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

Системи штучного інтелекту

Обчислення в обчислювальних графах: зворотне поширення помилки

Семінар

Виконав:
Владислав Бардін
Група:
ІМ-42мп
Курс:
5

20 січня 2025 р.

1 Обчислення в обчислювальних графах: зворотне поширення помилки

Сьогодні нейронні мережі — це не просто наукова теорія, а реальний інструмент, який активно використовується в повсякденому житті. Вони допомагають розпізнавати обличчя на фото, рекомендувати фільми, ставити медичні діагнози та навіть передбачати ринкові зміни. Але щоб нейронна мережа працювала правильно, її потрібно навчити. Для цього важливо оптимізувати її параметри. Тут на допомогу приходить алгоритм зворотного поширення помилки, або backpropagation.

Цей алгоритм став справжнім проривом у машинному навчанні. Завдяки йому вдалося навчити складні багатошарові нейронні мережі, які раніше вважалися практично неможливими для використання. Backpropagation дозволяє мережі поступово зменшувати свої помилки і покращувати результати, використовуючи досить просту математику.

1.1 Історія алгоритму

Ідеї, які лежать в основі алгоритму backpropagation, з'явилися ще в 1974 році [6]. Пол Вербос у своїй роботі запропонував спосіб обчислення градієнтів у складних моделях. Проте справжню популярність алгоритм здобув завдяки статті Джефрі Гінтона, Девіда Румельхарта і Рональда Вільямса у 1986 році [7]. Вони змогли детально пояснити, як алгоритм можна застосовувати для навчання багатошарових мереж. Ця стаття стала переломним моментом у дослідженні нейронних мереж. Вчені довели, що такі мережі можуть ефективно працювати, якщо правильно оновлювати їх параметри на основі похибок.

1.2 Як працює алгоритм

Головна ідея backpropagation дуже проста. Мережа спочатку обчислює результат (це називається forward pass), а потім порівнює його з правильним значенням. Якщо є помилка, алгоритм визначає, як потрібно змінити кожен параметр, щоб зменшити цю помилку (це називається backward pass).

Forward pass допомагає мережі зробити прогноз. Наприклад, у задачі розпізнавання цифр на зображеннях, мережа спробує вгадати, яка цифра намальована. Потім backward pass обчислює, наскільки кожен параметр вплинув на помилку. Ця інформація використовується, щоб трохи змінити параметри і зробити мережу точнішою.

1.3 Чому алгоритм важливий

Backpropagation дозволяє нейронним мережам навчатися автоматично. Це зробило їх доступними для вирішення складних задач у багатьох галузях. Наприклад, алгоритм допомагає: - Розпізнавати обличчя, текст або об'єкти на фото. - Аналізувати емоції у текстах або перекладати їх іншими мовами. - Пропонувати фільми чи товари, які можуть вас зацікавити.

Без цього алгоритму сучасні системи штучного інтелекту не могли б бути такими ефективними. Він забезпечує їхню здатність швидко адаптуватися до нових даних і став основою для багатьох технологій.

2 Поняття обчислювальних графів

Алгоритм зворотного поширення помилки є основою навчання нейронних мереж. Його ефективність значною мірою базується на використанні обчислювальних графів. Ці графи забезпечують зручний та структурований підхід до представлення обчислень, що робить алгоритм потужним і універсальним.

2.1 Обчислювальний граф?

Обчислювальний граф (англ. computational graph) — це математична модель, яка представляє обчислення у вигляді орієнтованого графа. У цьому графі:

- Вузли відповідають операціям або функціям (наприклад, додавання, множення, застосування активаційної функції).
- Ребра показують передачу результатів між операціями.

Ця структура дозволяє розбивати складні обчислення на простіші кроки, які можна виконувати послідовно або паралельно. Завдяки цьому обчислювальні графи є основою для автоматичного обчислення градієнтів за допомогою правила ланцюгового диференціювання.

2.2 Основні переваги обчислювальних графів

Ключовою особливістю обчислювального графа є можливість поділу обчислень на послідовні кроки. Це надає низку переваг:

- зручна візуалізація всього ланцюга обчислень;
- автоматизація процесу диференціювання;
- підтримка паралельного виконання незалежних кроків, що підвищує ефективність.

Завдяки цьому можна легко проаналізувати, яким чином початкові вхідні змінні впливають на кінцевий результат, а також кроки, що не залежать один від одного, можуть виконуватися паралельно, що робить обчислення швидшими та ефективнішими.

2.3 Приклад обчислювального графа

Як приклад розглянемо просту функцію з двома змінними x та y:

$$f(x, y) = (x + y)y \tag{1}$$

Обчислення цього графу можна подати у вигляді вузла А, який отримує 2 вхідні параметри x та y, та виконує операцію додавання x+y. Та вузла Б, який отримує як вхідні параметри y та результат виконання вузла А (x+y) виконуючи множення ((x+y)×y).

Сформувавши таку схему нею можна зробити 2 проходи, у прямому (forward pass) та зворотному напрямку (backward pass). Прямий обхід застосовується для обчислення кінцевого значення функції, а зворотній для обчислення градієнтів відносно змінних x та y.

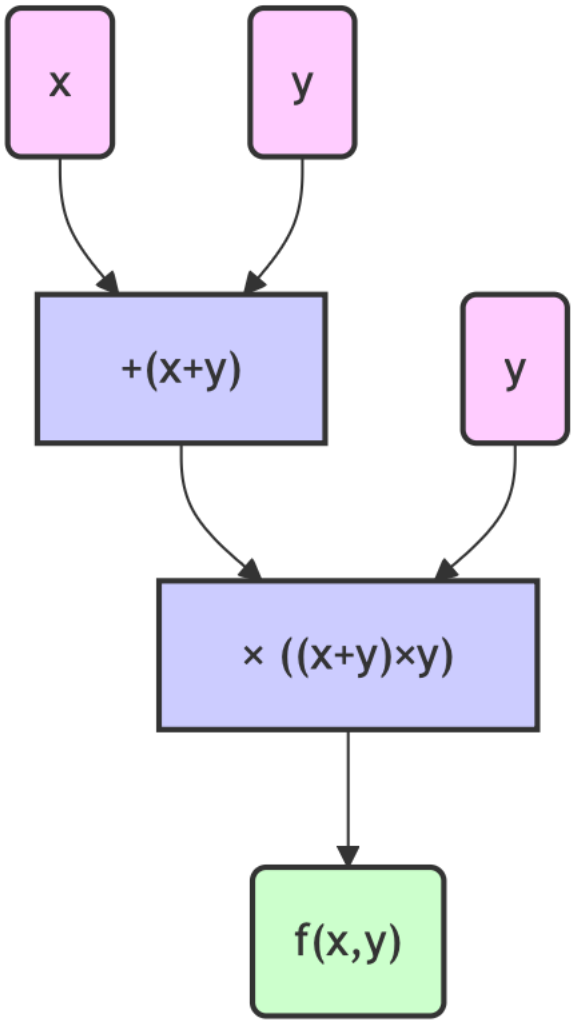

Рис. 1: Обчислювальний граф для функції

2.4 Значення для нейронних мереж

Завдяки такому підходу у складних нейронних мережах з багатьма шарами і безліччю параметрів ми можемо системно структурувати процес. Замість монолітного обчислення маємо послідовну (або й паралельну) сукупність вузлів і ребер, де кожна операція виконується у чітко визначеному порядку.

Таким чином, обчислювальні графи є фундаментом для реалізації алгоритму зворотного поширення помилки, оскільки саме завдяки чіткій структурі вузлів і ребер можна коректно та швидко знайти градієнти кожного параметра, впливаючи на точність і швидкість навчання штучних нейронних мереж.

3 Загальна ідея алгоритму зворотного поширення помилки

Зворотне поширення помилки — це ефективний метод оптимізації, що використовується для корекції вагових параметрів нейронної мережі з метою мінімізації функції втрат. Алгоритм ґрунтується на правилі ланцюгового диференціювання, що дозволяє обчислювати похідні (градієнти) функції втрат по кожному параметру мережі.

Для виконання прямого проходу вхідні дані подаються на вхід мережі, проходять через усі шари з відповідними функціями активації, і на виході отримується прогноз. Після цього обчислюється значення функції помилки, наприклад, сума квадратів різниць між передбаченими та реальними виходами, або крос-ентропія. Обчислене значення функції помилки використовується для руху у зворотному напрямку, послідовно обчислюючи частинні похідні відносно кожної ваги. Таким чином ми отримуємо інформацію про те, як змінити кожен параметр мережі, щоб зменшити загальну помилку.

У загальному випадку, якщо вихідна помилка позначається як E, а проміжна змінна як z, то похідну dE/dz можна обчислити через інші змінні u,v... згідно з правилом ланцюга:

$$\frac{dE}{dz} = \frac{dE}{du} \cdot \frac{du}{dv} \cdot \dots \cdot \frac{dw}{dz} \tag{2}$$

У контексті нейронної мережі кожен "ланцюг" відповідає шляху в обчислювальному графі від параметра до підсумкової функції помилки. Після обчислення градієнтів алгоритм зворотного поширення помилки зазвичай застосовується у поєднанні з методом градієнтного спуску, який оновлює кожну вагу W_{ij} у напрямку протилежному до градієнта:

$$W_{ij} \leftarrow W_{ij} - \eta \cdot \frac{dE}{dW_{ij}} \tag{3}$$

де η – швидкість навчання.

Це дозволяє послідовно зменшувати значення функції помилки на кожній ітерації, поки мережа не досягне прийнятної точності.

Загалом, алгоритм зворотного поширення помилки — це "серце" навчання більшості глибинних нейронних мереж. Він дає змогу поєднати переваги обчислювальних графів з математикою ланцюгового диференціювання, що в підсумку забезпечує ефективне й точне оновлення параметрів моделі.

Для прикладу розглянемо просту нейронну мережу з одним прихованим шаром. Прямий прохід на кожному шарі обчислюється як:

$$z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)} \tag{4}$$

$$a^{(l)} = \sigma(z^{(l)}) \tag{5}$$

де:

- $z^{(l)}$ – вхідні значення нейронів на шарі l,
- $W^{(l)} i b^{(l)}$ – ваги та зсуви,
- σ – функція активації,
- $a^{(l-1)}$ – вихід нейронів на шарі l.

обислення функції втрат відбувається за формулою:

$$\mathcal{L} = \frac{1}{2} \sum_i (y_i - \hat{y}_i)^2 \tag{6}$$

де:

- y_i – очікуваний результат,
- $\hat{y}_i$ – передбачення моделі.

Зворотній прохід обчислюється за формулами:

Градієнти обчислюються, починаючи з вихідного шару, за допомогою правила ланцюгового диференціювання:

$$\delta^{(L)} = \frac{\partial \mathcal{L}}{\partial z^{(L)}} = (\hat{y} - y) \cdot \sigma'(z^{(L)}) \tag{7}$$

для попередніх шарів використовується:

$$\delta^{(l)} = \left(W^{(l+1)}\right)^T \delta^{(l+1)} \cdot \sigma'(z^{(l)}) \tag{8}$$

Оновлення ваг здійснюється за правилом граждієнтного спуску:

$$W^{(l)} := W^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial W^{(l)}} \tag{9}$$

$$b^{(l)} := b^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial b^{(l)}} \tag{10}$$

де η – швидкість навчання.

Наприклад, розглянемо функцію з двома змінними $f(x, y) = (x + y)^2$.

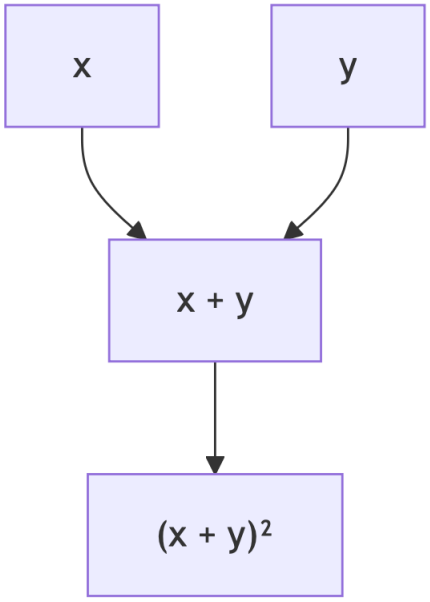

Рис. 2: Обчислювальний граф для функції

Прямий прохід: Вхідні дані $x = 2, y = 3$, тоді $f(x, y) = (x + y)^2 = (2 + 3)^2 = 25$.

Зворотній прохід:

Для похідних використовуємо

$$\frac{\partial f}{\partial x} = 2(x + y), \quad \frac{\partial f}{\partial y} = 2(x + y) \tag{11}$$

Підставляючи x та y отримаємо

$$\frac{\partial f}{\partial x} = 2(2 + 3) = 10, \quad \frac{\partial f}{\partial y} = 2(2 + 3) = 10 \tag{12}$$

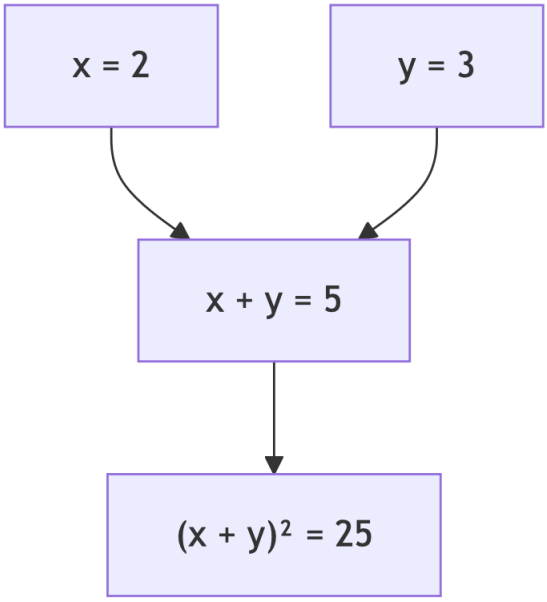

Рис. 3: Схема прямого проходу з обчисленням проміжних значень

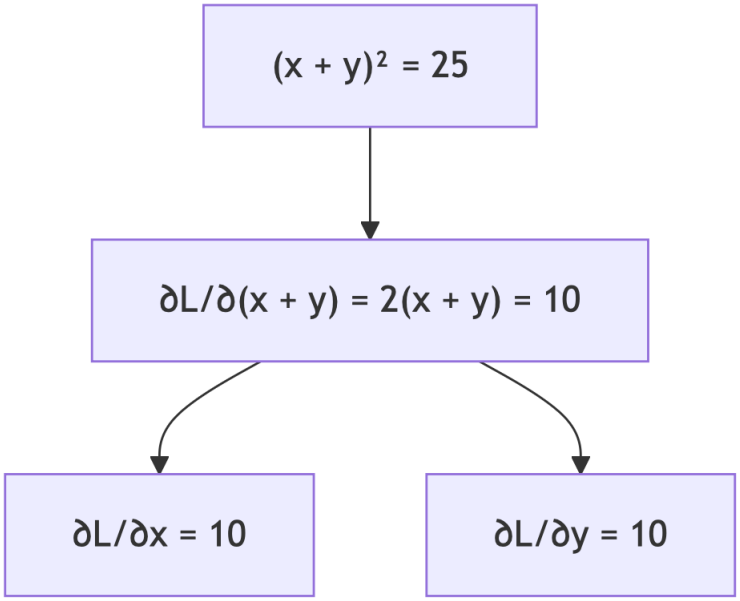

Рис. 4: Схема обчислення градієнтів при зворотньому проході

4 Практичне застосування backpropagation

Алгоритм зворотного поширення помилки (backpropagation) є фундаментальним елементом процесу навчання нейронних мереж. Завдяки своїй ефективності він став основою для багатьох практичних застосувань у різних сферах. Розглянемо детальніше, як саме алгоритм використовується і чому його застосування є доцільним.

4.1 Розпізнавання образів

Розпізнавання образів є однією з ключових задач машинного навчання. Алгоритм backpropagation використовується для навчання моделей класифікації зображень, таких як багатошарові персептрони (MLP) або згорткові нейронні мережі (CNN).

- Під час forward pass зображення, представлене у вигляді вектору пікселів, проходить через шари нейронної мережі, де кожен шар виконує лінійні перетворення і функції активації.
- На виході мережі отримується набір ймовірностей, що відповідають кожному класу (наприклад, цифрам 0-9 для набору даних MNIST).
- У backward pass алгоритм backpropagation обчислює градієнти функції втрат (наприклад, кросентропії) за кожною вагою мережі, використовуючи правило ланцюгового диференціювання.
- Ваги оновлюються за допомогою градієнтного спуску, щоб зменшити помилку між передбаченим і реальним класом.

Backpropagation дозволяє систематично знижувати помилку, адаптуючи ваги мережі до складних закономірностей у зображеннях. У результаті модель досягає точності понад 98% на MNIST, що робить її практично застосовною для автоматизації завдань розпізнавання.

4.2 Обробка природної мови

У задачах обробки природної мови (NLP), таких як класифікація тексту, машинний переклад або аналіз тональності, backpropagation використовується для оптимізації моделей, зокрема рекурентних нейронних мереж (RNN) та трансформерів.

- Вхідний текст перетворюється у числові векторні уявлення (word embeddings), які подаються на вхід нейронної мережі.
- Під час forward pass мережа аналізує послідовності слів, враховуючи їхній контекст.
- У backward pass алгоритм backpropagation через час (backpropagation through time, BPTT) розраховує похідні для кожного стану RNN, дозволяючи оновлювати ваги, що відповідають за довгострокові залежності у тексті.

Алгоритм дозволяє враховувати контекст попередніх слів, що є критично важливим для аналізу послідовностей. Наприклад, у задачі класифікації відгуків модель може визначити емоційне забарвлення фраз, таких як "дуже хороший"або "абсолютно жахливий". Це забезпечує точність понад 90%.

4.3 Рекомендаційні системи

Рекомендаційні системи використовують backpropagation для персоналізації рекомендацій на основі поведінки користувачів.

- Кожного користувача і кожен товар система представляє у вигляді векторів (embeddings), які знаходяться у спільному просторовому представленні.
- Forward pass моделює ймовірність того, що користувач обере конкретний товар, на основі подібності між векторами користувача і товару.
- Backward pass використовується для мінімізації функції втрат, наприклад, кореня середньоквадратичної похибки (RMSE) між реальними і передбаченими рейтингами.

Backpropagation дозволяє постійно покращувати точність рекомендацій, адаптуючи систему до поведінки користувачів. Це сприяє збільшенню конверсії та покращенню взаємодії з платформою.

4.4 Медицина та інші сфери

Застосування алгоритму виходить далеко за межі вищезгаданих прикладів. Наприклад:

- У медицині алгоритм використовується для аналізу зображень, таких як МРТ чи рентген, для виявлення патологій.
- У фінансах backpropagation допомагає прогнозувати ринкові тенденції, аналізуючи великі обсяги історичних даних.
- В ігровій індустрії алгоритм застосовується для навчання ігрових агентів, які здатні самостійно освоювати складні середовища.

4.5 Підсумки

Завдяки алгоритму backpropagation нейронні мережі здатні вирішувати задачі, які раніше здавалися неможливими. Його універсальність, ефективність і адаптивність роблять його незамінним інструментом у сучасному світі штучного інтелекту. Алгоритм забезпечує оптимізацію параметрів моделей, дозволяючи їм швидко і точно навчатися, що відкриває широкі можливості для інновацій у різних галузях.

5 ВИСНОВКИ

Зворотне поширення помилки (backpropagation) є фундаментальним алгоритмом, який забезпечив прорив у сфері машинного навчання та глибинних нейронних мереж. Завдяки послідовному обчисленню градієнтів кожного параметра мережі та врахуванню усіх проміжних операцій в обчислювальному графі стало можливим навчати багатошарові моделі та отримувати вражаючі результати в різноманітних завданнях: від комп'ютерного зору до обробки природної мови і рекомендаційних систем. У ході роботи над рефератом проаналізовано:

- суть обчислювальних графів: їхня роль у структуризації всього процесу обчислень та наочне представлення операцій над даними;
- механізм та ідея backpropagation: від історичних витоків до конкретного алгоритмічного втілення; описано, як кожен параметр впливає на помилку на виході і як його оновити за правилом ланцюгового диференціювання.

Застосування зворотного поширення помилки разом із потужними засобами апаратного прискорення дає змогу обробляти величезні обсяги даних, будувати глибинні мережі й успішно розв'язувати складні прикладні завдання. Попри наявність альтернативних підходів, backpropagation залишається основою більшості сучасних рішень у галузі штучного інтелекту.

Backpropagation має перспективи подальших досліджень, що полягають у розробці покращених методів оптимізації, вивченні способів розв'язання проблеми зникаючих та вибухаючих градієнтів, а також у пошуку нових архітектур, здатних ще ефективніше використовувати потенціал зворотного поширення помилки. Це означає, що backpropagation та обчислювальні графи залишатимуться важливою складовою інструментарію машинного навчання у найближчому майбутньому.

Література

[1] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016.

[2] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *Nature*, 1986, Vol. 323, pp. 533-536.

[3] Werbos, P. J. Applications of advances in nonlinear sensitivity analysis. *System Modeling and Optimization*, Springer, Berlin, Heidelberg, 1982, pp. 762-770.

[4] Nielsen, M. *Neural Networks and Deep Learning*. 2015. Retrieved from http://neuralnetworksanddeeplearning.com.

[5] Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer, 2006.

[6] Paul John Werbos. *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting*. Adaptive and Cognitive Dynamic Systems, 1974.

[7] Geoffrey E. Hinton *Learning Distributed Representations of Concepts*, 1986. Retrieved from https://www.cs.toronto.edu/~hinton/absps/families.pdf.