НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Системи штучного інтелекту

Анотована історія сучасного штучного інтелекту та глибокого навчання

Семінар

Виконав:
Олександр Ковальов
Група:
ІМ-51мн
Курс:
1

25 листопада 2025 р.

1 Вступ

Історія штучного інтелекту (ШІ) та глибокого навчання (Deep Learning) часто подається спрощено, ніби все почалося лише у 2012 році з появою потужних відеокарт. Однак справжня історія набагато глибша і сягає корінням не лише у 20-те століття, а й у часи Лейбніца та Гаусса. Цей реферат базується на дослідженнях професора Юргена Шмідхубера [1] та розкриває справжніх піонерів технологій, які сьогодні керують нашим світом.

2 Математичний фундамент: Від Лейбніца до Розенблатта *(1676 - 1958)*

Цей розділ має на меті зруйнувати міф про те, що штучний інтелект – це винахід виключно комп'ютерної ери. Ми покажемо, що «двигун» сучасного ШІ був винайдений ще в епоху перук і камзолів.

2.1 Ланцюгове правило: Готфрід Вільгельм Лейбніц *(1676)*

Коли ми говоримо про навчання нейромережі сьогодні, ми використовуємо термін *Backpropagation* (зворотне поширення помилки). Але якщо відкинути комп'ютерний жаргон, математично це – застосування ланцюгового правила диференціювання.

У 1676 році, працюючи над створенням математичного аналізу, Готфрід Вільгельм Лейбніц сформулював правило, яке дозволяє знаходити похідну складеної функції. У сучасних нейромережах це виглядає так: у нас є функція помилки E, яка залежить від виходу мережі y, який залежить від ваг w. Щоб зрозуміти, як змінити вагу w, щоб зменшити помилку, ми рахуємо:

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial w} \tag{1}$$

Шмідхубер наголошує: Лейбніц – це «перший програміст» концептуального рівня. Він описав двійкову систему числення, якою користуються всі комп'ютери, і дав нам математику для тренування нейромереж.

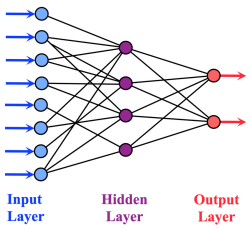
Без відкриття 1676 року ми б не могли ефективно обчислювати градієнти в глибоких мережах сьогодні. Це фундамент «Credit Assignment Problem» – визначення того, який саме нейрон винен у помилці.

2.2 Перша «нейромережа»: Гаусс, Лежандр і метод найменших квадратів *(бл. 1805)*

Чи була нейромережа до комп'ютерів? Шмідхубер стверджує: так. Близько 1800 року математики Адрієн-Марі Лежандр та Карл Фрідріх Гаусс (незалежно один від одного) розробили метод найменших квадратів (Linear Least Squares).

Чому ми можемо назвати це «нейромережею»?

1. **Архітектура**: Це по суті один шар лінійних нейронів.
2. **Навчання**: Мета – мінімізувати суму квадратів помилок між передбаченими даними (лінією регресії) та реальними точками.
3. **Результат**: Це приклад «дрібного навчання» (Shallow Learning). У нас немає прихованих шарів, немає глибокої ієрархії ознак, але є процес «підгонки» параметрів під дані.



2.3 Френк Розенблатт і Перцептрон *(1958)*: Ейфорія та обмеження

Переносимося у 20-те століття. У 1958 році Френк Розенблатт презентує Перцептрон Mark I. Це була не просто програма, а фізична машина з купою дротів та потенціометрів.

У чому був прорив? Розенблатт запропонував правило навчання, яке ітеративно оновлювало ваги. Якщо машина помилялася, ваги коригувалися пропорційно до вхідного сигналу. Це викликало величезний ажіотаж. Газета New York Times тоді писала, що це «ембріон комп'ютера, який зможе ходити, говорити, бачити, писати і усвідомлювати своє існування».

Чому це НЕ було глибоким навчанням? Багато хто плутає перцептрон із глибокими мережами.

Перцептрон Розенблатта мав вхідний шар, шар асоціативних елементів і вихідний шар. Але навчався лише останній шар. Попередні шари мали фіксовані, випадкові ваги. Оскільки навчався лише один шар, це все ще було «дрібне навчання» *(Shallow Learning)*, аналогічне методам Гаусса, але з функцією активації (порогом).

Шмідхубер підкреслює: Розенблатт думав про багатошарові перцептрони (MLP), але він не знав, як навчити внутрішні шари. Він не мав працюючого алгоритму *(Backpropagation)* для багатошарових структур.

Це обмеження призвело до першої «зими штучного інтелекту» після того, як Марвін Мінскі та Сеймур Пейперт у 1969 році опублікували книгу «Perceptrons», де математично довели, що одношаровий перцептрон не здатен вирішити навіть просту логічну задачу XOR (виключне АБО).

3 Народження Глибокого Навчання в Києві *(1965–1971)*

У той час як західний науковий світ у 1960-х роках поступово занурювався у скепсис щодо можливостей нейронних мереж, спричинений обмеженнями одношарових перцептронів, в Україні (тоді УРСР) відбувалася справжня технологічна революція. Саме в Києві було закладено фундамент того, що сьогодні світ називає *(Deep Learning)* (глибоке навчання).

3.1 Піонерська робота Олексія Івахненка та Валентина Лапи

У 1965 році професор Олексій Григорович Івахненко разом із колегою Валентином Лапою опублікував фундаментальну працю «Кібернетичні передбачаючі пристрої». Це сталося за чотири роки до сумнозвісної критики Мінського та Пейперта, яка на десятиліття заморозила дослідження нейромереж у США.

Івахненко запропонував підхід, який кардинально відрізнявся від спроб Розенблатта. Замість того щоб намагатися налаштувати ваги у фіксованій структурі (що було неможливо для глибоких мереж без ефективного методу зворотного поширення помилки), він розробив алгоритм, який дозволяв мережі рости та самоорганізовуватися шар за шаром.

3.2 Метод групового врахування аргументів (МГУА / GMDH)

Центральним винаходом Івахненка став Метод групового врахування аргументів (МГУА), відомий англійською як *Group Method of Data Handling* (GMDH). Цей метод став першим у світі успішним алгоритмом для навчання глибоких нейронних мереж з довільною кількістю шарів.

Суть методу полягала в ітеративному відборі найкращих моделей. Кожен нейрон у такій мережі являв собою не просто суматор, а складнішу обчислювальну одиницю, що реалізовувала квадратичний поліном від двох входів (поліном Колмогорова-Габора).

Математично функція активації такого нейрона для двох вхідних змінних описується наступним рівнянням:

$$y = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 \tag{2}$$

Де змінні означають наступне:

- y – вихідний сигнал нейрона;
- x_1, x_2 – вхідні змінні (ознаки);

- $w_0, \ldots, w_5$ – вагові коефіцієнти, які оптимізувалися методом найменших квадратів.

Процес навчання відбувався наступним чином:

1. **Генерація**: Перший шар генерував безліч нейронів, комбінуючи різні пари вхідних даних.
2. **Селекція**: Мережа оцінювала ефективність кожного нейрона на окремій перевірочній вибірці даних (мінімізуючи середньоквадратичну помилку).
3. **Еволюція**: Ті нейрони, які показували найменшу помилку, пропускалися до наступного шару. Їхні виходи ставали входами для нейронів наступного рівня.

Таким чином, складність моделі зростала автоматично, допоки точність розпізнавання або прогнозування продовжувала покращуватися. Це дозволяло уникнути перенавчання — проблеми, яка є актуальною і для сучасних алгоритмів.

3.3 Перший у світі Глибокий Перцептрон *(1971)*

Кульмінацією цих досліджень стала стаття 1971 року [2], в якій Івахненко описав успішне навчання глибокої багатошарової мережі. У той час, як західні вчені ледве працювали з 1-2 шарами, київська система вже мала 8 шарів обробки інформації.

За словами Юргена Шмідхубера, ця робота є першим задокументованим випадком глибокого навчання *(Deep Learning)* в історії. Мережа Івахненка успішно вирішувала задачі ідентифікації систем та розпізнавання образів, фактично випередивши свій час на кілька десятиліть.

3.4 Чому західний світ «проспав» цей момент

Незважаючи на очевидні успіхи, метод Івахненка не отримав миттєвого глобального визнання, яке він мав би мати. Цьому сприяло кілька факторів:

1. **Мовний та геополітичний бар'єр**: Основні публікації виходили російською та українською мовами в радянських журналах «Автоматика». Переклади доходили до західних читачів із запізненням, а Холодна війна обмежувала науковий обмін.
2. **Обчислювальна складність**: Для комп'ютерів 1970-х років перебір тисяч варіантів поліномів був надзвичайно ресурсномістким завданням.
3. **Домінування іншої парадигми**: Після публікації книги Мінського та Пейперта *(1969)* західна спільнота зосередилася на символьному ШІ *(Symbolic AI)* та логічному програмуванні, вважаючи нейромережі тупиковою гілкою еволюції.

Тим не менш, саме київська школа кібернетики довела, що глибокі архітектури здатні навчатися, заклавши фундамент для майбутніх проривів 21-го століття.

4 Справжня історія Зворотного Поширення Помилки (Backpropagation) *(1960–1970)*

Одним із найпоширеніших міфів у історії штучного інтелекту є твердження, що метод зворотного поширення помилки *(Backpropagation)* був винайдений у 1986 році групою вчених на чолі з Девідом Румельхартом та Джеффрі Хінтоном. Хоча їхня публікація в журналі *Nature* дійсно зробила цей метод відомим серед комп'ютерних фахівців, з математичної та історичної точки зору це було «перевідкриттям».

Справжня історія цього фундаментального алгоритму починається значно раніше і пов'язана не стільки з психологією чи біологією, скільки з керуванням ракетами та чистою математикою.

4.1 Ракетна наука: Генрі Келлі та Артур Брайсон *(1960–1961)*

Ще на початку 1960-х років, у розпал космічних перегонів, дослідники шукали методи оптимізації траєкторій польоту ракет.

У 1960 році Генрі Келлі, а в 1961 році Артур Брайсон описали метод, який базувався на принципах динамічного програмування. Вони вивели алгоритм «аналітичного градієнта», який дозволяв оптимізувати параметри складних систем керування поетапно, рухаючись від кінця процесу до його початку.

Фактично, це і був *Backpropagation*, але застосований до неперервних систем керування, а не до дискретних нейронних мереж. Однак, для повноцінного використання в комп'ютерних науках не вистачало формалізації саме для дискретних обчислень.

4.2 Сеппо Ліннаінмаа та «Зворотний режим автоматичного диференціювання» *(1970)*

Ключовий момент, який часто ігнорується підручниками, настав у 1970 році. Фінський дослідник Сеппо Ліннаінмаа *(Seppo Linnainmaa)* у своїй магістерській дисертації (а згодом і в публікації 1976 року) вперше чітко описав зворотний режим автоматичного диференціювання *(Reverse Mode of Automatic Differentiation)* [3].

Саме цей метод лежить в основі навчання всіх сучасних нейромереж, від ChatGPT до систем комп'ютерного зору. Ліннаінмаа показав, як можна ефективно обчислювати похідні складеної функції, представленої у вигляді комп'ютерного графа обчислень.

Чому це відкриття є критичним? Воно вирішило проблему ефективності. Якби ми використовували «прямий режим» диференціювання, обчислювальні витрати зростали б пропорційно кількості вхідних параметрів. Для сучасних мереж із мільярдами параметрів це було б неможливо.

Метод Ліннаінмаа довів фундаментальне співвідношення витрат:

$$\text{Cost}(\nabla f) \approx K \cdot \text{Cost}(f) \tag{3}$$

Де:

- $\text{Cost}(\nabla f)$ – вартість обчислення градієнта (всіх похідних);
- $\text{Cost}(f)$ – вартість одного прямого проходу (обчислення функції);
- K – невелика константа (зазвичай від 2 до 4).

Це означає, що обчислення поправок для всіх ваг мережі займає приблизно стільки ж часу, скільки й один запуск мережі. Без цього математичного факту глибоке навчання було б технічно неможливим.

4.3 Відновлення історичної справедливості щодо 1986 року

Чому ж тоді 1986 рік вважається роком народження *Backpropagation*?

Стаття Румельхарта, Хінтона та Вільямса «Learning representations by back-propagating errors» [4] показала застосування цього старого математичного методу саме до нейронних мереж і, що найважливіше, продемонструвала, що це працює на практиці для створення внутрішніх репрезентацій даних.

Однак Шмідхубер наполягає: ігнорувати внесок Ліннаінмаа *(1970)* або Пола Вербоса (який у 1974 році у своїй дисертації також застосував цей метод до нейромереж) – це помилка. Сучасний ШІ стоїть на плечах математиків 60-х та 70-х років, які дали нам інструмент автоматичного диференціювання.

Таким чином, *Deep Learning* – це не винахід 21-го століття, а результат ефективної реалізації алгоритмів, відомих ще півстоліття тому, на сучасному апаратному забезпеченні.

5 Еволюція архітектур: Згорткові мережі та перші рекурентні моделі *(1979–1989)*

Після закладення математичного фундаменту *(Backpropagation)* та перших експериментів з глибокими структурами (МГУА Івахненка), наступним логічним кроком стала спеціалізація архітектур. Дослідники зрозуміли, що для обробки зображень та звуку потрібні специфічні структури мереж, натхненні біологією. Саме в цей період, у 1980-х роках, були створені архітектури, які сьогодні лежать в основі систем комп'ютерного зору *(Computer Vision)* та розпізнавання мови.

5.1 Куніхіко Фукусіма та Неокогнітрон *(1979)*

Сьогоднішній бум штучного інтелекту значною мірою базується на згорткових нейронних мережах (CNN). Часто їхню популяризацію пов'язують з ім'ям Яна Лекуна *(1989, 1998)*, проте архітектурна основа цих мереж була розроблена в Японії за десятиліття до того.

У 1979 році Куніхіко Фукусіма в лабораторіях NHK представив Неокогнітрон *(Neocognitron)* [5]. Це була перша глибока нейромережа, архітектура якої була ідентичною до сучасних CNN.

Натхненням для Фукусіми стали нобелівські лауреати з нейрофізіології Девід Хьюбел та Торстен Візел. У 1959 році вони відкрили, що зорова кора мозку котів має два типи клітин:

- **Прості клітини (Simple cells):** реагують на конкретні орієнтації ліній (наприклад, вертикальні чи горизонтальні краї) у певному місці поля зору.
- **Складні клітини (Complex cells):** реагують на ті ж самі ознаки, але інваріантні до їхнього точного розташування (зміщення).

Фукусіма інженерно реалізував це відкриття, створивши мережу з чергуванням двох типів шарів:

1. **S-шари (Simple layers):** Виконували операцію згортки (convolution). Вони виділяли ознаки.
2. **C-шари (Complex layers):** Виконували операцію підвибірки (downsampling), яку ми сьогодні називаємо *Pooling* (наприклад, Max-Pooling або Average-Pooling).

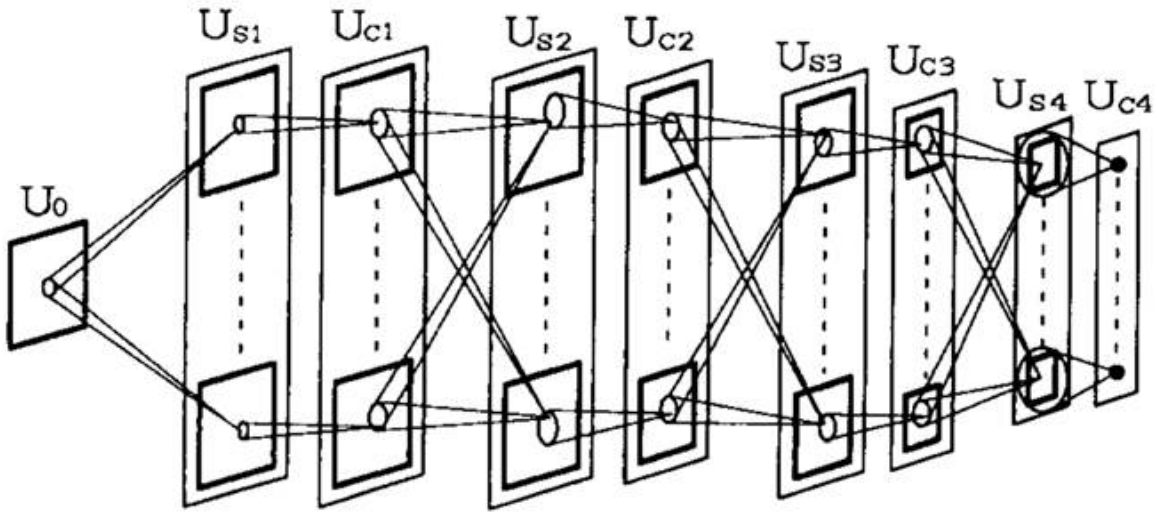


Рис. 1: Архітектура Неокогнітрона Фукусіми (1979), що демонструє чергування S-шарів та C-шарів.

Фактично, Неокогнітрон мав усе необхідне для глибокого навчання, окрім методу навчання. Фукусіма використовував локальні правила навчання без вчителя, а не Backpropagation. Тим не менш, Юрген Шмідхубер наголошує: саме Фукусіма є батьком архітектури CNN. Ян Лекун пізніше (у 1989 році) взяв цю архітектуру і застосував до неї метод зворотного поширення помилки, створивши знамениту мережу LeNet.

5.2 TDNN та внесок Алекса Вайбеля *(1987–1989)*

Якщо Фукусіма працював із зображеннями (2D-сигнали), то для звуку (1D-сигнали, що змінюються у часі) потрібен був інший підхід.

У 1987 році Алекс Вайбель (Alex Waibel) представив **TDNN** (Time-Delay Neural Networks – Нейронні мережі з часовою затримкою). Це була перша у світі згорткова мережа, яка навчалася за допомогою *Backpropagation*.

Ключова ідея Вайбеля полягала у використанні принципу розділення ваг *(Weight Sharing)* у часі.

- Уявіть, що ви хочете розпізнати слово «Привіт». Воно може бути вимовлене швидко або повільно, на початку запису або в кінці.
- TDNN використовувала одні й ті самі ваги нейронів, скануючи вхідний звуковий спектрограму вікном фіксованого розміру.

Математично це еквівалентно одновимірній згортці. Якщо x_t — вхідний сигнал у момент часу t, а w – ваги фільтра, то вихід y_t обчислюється як:

$$y_t = \sum_{k=0}^{K-1} w_k \cdot x_{t-k} \tag{4}$$

Ця робота мала критичне значення, оскільки довела, що глибокі мережі з Backpropagation можуть досягати надлюдської точності у складних задачах (в даному випадку – розпізнавання фонем). Це стало предтечею сучасних систем обробки аудіо.

5.3 Проблема послідовностей та перехід до рекурентності

І Неокогнітрон, і TDNN мали спільне обмеження: вони були мережами прямого поширення (Feedforward). Вони мали фіксоване «вікно» сприйняття. Але людська мова чи текст мають довільну довжину, і контекст може залежати від слів, сказаних набагато раніше.

У 1980-х роках дослідники почали активно розробляти **Рекурентні нейронні мережі (RNN)**. На відміну від звичайних мереж, RNN мають «пам'ять» — зворотні зв'язки, які дозволяють передавати інформацію з попереднього кроку на наступний.

Найпростіша формула RNN виглядає так:

$$h_t = \sigma(W_{hh}h_{t-1} + W_{xh}x_t + b) \tag{5}$$

Де h_t — прихований стан (пам'ять) у момент часу t, який залежить від попереднього стану h_{t-1} та поточного входу x_t.

Хоча ідея була елегантною, на практиці такі мережі було неможливо навчити на довгих послідовностях через проблему, яку ми розглянемо у наступному розділі – проблему зникаючого градієнта. Саме це стало головним викликом для ШІ наприкінці 80-х років.

6 Фундаментальні проблеми та прорив LSTM *(1990-mi)*

Початок 1990-х років ознаменувався новою кризою в галузі нейромереж. Дослідники зіткнулися з тим, що хоча теоретично рекурентні мережі (RNN) могли обробляти послідовності будь-якої довжини, на практиці вони не могли навчитися зв'язкам, розтягнутим у часі більш ніж на 10 кроків. Це явище отримало назву «проблема зникаючого градієнта». Вирішення цієї проблеми в лабораторії Шмідхубера призвело до створення найуспішнішої архітектури нейромереж 20-го століття – LSTM.

6.1 Формулювання проблеми «зникаючого градієнта» *(1991)*

У своїй дисертації 1991 року Юрген Шмідхубер (а також Зепп Хохрайтер у своїй дипломній роботі) формалізували причину, чому глибокі мережі так важко тренувати.

Суть проблеми полягає в математиці зворотного поширення помилки крізь час *(Backpropagation Through Time – BPTT)*. Коли ми обчислюємо градієнт помилки для перших шарів (або перших кроків у часі), ми змушені множити похідні функцій активації та матриці ваг багато разів підряд (відповідно до ланцюгового правила).

Якщо розглядати спрощену лінійну модель, градієнт поводиться як геометрична прогресія:

$$\frac{\partial E}{\partial w} \propto \prod_{t=1}^{T} w_{rec} \tag{6}$$

Де w_{rec} – рекурентна вага, а T – кількість кроків у часі.

- Якщо $|w_{rec}| < 1$, то при великому T добуток прямує до нуля ($0.9^{100} \approx 0.00002$). Градієнт **зникає**, і мережа перестає вчитися.
- Якщо $|w_{rec}| > 1$, добуток зростає експоненціально. Градієнт **вибухає** (exploding gradient), що призводить до нестабільності.

Це відкриття показало, чому стандартні RNN, придумані у 80-х, були фактично недієздатними для складних задач.

6.2 Long Short-Term Memory (LSTM) *(1997)*

У 1997 році Зепп Хохрайтер та Юрген Шмідхубер опублікували статтю «Long Short-Term Memory» в журналі *Neural Computation* [6]. Вони запропонували геніальне інженерне рішення: якщо градієнт зникає при множенні, давайте створимо шлях, де градієнт не множиться, а додається.

Основою LSTM є концепція **Constant Error Carousel (CEC)** — каруселі постійної помилки. Це спеціальна комірка пам'яті, яка зберігає свій стан c_t без змін, якщо на неї не діють зовнішні фактори.

Для керування цією пам'яттю були введені «ворота» *(gates)* — спеціальні нейронні шари з сигмоїдною активацією (значення від 0 до 1), які діють як фізичні крани:

1. **Input Gate (i_t):** Вирішує, скільки нової інформації записати в комірку.
2. **Forget Gate (f_t):** Вирішує, скільки старої інформації забути (додано Герсом у 1999).
3. **Output Gate (o_t):** Вирішує, яку частину пам'яті передати на вихід.

Математично оновлення стану комірки c_t виглядає як лінійна комбінація старої пам'яті та нової інформації $\tilde{c}_t$:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{7}$$

Завдяки доданню (+) замість множення, градієнт може безперешкодно протікати крізь тисячі кроків у часі. **Значення:** До появи Трансформерів у 2017 році, LSTM була домінуючою архітектурою для всього: від розпізнавання мови на Android та iOS до перекладача Google Translate і генерації тексту.

6.3 Попередники Трансформерів: Fast Weights та Self-Attention *(1991–1993)*

Сучасний бум Великих Мовних Моделей (LLM) типу GPT базується на архітектурі *Transformer*, ключовим елементом якої є механізм уваги *(Attention)*. Шмідхубер аргументовано стверджує, що ці ідеї також мають коріння в його лабораторії початку 90-х.

Fast Weights (Швидкі ваги, 1991). У стандартній нейромережі ваги змінюються повільно (під час навчання) і фіксуються під час роботи. Шмідхубер запропонував концепцію «швидких ваг», де одна нейромережа генерує або змінює ваги іншої мережі *на льоту*, залежно від вхідних даних (контексту).

Це математично еквівалентно тому, що роблять Трансформери. У механізмі *Self-Attention* (самоуваги) матриця уваги обчислюється динамічно для кожного входу:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{8}$$

Шмідхубер вказує, що його роботи 1992–1993 років описували «лінеаризовану самоувагу» (Linearized Self-Attention), яка є окремим випадком сучасних трансформерів. Таким чином, сучасні LLM є логічним продовженням (і масштабуванням на кращому залізі) ідей «програмованих нейромереж» з початку 90-х.

7 Ера GPU та комп'ютерного зору *(2010–2012)*

Якщо алгоритми глибокого навчання (Backpropagation, CNN, LSTM) були винайдені ще у 20-му столітті, чому революція сталася лише після 2010 року? Відповідь криється не в математиці, а в фізиці напівпровідників.

Довгий час тренування глибоких мереж на звичайних центральних процесорах (CPU) займало тижні або навіть місяці. Це робило експерименти надто повільними. Ситуацію змінило нестандартне використання графічних процесорів (GPU), які спочатку створювалися для відеоігор.

7.1 Команда Шмідхубера та прорив Дена Чірешана *(2010)*

У 2010 році в лабораторії IDSIA (Швейцарія), якою керував Юрген Шмідхубер, його постдок Ден Чірешан *(Dan Cireșan)* здійснив технічний прорив. Він реалізував навчання глибоких нейронних мереж на відеокартах NVIDIA, використовуючи архітектуру CUDA.

Результат був приголомшливим: прискорення обчислень у **50 разів** порівняно зі звичайними процесорами. Те, на що раніше йшли місяці, тепер робилося за дні.

Це дозволило команді Шмідхубера тренувати не просто «іграшкові» моделі, а справжні глибокі монстри з мільйонами ваг. Вони довели, що старий добрий метод *Backpropagation* (Ліннаінмаа, 1970) чудово працює для дуже глибоких мереж, якщо у вас є достатньо швидке «залізо». Ніякої попередньої «магії» *(pre-training)*, яку пропонував Джеффрі Хінтон у 2006 році, було не потрібно.

7.2 DanNet: Перша надлюдська система *(2011)*

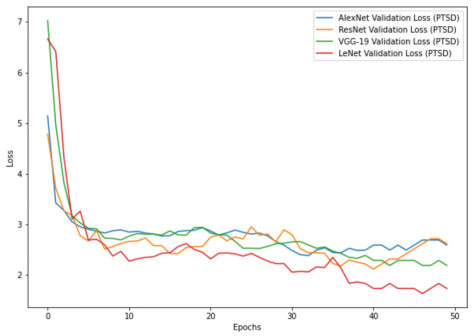
Існує поширене хибне уявлення, що ера глибокого навчання в комп'ютерному зорі почалася з перемоги мережі AlexNet у 2012 році. Однак історичні факти свідчать про інше.

За рік до AlexNet, у серпні 2011 року, команда Шмідхубера (Чірешан, Мейєр, Маші) виграла престижний конкурс з розпізнавання образів **IJCNN 2011 Traffic Sign Recognition Competition** [7].

Їхня система, яку часто називають *DanNet*, була глибокою згортковою мережею (CNN), що працювала на GPU.

- **Завдання:** Розпізнавання дорожніх знаків (важливо для автопілотів).
- **Результат:** Мережа показала коефіцієнт помилки **0.56%**.
- **Людина:** Найкращі люди-учасники показали помилку **1.16%**.

Це був історичний момент: вперше комп'ютерна програма на базі глибокого навчання офіційно перевершила людину у задачі візуального розпізнавання. Потім послідували перемоги на конкурсах ICDAR 2011 (китайські ієрогліфи) та ISBI 2012 (сегментація нейронних тканин).



7.3 ImageNet 2012 та початок глобального «хайпу»

Чому ж тоді всі говорять про 2012 рік?

У 2012 році Алекс Крижевський (студент Джеффрі Хінтона) представив мережу **AlexNet** на конкурсі ImageNet [8]. Цей конкурс відрізнявся масштабом: замість тисяч зображень там був мільйон картинок, розбитих на 1000 класів.

Крижевський використав ті ж самі принципи, що й Чірешан (глибока CNN + GPU + Backpropagation), але застосував їх до значно більшого набору даних. AlexNet розгромила конкурентів (які не використовували нейромережі) з величезним відривом:

- Помилка AlexNet: **15.3%**
- Помилка найближчого конкурента: **26.2%**

Саме цей відрив привернув увагу Кремнієвої долини. Google, Facebook та Microsoft миттєво зрозуміли, що технологія дозріла для комерції. Почалася «золота лихоманка» купівлі стартапів та найму вчених.

Шмідхубер підкреслює: AlexNet – це безумовно важлива віха, але вона була *популяризацією* підходу, який вже довів свою ефективність і перевагу над людиною завдяки роботам Чірешана у 2011 році.

8 Сучасність: GAN, Трансформери та висновки *(2014–Сьогодення)*

Останнє десятиліття ознаменувалося появою генеративного штучного інтелекту. Комп'ютери навчилися не просто класифікувати дані (розпізнавати котів), а й створювати нові (малювати неіснуючих котів, писати вірші, писати код). Однак, як і у випадку з попередніми епохами, коріння цих «магічних» технологій сягає глибше, ніж прийнято вважати.

8.1 Генеративно-змагальні мережі (GAN) та принцип змагальності *(1990 vs 2014)*

У 2014 році Ян Гудфеллоу представив архітектуру **GAN (Generative Adversarial Networks)** [9]. Ідея була блискучою: дві нейромережі змагаються одна з одною.

- **Генератор (Generator):** намагається створити підробку (наприклад, фото обличчя).
- **Дискримінатор (Discriminator):** намагається відрізнити підробку від реального фото.

Це нагадує гру фальшивомонетника та поліцейського. У процесі цього змагання (minimax game) генератор вчиться створювати ідеальні копії.

Однак Юрген Шмідхубер нагадує про свою роботу 1990 року під назвою **Predictability Minimization** (Мінімізація передбачуваності). У тій роботі також описувалася система з двох мереж, що змагаються: одна мережа генерувала дані, а інша намагалася передбачити їхні властивості. Генератор намагався максимізувати помилку провісника, а провісник — мінімізувати її.

Це математично еквівалентно принципу GAN:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{9}$$

Шмідхубер стверджує: Гудфеллоу популяризував цей принцип і дав йому красиву назву, але концепція «штучної цікавості» через змагання мереж була розроблена ще на початку 90-х.

8.2 Трансформери та LLM: Масштабування старих ідей

З 2017 року (після статті Google «Attention is All You Need» [10]) світ захопили Трансформери. Це архітектура, на якій працюють ChatGPT, Claude, Gemini та інші великі мовні моделі (LLM).

Чи є це фундаментально новим винаходом? З точки зору історії науки — ні. Сучасний успіх LLM базується на трьох компонентах:

1. **Highway Networks (ResNets):** У травні 2015 року команда Шмідхубера опублікувала принцип Highway Networks — перші мережі з сотнями шарів, які можна було тренувати завдяки прокиданню зв'язків повз шари (skip connections). Пізніше це стало відомо як ResNet. Без цього сучасні глибокі трансформери просто неможливо було б навчити.
2. **Механізм уваги (Attention):** Як згадувалося в Розділі 5, це варіація на тему «швидких ваг» (Fast Weights, 1991), де мережа динамічно змінює свою реакцію залежно від контексту.
3. **Обчислювальна потужність:** Це найголовніший фактор. Закон Мура дозволив нам взяти алгоритми 80-х і 90-х років і запустити їх на кластерах з тисяч GPU, «згодувавши» їм весь інтернет.

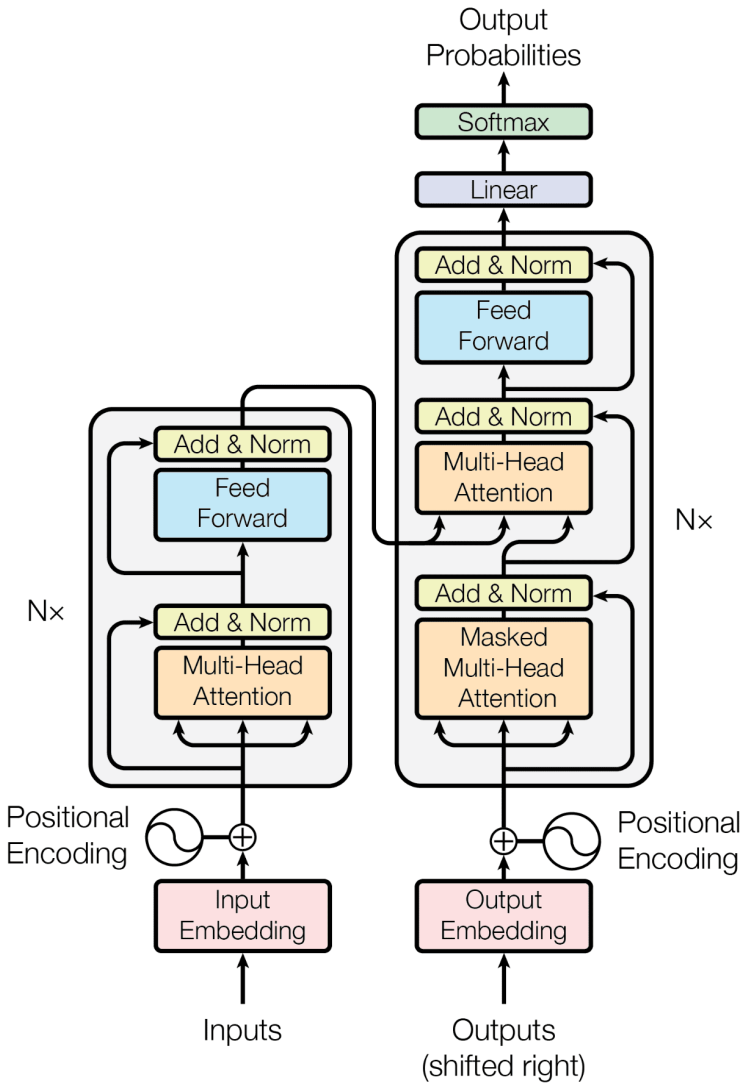


Рис. 2: Архітектура Transformer. Ключовим елементом є блок Multi-Head Attention.

Отже, інтелект сучасних LLM – це не результат якогось одного магічного рівняння, відкритого вчора, а результат інженерного масштабування математичних принципів, відкритих десятиліття тому.

9 Висновки

Оглядаючись на історію штучного інтелекту від 1676 року до сьогодення, можна зробити кілька важливих висновків для розуміння майбутнього:

- **Історія циклічна.** Те, що сьогодні подається як революція, часто є поверненням до ідей, які були технічно неможливі для реалізації в минулому (як глибокі мережі Івахненка у 60-х).
- **Ми стоїмо на плечах гігантів.** За кожним чат-ботом стоять тіні Лейбніца (ланцюгове правило), Гаусса (регресія), Івахненка (глибина), Ліннаінмаа (Backprop) та Фукусіми (згортки).
- **Залізо має значення.** Багато «зим штучного інтелекту» були спричинені не помилковістю теорій, а слабкістю комп'ютерів. Як тільки з'явилися GPU, старі теорії запрацювали.

Завершуючи словами Юргена Шмідхубера: «Все це – лише початок». Фундамент, закладений у XX столітті, настільки міцний, що ми лише починаємо будувати на ньому справжній Штучний Загальний Інтелект (AGI).

Література

[1] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[2] A. G. Ivakhnenko, "Polynomial theory of complex systems," *IEEE transactions on Systems, Man, and Cybernetics*, vol. 1, no. 4, pp. 364–378, 1971.

[3] S. Linnainmaa, "The representation of the cumulative rounding error of an algorithm as a taylor expansion of the local rounding errors," Master's Thesis, University of Helsinki, 1970.

[4] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[5] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, vol. 22, no. 1, 2011, p. 1237.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, vol. 25, 2012.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, vol. 27, 2014.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, vol. 30, 2017.