принаймні в рази вища.

Як показано на рисунку, завдяки низькій складності екстрактора функцій і мережевих пристроїв гнучка НМ споживає пам'ять і сховище локального пристрою менше допустимого порогового значення. Така висока ефективність пам'яті та зберігання особливо важлива для слабких вбудованих пристроїв із дуже обмеженими ресурсами зберігання, оскільки це дозволяє розгортати набагато потужніші моделі мережевої топології на цих пристроях і, отже, забезпечити надійну підтримку складніших програм мережі. Маніпулювання нерівністю є наріжним каменем ефективного розвантаження гнучкої НМ. Щоб дослідити ефективність маніпуляції з нерівністю в такому випадку , застосовуються різні вимоги до нерівності важливості ознак, змінюючи значення k відповідним чином, щоб зберегти відсоток особливостей з найвищою важливістю в локальній НМ. Таким чином, вимагається, щоб нормалізована важливість цих функцій зростала відповідно в напрямі максимального порогового значення. Спершу ми перевіряємо, чи може маніпуляція з асиметрією гнучкої НМ адекватно досягти необхідної асиметрії в витягнутих функціях. На рисунках показано, що гнучка НМ завжди може досягти необхідної цілі асиметрії з незначною різницею. Важливо, що шуканому наборі даних, досягнута асиметрія навіть вища за цільову. Це демонструє, що шукана функція втрати асиметрії є достатньо ефективною.

На рисунках показано, що, оскільки така ж кількість важливих функцій зберігається в локальній НМ, застосування вищих нерівностей для цих функцій може збільшити розрідженість функцій загалом, а для тих що залишилися, зробити їх менш важливими функціями, таким чином зменшуючи затримку передачі мережі. У той же час, така більша асиметрія також впливає на точність виведення НМ, як показано на рисунках. Основна причина полягає в тому, що, коли нормалізована важливість локально збережених функцій є занадто високою, полегшена локальна НМ може не мати достатньої потужності представлення для правильного сприйняття цих функцій, що призводить до додаткової втрати точності. Однак, оскільки локальні та віддалені НМ навчаються спільно, таке падіння точності завжди може бути обмежене в межах кількох відсотків.

Наведені аргументи демонструють, що гнучка НМ може ефективно маніпулювати нерівністю важливості функцій у різних налаштуваннях, отже, дозволяючи гнучкі компроміси між точністю та

вартістю виведення НМ. Збереження додаткових функцій на локальних пристроях може допомогти пом'якшити таке падіння точності за рахунок додаткових локальних обчислень НМ.

Коефіцієнт розподілу залежатиме від обчислювальної потужності конкретного пристрою та характеристик навчального набору даних. Загалом вважається, що оптимальним вибором дизайну є збереження важливих функцій на локальному пристрої та вимога, щоб нормалізована важливість цих функцій була максимальна відповідно. Така вимога асиметрії була використана у всіх представленнях.

Щоб вивчити вплив частоти процесора на продуктивність гнучкої НМ, ми регулюємо частоту плати, налаштовуючи коефіцієнт масштабування тактової частоти. Тут ми припускаємо, що більшість вбудованих пристроїв будуть використовуватися виключно для висновку НМ під час виконання пов'язаних обчислювальних завдань. Отже, ми вважаємо, що центральний процесор локального пристрою може бути повністю використаний для висновку НМ.

Як показано на рисунку, хоча затримка виведення збільшується, коли частота падає, таке збільшення завжди невелике, завдяки його полегшеній функції екстрактора і локальній НМ. Порівняно, існуючі схеми зазнають значно більшого зниження продуктивності через виконання дорогого локального НМ на вбудованому пристрої.

Через місцеві обмеження щодо енергоспоживання та форм-фактори не всі вбудовані пристрої оснащені високошвидкісними модулями WiFi. Натомість багатьом із них доводиться використовувати вузькосмугові радіостанції з низьким споживанням енергії, такі як Bluetooth. Результати на рисунках показують, що навіть можливі випадки, коли доступна пропускна здатність бездротової мережі становить лише сотні Кбіт/с що нижче навіть, ніж yWiFi. Відповідно, висока розрідженість функцій гнучкої НМ гарантує такий вплив, і ми використовуємо відповідні інструменти ПШІ для побудови гнучкої НМ. Як було встановлено, мережа залишається стабільною з різними виборами ПШІ. Гнучка НМ працює трохи краще, оскільки вона агрегує більше інтерполяцій градієнтів вихідних даних НМ. З іншого боку, гнучка реалізаця є відповідно дорожчою з точки зору обчислень, оскільки зазвичай вимагає суттєво більше градієнтних обчислень для отримання кожного вимірювання важливості.

Висновки

У цьому дослідженні ми представляємо нову техніку, яка змінює обґрунтування *розділення та розвантаження* нейромереж із фіксованої на гнучку та орієнтовану на дані шляхом використання методів штучного інтелекту. Гнучкі НМ забезпечують точний висновок НМ у режимі реального часу на надзвичайно слабких пристроях шляхом переміщення необхідних обчислень у розвантаженні НМ з онлайн-виведенням на офлайн-навчання, що загалом суттєво зменшує затримку висновку НМ.

Пропонована гнучка НМ використовує поточні інструменти атрибуції(присвоєння атрибутів) для оцінки

важливості кожної функції. Традиційні підходи до атрибуції застосовують випадкову перестановку або нульові маски до конкретних вхідних змінних і використовують індуковану вихідну варіацію для емпіричного визначення важливості. Підходи на основі ваг вбудовують рівень зважування на основі навчання в НМ, а вивчені ваги використовуються для вказівки важливості ознаки. Однак ці вимірювання чутливі до різних структур НМ і не завжди можуть забезпечити точну оцінку. Новітні методи штучного інтелекту забезпечують більш точні та надійні інструменти атрибуції. Вони приймають градієнти вихідних даних НМ щодо вхідних змінних, щоб визначити їх важливість, тобто яка є більш дрібнозернистою та ми зможемо чітко визначити у відсотках, який внесок кожної вхідної змінної у вихідне значення.

Реалізація цифрового компоненту здатного формувати локальні висновки ускладнена обмеженими обчислювальними ресурсами, що потребує досліджень в частині структурних рішень базових компонентів нейронних мереж. Зокрема варіантом покращення функціональних можливостей цифрового перцептрону може бути застосування не тільки амплітудного а також ймовірнісного підходу, зокрема для аналізу інформаційної ентропії даних. У межах розвитку теорії інформації було запропоновано та досліджено властивості різноманітних аналітичних формул для обчислення оцінок інформаційної ентропії, адаптованих для обробки сигналів і даних із різними ймовірнісними розподілами станів у системі, а також запропоновано ряд апаратних рішень на основі програмованих логічних інтегральних схем, що суттєво спрощує імплементацію на етапах формування локальних висновків.

Методи штучного інтелекту в основному використовуються для аналізу характеристик даних і розуміння поведінки НМ, але їх використання для підвищення ефективності розвантаження буде темою наступних досліджень проблеми.

Література

1. Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., &Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon*, *4*(11).

2. Taye, M. M. (2023). Theoretical understanding of convolutional neural network: Concepts, architectures, applications, future directions. *Computation*, *11*(3), 52.

3. Dastres, R., &Soori, M. (2021). Artificial neural network systems. *International Journal of Imaging and Robotics (IJIR)*, *21*(2), 13-25.

4. Li, Z., Liu, F., Yang, W., Peng, S., &Zhou, J. (2021). A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, *33*(12), 6999-7019.

5. Madhiarasan, M., &Louzazni, M. (2022). Analysis of artificial neural network: Architecture, types, and forecasting applications. *Journal of Electrical and Computer Engineering*, *2022*(1), 5416722.

6. Weng, O. (2021). Neural network quantization for efficient inference: A survey. *arXivpreprint arXiv:2112.06126*.

7. Lin, J., Zhu, L., Chen, W. M., Wang, W. C., Gan, C., &Han, S. (2022). On-device training under 256kb memory. *Advances in Neural Information Processing Systems*, *35*, 22941-22954.

8. Yao, K., Cao, F., Leung, Y., &Liang, J. (2021). Deep neural network compression through interpretability-based filter pruning. *PatternRecognition*, *119*, 108056.

9. Liang, T., Glossner, J., Wang, L., Shi, S., &Zhang, X. (2021). Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing*, *461*, 370-403.

10. He, Y., &Xiao, L. (2023). Structured pruning for deep convolutional neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, *46*(5), 2900-2919.

11. Liu, Y., Sun, Y., Xue, B., Zhang, M., Yen, G. G., &Tan, K. C. (2021). A survey on evolutionary neural architecture search. *IEEE transactions on neural networks and learning systems*, *34*(2), 550-570.

12. Ren, P., Xiao, Y., Chang, X., Huang, P. Y., Li, Z., Chen, X., &Wang, X. (2021). A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, *54*(4), 1-34.

13. Yu, M., Jiang, Z., Ng, H. C., Wang, W., Chen, R., &Li, B. (2021, July). Gillis: Serving large neural networks in server less functions with automatic model partitioning. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)* (pp. 138-148). IEEE.

14. Tsay, C., Kronqvist, J., Thebelt, A., &Misener, R. (2021). Partition-based formulations for mixed-integer optimization of trained ReLU neural networks. *Advances in neural information processing systems*, *34*, 3068-080.