Гібридний еволюційний алгоритм формування топології глибокої нейромережі

Д.Ю. Коваль, О.І. Чумаченко
Факультет інформатики та обчислювальної техніки
НТУУ «КПІ»
Київ, Україна

Hybrid evolutional algorithm for formation of deep neural network topology

D. Koval., O. Chumachenko
Faculty of Information Technology and Computer Engineering
NTUU "KPI"
Kiev, Ukraine

***Анотація*—Запропоновано використання еволюційних алгоритмів для формування топології глибоких нейронних мереж.**

***Abstract*—Usage of evolutional algorithms for formation of deep neural network topology is proposed.**

Ключові слова—алгоритм рою частинок; генетичний алгоритм; глибокі нейронні мережі.

Keywords—particle swarm optimisation; genetic algorithm: deep neural network.

I. Вступ

В даний час широке розповсюдження знайшли глибокі нейронні мережі. Синтез оптимальної структури такої нейронної мережі (визначення кількості шарів, нейронів, зв'язків між нейронами) є доволі непростим завданням, оскільки, з одного боку, необхідно забезпечити високу точність роботи, яка залежить від кількісті нейронів. З іншого боку, для уникнення проблеми перенавчання, зменшення обчислювальної складності та підвищення швидкості роботи, необхідно мінімізувати кількість нейронів та зв'язків між ними. В даній роботі пропонується для синтезу оптимальної структури штучних нейронних мереж (ШНМ) скористатися модифікованими версіями генетичного алгоритму і алгоритму рою частинок, а також комбінованим алгоритмом, що поєднує в собі два вищезгаданих, та порівняти їх ефективність.

II. Постановка задачі

Задана вхідна вибірка, елементами якої є пари (x_i,d_i), де x_i – вхідні дані, d_i – еталонні виходи. Необхідно визначити структуру ШНМ (кількість шарів, кількість нейронів у шарах, міжшарові зв'язки) для основної глибокої мережі.

Вирішення поставленої задачі виконується на основі мінімізації помилки роботи мережі (оцінюється по перевірочній вибірці) та складності ШНМ (оцінюється по кількості зв'язків між нейронами).

III. Загальна структура глибокої нейромережі

В загальному вигляді структура нейронної мережі глибокого навчання представлена на рис.1.

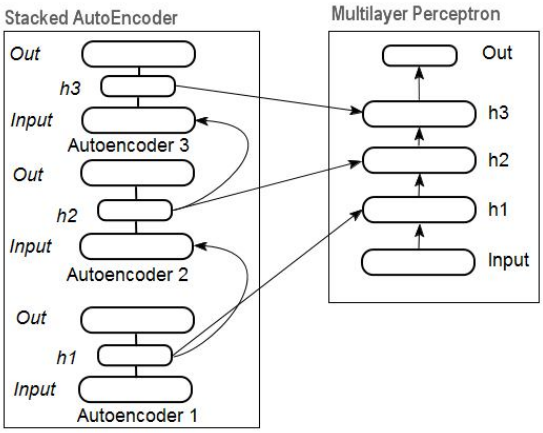

Рис. 1. Глибока нейронна мережа з використанням автоенкодерів

Набір автоенкодерів (або машин Больцмана) використовується для попереднього налаштування ваг основної нейромережі, в якості якої використовується

багатошаровий перцептрон. Використання попереднього налаштування дозволяє прискорити процес навчання та запобігти потраплянню в локальні мінімуми під час навчання. Одним із шляхів підвищення ефективності навчання є попередній вибір оптимальної структури основної нейромережі. Для розв'язання даної задачі можна скористатися еволюційними алгоритмами: методом рою частинок або генетичним алгоритмом.

IV. Огляд методів формування топології ШНМ

A. Алгоритм рою частинок

Щоб зрозуміти алгоритм рою частинок, уявімо *n*-мірний простір (область пошуку), в якому нишпорять частки (агенти алгоритму) [1]. На початку частки розкидані випадковим чином по всій області пошуку і кожна частинка має випадковий вектор швидкості. У кожній точці, де побувала частинка, розраховується значення цільової функції. При цьому кожна частка запам'ятовує, яке (і де) краще значення цільової функції вона особисто знайшла, а також кожна частка знає де розташована точка, яка є найкращою серед усіх точок, які розвідали частки. На кожній ітерації частки коректують свою швидкість (модуль і напрямок), щоб з одного боку бути ближче до кращої точці, яку частка знайшла сама і, в той же час, наблизитися до точки, яка в даний момент є глобально кращою. Через деякий кількість ітерацій частки повинні зібратися поблизу найбільш хорошою точки..

B. Генетичний алгоритм

Для опису генетичних алгоритмів використовується біологічна термінологія, де ключовим поняттям є хромосома, що представляє собою вектор (ланцюжок), утворений нулями і одиницями [2]. Алгоритм починає свою роботу з генерації) початкової популяції хромосом, розмір якої P вважається постійним. Для кожної із хромосом можна оцінити її пристосованість, яка визначається значенням цільової функції E. Далі починається процес репродукції популяції, який формується генетичними операторами кросовера, мутації та інверсії та операцією селекції. У результаті, формується розширена популяція хромосом, що містить як вихідну множину хромосом-батьків, так і множину нащадків. Кожен стринг розширеної популяції оцінюється з точки зору його пристосованості за критерієм E, після чого формується нова популяція W, яка містить P(1) хромосом з найменшими значеннями критерію E. Таким чином, алгоритм накопичує вдалі рішення, «стягуючи» популяцію до глобального екстремуму цільової функції.

V. Модифіковані алгоритми для формування структури нейромережі

В усіх розглянутих нижче алгоритмах для навчання та оцінки якості отриманої конфігурації нейронної мережі вхідна вибірка розбивається на дві частини:

- навчальна вибірка — містить 70% елементів та використовується для навчання ШНМ;
- контрольна вибірка — містить 30% елементів та використовується для оцінки точності роботи ШНМ

A. Алгоритм рою частинок

Нехай необхідно розрахувати кількість нейронів прихованих шарів для n-шарової мережі.

Крок 1. Задаємо розмір популяції (кількість частинок) *P*, ліміт на кількість ітерацій пошуку *L* та задовільне значення помилки роботи нейромережі ε.

Крок 2. Побудуємо випадковим чином набір векторів

$$Y_i(y_1\, y_2\, \ldots\, y_n), \qquad (1)$$

де y_k — кількість нейронів у k-тому шарі,$k=1..n$, $i=1..P$.

Кожен такий вектор відповідатиме одній частинці рою.

Крок 3. Для кожної частинки випадковим чином задаємо її швидкість v_i, і — номер частинки, $i=1..P$.

Крок 4. Для кожної частинки будуємо відповідну їй нейронну мережу. Навчаємо кожну з отриманих мереж та обчислюємо середню квадратичну помилку її роботи на контрольній вибірці даних. Обчислена помилка слугуватиме функцією якості для відповідної частинки.

Крок 5. Для кожної частинки знаходимо мінімальне значення її функції якості за всю історію. Позначаємо значення вектора Y_i (де i — номер частинки), що відповідає знайденому мінімальному значенню функції якості, як Y_i^{lbest} — локальне найкраще рішення для відповідної частинки

Крок 6. Серед усіх частинок знаходимо мінімальне значення функції якості за всю історію рою. Позначаємо значення вектора Y, що відповідає знайденому значенню функції якості, як Y^{gbest} — глобальне найкраще рішення

Крок 7. Модифікуємо швидкість v_i, i — номер частинки, кожної частинки наступним чином [3]:

$$v_{i,t+1} = v_{i,t} + \varphi_p\, r_p\, (Y_i^{lbest} - Y_{i,t}) + \varphi_g\, r_g\, (Y^{gbest} - Y_{i,t}) \qquad (2)$$

де $v_{i,t}$ – швидкість i-ї частинки при t-ій ітерації,

$Y_{i,t}$ – вектор i-ої частинки при t-ій ітерації алгоритму,

Y_i^{lbest} – локальне найкраще рішення i-ої частинки,

Y^{gbest} – глобальне найкраще рішення,

r_p, r_g – випадкові числа з інтервалу (0, 1),

φ_p, φ_g – вагові коефіцієнти, що обираються довільним чином. Є аналогами «швидкості навчання» для ШНМ.

Крок 8. Модифікуємо значення вектора Y_i , що відповідає кожній частинці наступним чином:

$$Y_{i,t+1} = int[Y_{i,t} + v_{i,t+1}] \qquad (3)$$

де i — номер частинки, t — номер ітерації, $int[]$ – ціла частина.

Крок 9. Критерій зупинки.

Якщо знайдене на 6тому кроці значення глобального найкращого рішення забезпечує необхідну точність

$$Y^{gbest} < \varepsilon, \qquad (4)$$

або досягнуто ліміту кількості ітерації

$$t >= L, \quad (5)$$

то закінчуємо процес пошуку. А вектор Y^{gbest} відповідає найкращій знайденій конфігурації нейромережі. Інакше — збільшуємо лічильник ітерацій і переходимо до кроку 4.

B. Генетичний алгоритм

Крок 1. Задаємо розмір популяції (кількість хромосом) *P*, ліміт на кількість ітерацій пошуку *L* та задовільне значення помилки роботи нейромережі ε.

Крок 2. Генеруємо випадковим чином початковий набір хромосом

$$W_i(w_{11}...w_{1k}\, w_{21}...\, w_{2k}\, ...\, w_{n1}...\, w_{nk}), \quad (6)$$

де $w_{p1}...w_{pk}$ - закодований у двійковому форматі вектор, що описує кількість нейронів у p-тому шарі нейронної мережі, $k=1..n$, $i=1..P$.

Кожна хромосома відповідатиме певній архітектурі ШНМ.

Крок 3. Для новостворених хромосом виконуємо операції кроссовера та мутації.

Крок 4. У результаті схрещувань та мутацій отримуємо розширену популяцію хромосом, що містить як вихідну множину хромосом-батьків, так і множину нащадків.

Крок 5. Для кожної хромосоми з розширеної популяції будуємо відповідну їй нейронну мережу. Навчаємо кожну з отриманих мереж та обчислюємо середню квадратичну помилку її роботи на контрольній вибірці даних.. Обчислена помилка слугуватиме функцією пристосованості для відповідної хромосоми.

Крок 6. Формуємо нову популяцію *W*, яка містить *P* хромосом з найменшим значенням функції пристосованості.

Крок 7. Критерій зупинки.

Якщо знайдене на 6тому кроці значення помилки E забезпечує необхідну точність

$$E_p < \varepsilon, \quad (7)$$

або досягнуто ліміту кількості ітерації

$$t >= L, \quad (8)$$

то закінчуємо процес пошуку. А хромосома w_p відповідає найкращій знайденій конфігурації ШНМ. Інакше — збільшуємо лічильник ітерацій і переходимо до кроку 3.

C. Комбінований алгоритм

Алгоритм рою частинок забезпечує вищу точність рішення (за рахунок пам'яті частинок) і швидше отримання прийнятного розв'язку (за рахунок меншої обчислювальної складності). В той же час генетичний алгоритм краще пристосований для розв'язання дискретних проблем і має кращі механізми боротьби з локальними мінімумами (за рахунок мутацій і вдалих кроссоверів). Авторський алгоритм дозволяє поєднати в собі переваги обох алгоритмів і тим самим досягти швидшого і точнішого вирішення поставленої задачі.

Крок 1. Обидва алгоритми (рою частинок і генетичний) запускаються одночасно в паралельному режимі для синтезу структури однієї і тієї нейромережі.

Крок 2. Виконується одна ітерація кожного алгоритму відповідно до описаної в попередніх пунктах методики.

Крок 3. Після кожної ітерації порівнюються результати знайдені обома алгоритмами і вибирається найкраще рішення.

Нехай $Y^{(i)}_{pso}$ – найкраще рішення знайдене алгоритмом рою частинок на і-тій ітерації, а $W^{(i)}_{ga}$ – найкраще рішення знайдене генетичним алгоритмом.

Якщо

$$E(Y^{(i)}_{pso}) < E(W^{(i)}_{ga}), \quad (9)$$

тобто, отримане алгоритмом рою частинок рішення забезпечує менше значення функції якості *E*, то переходимо до кроку 4a. Інакше–переходимо до кроку 4b.

Крок 4a. Замінюємо найгірше рішення генетичного алгоритму $W^{(i)}_{ga\_worst}$ рішенням $Y^{(i)}_{pso}$

$$W^{(i)}_{ga\_worst} := Y^{(i)}_{pso} \quad (10)$$

І переходимо до кроку 5.

Крок 4b. Замінюємо найгірше рішення алгоритму рою частинок $Y^{(i)}_{pso\_worst}$ рішенням $W^{(i)}_{ga}$,

$$Y^{(i)}_{pso\_worst} := W^{(i)}_{ga} \quad (11)$$

І переходимо до кроку 5.

Крок 5. Якщо обидва алгоритми продовжують свою роботу (тобто, критерій зупинки не виконується для жодного з них), то переходимо до кроку 2.

Даний підхід повторюється до зупинки одного з алгоритмів. В якості остаточного рішення приймається краще із рішень, знайдених обома алгоритмами на момент зупинки.

Висновки

Запропоновано алгоритм формування топології ШНМ, який дозволяє сформувати оптимальну з точки зору точності, швидкості навчання та швидкості роботи ШНМ. Алгоритм було перевірено на вибірці MNIST і помилка роботи глибокої ШНМ скоротилася з 2,51% до 2,09%.

Література References

[1] Алгоритм роя частиц. Описание и реализации на языках Python и C#. http://jenyay.net/Programming/ParticleSwarm#outro

[2] Е. В. Бодянский, О. Г. Руденко Искусственные нейронные сети: архитектуры, обучение, применения. – Х: Телетех, 2004. – 369 с.

[3] J. Kennedy and R. Eberhart, Swarm Intelligence, Academic Press, 1st ed., San Diego, CA, 2001