

УДК 519.688

DOI 10.31651/2076-5886-2019-1-61-75

PACS 02.60.-x, 02.60.Pn

КУЧЕР Олександр Іванович,

аспірант кафедри прикладної математики та інформатики Черкаського національного університету імені Богдана Хмельницького
e-mail: olexandr.kucher@gmail.com
ORCID 0000-0002-6902-5068

ОСОБЛИВОСТІ ЗАСТОСУВАННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ЗАДАЧ ОБРОБКИ ЗОБРАЖЕНЬ

У статті дано базове поняття та характеристики згорткових нейронних мереж, описана структура та математичні підходи до реалізації нейронної мережі цього типу. Наданий опис основних шарів згорткових нейронних мереж, наведені параметри, завдяки яким можна змінювати процес навчання мережі. Детально розглянутий процес навчання кожного з шарів нейронної мережі. Додатково описані техніки, що дають можливість збільшити набір даних та покращити процес навчання мережі за допомогою кількох перетворень

Ключові слова: глибоке навчання, машинне бачення, нейронні мережі, розпізнавання облич, згорткові нейронні мережі, рецептивні поля, глибоке навчання, нейронні мережі, згорткові нейронні мережі, класифікація, локалізація, виявлення, сегментація, випрямлені лінійні одиниці.

Постановка проблеми

Згорткові нейронні мережі (ЗНМ, Convolutional Neural Networks, CNN) – одні з найвпливовіших інновацій в області комп'ютерного зору. Вперше нейронні мережі привернули до себе увагу в 2012 році на конкурсі ImageNet. За допомогою ЗНМ було встановлено новий рекорд помилок при класифікації – 15% (попереднє значення було 26%) [1]. На сьогоднішній день глибоке навчання (deep learning) використовують безліч відомих компаній:

- Facebook – використовує для алгоритмів автоматичного виставлення тегів
- Google – для пошуків по фотографіях
- Amazon – для генерації рекомендації товарів

Але класичний і найбільш популярний спосіб застосування нейронних мереж це обробка зображень. Розглянемо як ЗНМ використовуються для класифікації зображень.

Виклад основного матеріалу

Задача класифікації зображень – це обробка зображення і визначення класу до якого воно належить (автомобіль, тварина і т. д.) або групи класів, які найкраще його характеризують.

Вхідні і вихідні дані

Коли комп'ютер отримує на вхід оцифроване зображення, він працює з масивом пікселів. Залежно від розширення зображення розміри масиву можуть бути, наприклад, 64x64x3 (де 3 – це значення кожного з каналів RGB). Для кращого розуміння уявимо, що в нас є кольорове зображення в форматі JPG розміром 360x360 пікселів. Воно буде представлено масивом з розмірами 360x360x3. Кожне із значень масиву набуває значення від 0 до 255, що описує інтенсивність пікселя. Для людини ці значення не мають сенсу, але це єдиний спосіб представити зображення в комп'ютері. Ідея полягає в тому, щоб надати програмі (алгоритму) цю матрицю і отримати в результаті значення

що описують ймовірність відношення цього зображення до певного класу (0.8 – автомобіль, 0.15 – тварина, 0.05 - людина).



```
08 02 22 97 38 15 00 40 00 75 04 05 07 78 52 12 50 77 91 08
49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 04 56 62 00
81 49 31 73 55 79 14 29 93 71 40 47 53 88 30 03 49 13 36 65
52 70 95 23 04 60 11 42 69 24 68 56 01 32 56 71 37 02 36 91
22 31 16 71 51 67 63 89 41 92 36 54 22 40 40 28 66 33 13 80
24 47 32 60 99 03 45 02 44 75 33 53 78 36 84 20 35 17 12 50
32 98 81 28 64 23 67 10 26 38 40 67 59 54 70 66 18 38 64 70
67 26 20 68 02 62 12 20 95 63 94 39 63 08 40 91 66 49 94 21
24 55 58 05 66 73 99 26 97 17 78 78 96 83 14 88 34 89 63 72
21 36 23 09 75 00 76 44 20 45 35 14 00 61 33 97 34 31 33 95
78 17 53 28 22 75 31 47 15 94 03 80 04 42 16 14 09 53 54 92
16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57
86 56 00 48 35 71 89 07 05 44 44 37 44 60 21 58 51 54 17 58
19 80 81 68 05 94 47 69 28 73 92 13 86 52 17 77 04 89 55 40
04 52 08 83 97 35 99 16 07 97 57 32 16 26 26 79 33 27 98 66
88 36 68 87 57 62 20 72 03 46 33 67 46 55 12 32 63 93 53 69
04 42 16 73 38 25 39 11 24 94 72 18 08 46 29 32 40 62 76 36
20 69 36 41 72 30 23 88 34 62 99 69 82 67 59 85 74 04 36 16
20 73 35 29 78 31 90 01 74 31 49 71 48 86 81 16 23 57 05 54
01 70 54 71 83 51 54 69 16 92 33 48 41 43 52 01 89 19 47 48
```

Рис. 1. Що бачить людина та що «бачить» комп'ютер

Зв'язок з біологією

ЗНМ – це прототип зорової кори головного мозку. Зорова кора має невеликі ділянки клітин, що є чутливими до певних областей поля зору. Цю ідею детально розглянули за допомогою експерименту Г'юбел і Візел в 1962 році в якому показали, що окремі нервові клітини реагували лише при візуальному сприйнятті меж певної орієнтації. Наприклад, деякі нейрони активувалися, коли сприймали вертикальні границі, а деякі – горизонтальні або діагональні. Г'юбел і Візел з'ясували, що всі нейрони розташовані в виді стержневої архітектури і разом формують візуальне сприйняття. Цю ідею спеціалізованих компонентів, що розв'язують якусь задачу і використовують машини, і ця ідея – основа ЗНМ. [7]

Структура

Зображення пропускається через серію згорткових, нелінійних шарів, шарів об'єднання і повнозв'язних шарів в результаті чого генеруються вихідні дані. Вихідними даними може бути клас або ймовірність класів, які найкраще описують зображення.

Перший шар – математична частина

Перший шар в ЗНМ завжди згортковий. Найлегше зрозуміти, що таке згортковий шар, допоможе представлення його у вигляді ліхтарика, який світить на верхню ліву частину зображення. Припустимо, що світло з ліхтарика покриває площу 5x5 пікселів. В термінах комп'ютерного навчання цей ліхтарик – це фільтр, а область на яку він світить називається рецептивним полем. Глибина фільтра повинна бути такою ж як глибина вхідного зображення (5x5x3). Оскільки фільтр здійснює згортку, тобто рухається по зображенню, він перемножує значення фільтра на вихідні значення пікселів зображення (поелементно). В результаті отримуємо одне число. Тепер необхідно повторити цей процес в кожній позиції (Наступний крок – це переміщення фільтра вправо на 1 піксель). Кожна унікальна позиція зображення генерує одне число. Після проходження фільтра по всіх позиціях (для зображення 32x32x3) отримуємо матрицю 28x28x1, яку називають функцією активації або картою ознак. Матриця 28x28 отримується тому, що є 784 різні позиції, які можуть пройти через фільтр 5x5. Ці 784 числа перетворюються в матрицю 28x28. [6]

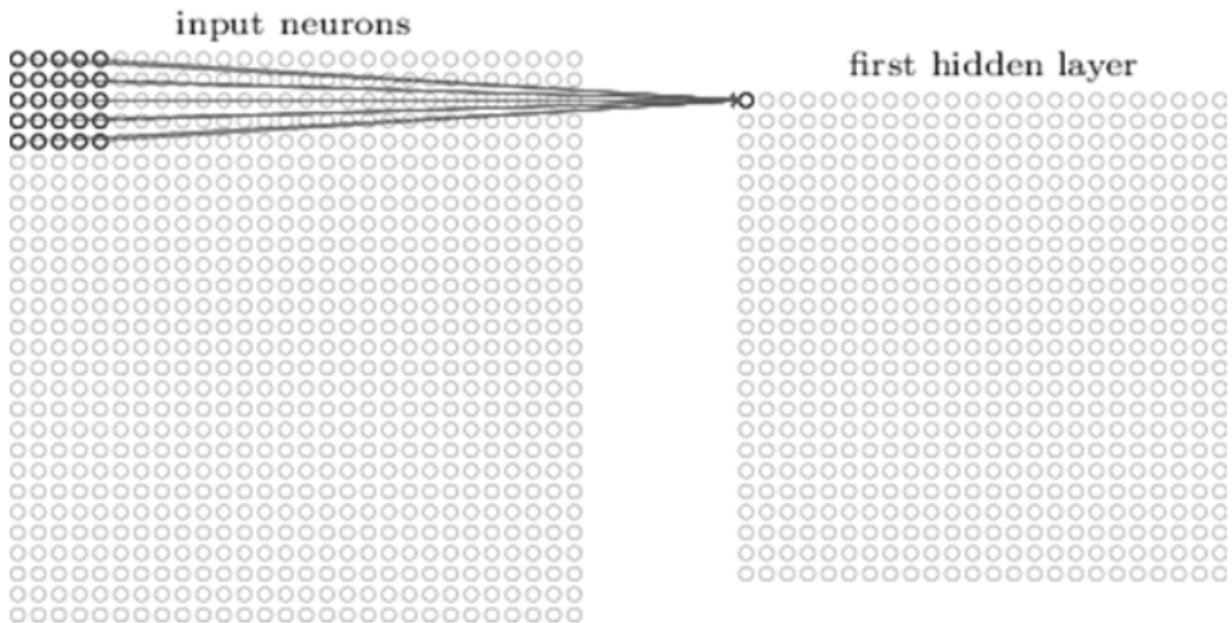


Рис. 2. Візуалізація фільтра 5x5, що створює карту активації з вхідних даних

Якщо ми використовуватимемо два 5x5x3 фільтри замість одного, то отримаємо матрицю 28x28x2.

Перший шар

Кожен фільтр можна розглядати як ідентифікатор властивості. Під властивостями маються на увазі прямі межі, прості кольори і криві. Припустимо, що наш перший фільтр 7x7x3 і він буде детектором кривих. (Для простоти розглянемо лише верхні шари фільтра і зображення). Фільтр має піксельну структуру, у якій числові значення більші вздовж області, що визначає форму кривої.

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

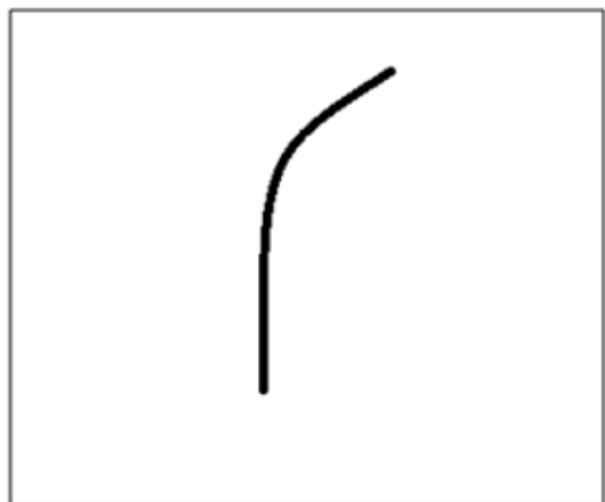


Рис. 3. Цифрове та візуальне представлення фільтра що знаходить криві

Повернемося до математичної візуалізації. Коли в верхньому лівому куті вхідного зображення є фільтр, він здійснює множення значень фільтра на значення пікселів області. Розглянемо приклад зображення, яке потрібно класифікувати, і встановимо фільтр в верхньому лівому куті.

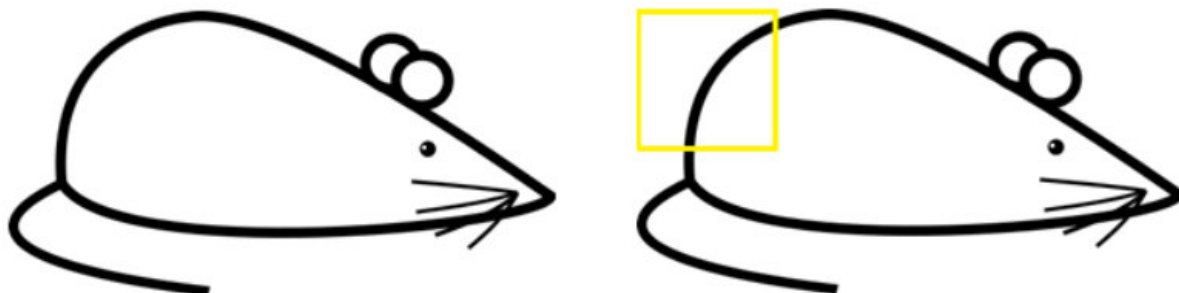


Рис. 4. Візуалізація фільтра на зображенні

Все що необхідно зробити – це перемножити значення фільтра на значення пікселів зображення.

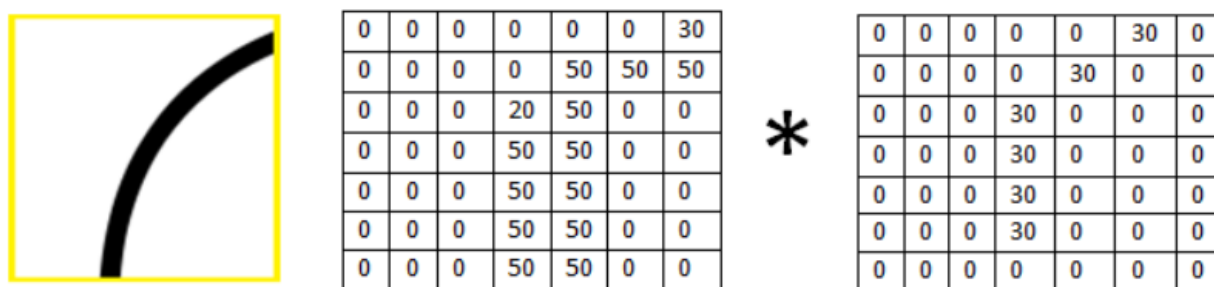


Рис. 5. Застосування фільтра в верхньому лівому куті зображення

Multiplication and Summation =

$$(50 \times 30) + (50 \times 30) + (50 \times 30) + (20 \times 30) + (50 \times 30) = 6600.$$

Якщо на зображенні є форма, що схожа на криву, яку представляє фільтр, то всі перемножені і додані значення утворюють значення на кілька порядків більше за 0 (порогове значення встановлюється програмістом).

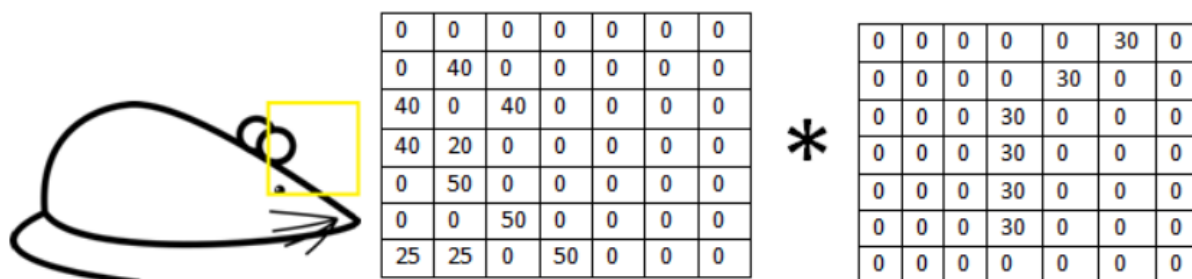


Рис. 6. Застосування фільтра в верхньому правому куті зображення

Multiplication and Summation = 0

Коли ми перемістимо фільтр, то отримаємо значення близьке до нуля. Причиною є те, що в цій області зображення немає нічого, що фільтр кривої міг виявити. Пам'ятайте – результат цього згорткового шару – це карта властивостей. В найпростішому випадку, при наявності одного фільтра згортки, карта властивостей виявить області, в яких з більшою ймовірністю є криві (якщо фільтр – детектор кривих). В цьому прикладі значення карти властивостей у верхньому лівому куті буде 6600. Це значення показує, що, можливо, щось схоже на криву є на зображенні, і така ймовірність активувала фільтр. В правому верхньому куті значення карти властивостей

буде 0, тому що на зображенні не було нічого, що могло б активувати фільтр. Ці значення є характеристиками лише одного фільтра, фільтра що знаходить криві зі згином назовні. Можуть бути й інші фільтри. Чим більше фільтрів, тим більша глибина карти властивостей і більше інформації про зображення можна отримати. [8]

Глибше через мережу

Зараз в традиційній згортковій мережі існують й інші шари, що перемножуються з згортковими. Класична архітектура ЗНМ матиме наступний вигляд:

Input -> Conv -> ReLU -> Conv -> ReLU -> Pool -> ReLU -> Conv -> ReLU -> Pool -> Fully Connected

де ***Conv*** – згортковий шар, ***ReLU*** – функція активації, ***Pool*** – об'єднуючий шар, ***Fully Connected*** – повнозв'язний шар.

Ми розглянули фільтри першого шару. Вони виявляють властивості базового рівня, такі як межі та криві. Для того щоб розпізнавати типи об'єктів на зображеннях, нам потрібна мережа, що здатна розпізнавати об'єкти більш високого рівня, наприклад руки, лапи або вуха. Результат обробки зображення першим фільтром має розміри 28x28x3 (за умови що використовувався фільтр 5x5x3). Коли зображення проходить через один згортковий шар, результат першого шару стає вхідними даними другого. Коли ми описували перший шар, вхідними даними було лише початкове зображення. Але коли ми перейшли до другого шару, вхідними значеннями для нього стала одна або декілька карт властивостей – результат обробки попереднього шару. Кожен набір даних описує позиції, де на початковому зображенні знаходяться певні базові ознаки. [1]

Тепер, коли застосовується набір фільтрів поверх цього (зображення пропускається через другий згортковий шар), в результаті будуть активовані фільтри, які представляють властивості більш високого рівня. Типами цих властивостей можуть бути півкілця або квадрати. Чим більше згорткових шарів застосовується до зображення, тим більш складні характеристики проявляються в картах активації. В кінці мережі можуть бути фільтри, що активуються при наявності рукописного тексту на зображенні, при наявності зелених об'єктів і т. д.

При русі в глибину мережі, фільтри працюють з усе більшим полем сприйняття, а значить, вони в змозі опрацьовувати інформацію з більшої площі початкового зображення (вони краще адаптуються до обробки більшої області піксельного простору).

Повнозв'язні шари

Тепер коли можливо виявити високорівневі властивості, можна прикріпити повнозв'язний шар в кінці мережі. Цей шар бере вхідні дані і виводить N-мірний вектор, де N – число класів, серед яких програма обирає потрібний. Наприклад, якщо ви працюєте над програмою розпізнавання цифр, N буде рівне 10. Кожне значення в цьому векторі представлятиме собою ймовірність конкретного класу. Наприклад, якщо результуючий вектор для програми розпізнавання цифр це: ***[0 0.1 0.1 0.75 0 0 0 0 0 0.05]***, значить існує ймовірність 10% що на зображенні «1», ймовірність 10%, що на зображенні «2», ймовірність 75%, що на зображенні «3», і ймовірність 5%, що на зображенні «9».

Спосіб, за допомогою якого працює повнозв'язний шар – це взаємодія з вихідними даними попереднього шару (який повинен виводити високорівневі карти властивостей) і визначення властивостей, які більш зв'язані з певним класом. Повнозв'язний шар виявляє, що функції високого рівня сильно зв'язані з певним класом і мають певні коефіцієнти ваги, тобто, коли відбувається обрахунок ваги з попереднім шаром, отримуються правильні ймовірності для різних класів.

Навчання (або що змушує це працювати)

Спосіб за допомогою якого комп'ютер здатний корегувати значення фільтра (або коефіцієнтів ваги) – це процес навчання, який називається методом зворотного розповсюдження помилки.

Що ж нейронній мережі необхідно для роботи. До моменту побудови мережі, ваги або значення фільтра випадкові. Фільтри не вміють шукати межі і криві. Фільтри верхніх шарів не вміють шукати руки, лапи і вуха. Для навчання мережі застосовується механізм надання їй зображення і ярлика чи класу, що йому відповідає.

Метод зворотного розповсюдження помилки можна розділити на 4 окремих блоки: пряме розповсюдження, функція втрати, зворотне розповсюдження і оновлення ваги. Під час прямого розповсюдження, береться тренувальне зображення (це матриця $32 \times 32 \times 3$) і пропускається через всю мережу. В першому прикладі, оскільки всі ваги або значення фільтрів були ініційовані випадковим чином, вихідним значенням буде щось на зразок $[0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1]$, тобто таке значення, яке не надає перевагу жодному з наявних класів. Мережа з такими властивостями не здатна знайти властивості базового рівня і не здатна аргументовано визначити клас зображення. Це веде до функції втрати. На цьому етапі використовуються дані для навчання. У таких даних є зображення і ярлик. Якщо перше зображення для навчання це цифра 3, то ярликом буде $[0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]$. Функція втрати може бути виражена по різному, але часто використовується середньоквадратична помилка.

$$E_{total} = \sum \frac{1}{2} (target - output)^2 . \quad (1)$$

Нехай це значення буде L . Втрата буде дуже великою для перших двох зображень. Для того щоб прогнозований ярлик був таким же, як ярлик зображення для навчання, потрібно звести до мінімуму кількість втрат, які в нас є. Розглядаючи цю задачу як задачу оптимізації з математичного аналізу, необхідно з'ясувати, які вихідні дані сприяли втратам (помилкам) мережі.

Один зі способів візуалізувати ідею мінімізації втрат – це 3-D графік, де ваги нейронної мережі (їх більше 2-х, але тут приклад спрощений) це незалежні змінні, а залежна змінна – це втрата. Задача мінімізації втрат – відрегулювати ваги таким чином, щоб знизити втрату. Візуально необхідно наблизитися до найнижчої точки чашоподібного об'єкту. Щоб зробити це, необхідно знайти похідну втрати (в рамках зображеного графіка – розрахувати кутовий коефіцієнт в кожному з напрямків) з врахуванням ваг.

Це математичний еквівалент dL/dW , де W – ваги одного шару. Далі необхідно виконати **зворотне розповсюдження** через мережу, яке визначає, які ваги здійснили більший вплив на втрати, і знайти способи, як їх налаштувати, щоб зменшити втрати. Після знаходження похідної, переходимо до останнього етапу – оновлення ваг. Необхідно взяти всі ваги фільтрів і оновити їх так, щоб вони змінювалися в напрямку градієнту.

$$w = w_i - n \frac{dL}{dW} , \quad (2)$$

де w - вага, w_i - початкова вага, n - швидкість навчання.

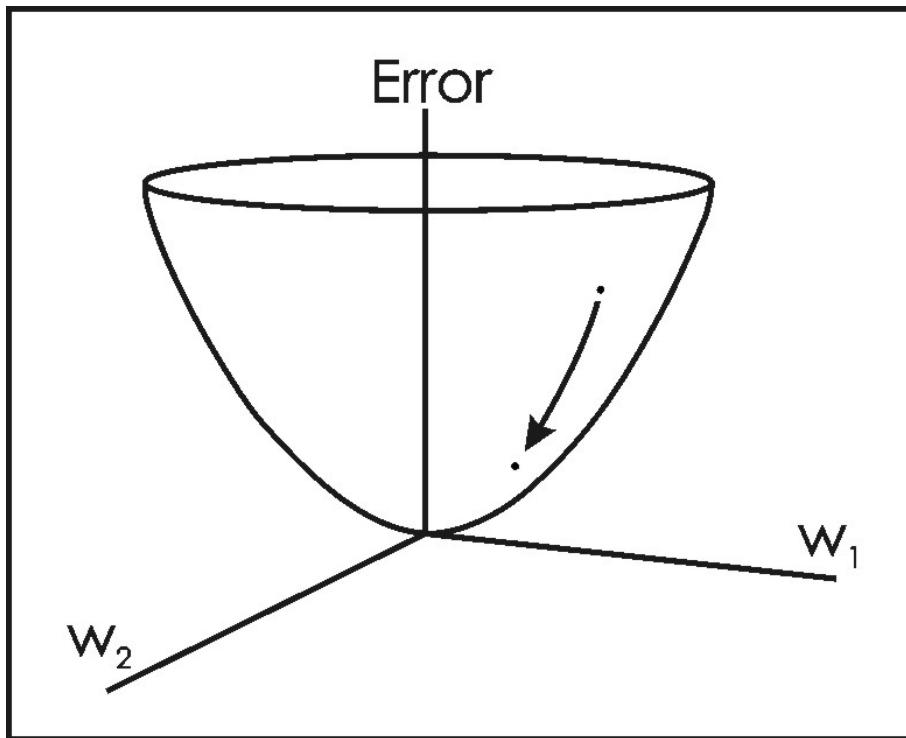


Рис. 7. Візуалізація задачі мінімізації

Швидкість навчання – це параметр, який обирається програмістом. Висока швидкість навчання значить, що в нових коефіцієнтах ваги були зроблені більші кроки, завдяки чому може знадобитися менше часу, щоб отримати оптимальний набір ваг. Але занадто велика швидкість навчання може привести до дуже великих і не достатньо точних стрибків, які завадять досягненню оптимальних показників.

Процес прямого розповсюдження, функцію втрати, зворотне розповсюдження і оновлення ваг, зазвичай називають одним періодом дискретизації (або epoch - епохою). Програма буде повторювати цей процес фіксовану кількість періодів для кожного тренувального зображення. Після того, як оновлення параметрів завершиться на останньому тренувальному зразку, мережа повинна бути достатньо добре навчена і ваги шарів повинні бути налаштовані правильно.

Тестування

Нарешті, щоб побачити, чи працює ЗНМ, ми беремо інший набір зображень і ярликів і пропускаємо зображення через мережу. Порівнюємо результати з реальними даними і перевіряємо чи працює наша мережа.

Крок і відступ

При роботі з згортковими нейронними мережами (ЗНМ) ми можемо змінювати 2 параметри, для того, щоб змінити поведінку кожного з шарів. Після обранні розміру фільтра, ми можемо також обрати **крок і відступ**.

Крок – це параметр, що контролює процес згортки вхідного зображення за допомогою фільтра. На Рис. 8 зображено приклад згортки вхідних даних зі зміщенням на одну комірку за раз. Кількість комірок, на які зміщується фільтр на кожному кроці, називається кроком. На Рис. 8 розмір кроку – 1. За крок завжди беруть ціле число.