***5.2. Перенавчання і узагальнення***. Одна з найбільш серйозних труднощів висловленого підходу полягає у тому, що таким чином ми мінімізуємо не ту помилку, яку насправі потрібно мінімізувати – помилку, яку можна чекати від мережі, коли їй подаватимуться абсолютно нові спостереження.

Інакше кажучи, ми хотіли б, щоб нейронна мережа мала здатність узагальнювати результат на нові спостереження. У дійсності мережа навчається мінімізувати помилку на досліджуваній множині, і при відсутності ідеальної і нескінченно великої навчальної множини це зовсім не те ж саме, що мінімізувати «справжню» помилку на поверхні помилок у наперед невідомій моделі явища.

Сильніше за все ця відмінність виявляється у *проблемі перенавчання*, або дуже близької підгонки. Це явище буде простіше продемонструвати не для нейронної мережі, а на прикладі апроксимації за допомогою поліномів – при цьому суть явища абсолютно така ж сама.

*Поліном (або многочлен)* – це вираз, що містить тільки константи і цілі ступені незалежної змінної.

Приклади: $y = 2x + 3, \quad y = 3x^2 + 4x + 1.$

Графіки поліномів можуть мати різну форму, причому чим вище ступінь многочлена (і, тим самим, чим більше членів в нього входить), тим більш складною може бути ця форма. Якщо у нас є деякі дані, ми можемо поставити мету підігнати до них поліноміальну криву (модель) і одержати таким чином пояснення для даної залежності. Наші дані можуть бути зашумлені, тому не можна вважати, що найкраща модель задається кривою, яка в точності проходить через всі наявні точки. Поліном низького порядку може бути *недостатньо гнучким* засобом для апроксимації даних, тоді як поліном високого порядку може виявитися *занадто гнучким*, і точно слідуватиме даним, приймаючи при цьому складну форму, що не має ніякого відношення до форми справжньої залежності.

Нейронна мережа стикається з точно такою ж трудністю. Мережі з великим числом вагів моделюють складніші функції і, отже, схильні до *перенавчання*. Мережа ж з невеликою кількістю вагів може виявитися недостатньо гнучкої, щоб змоделювати наявну залежність.

Як же вибрати «правильний» ступінь складності для мережі? Майже завжди складніша мережа дає меншу помилку, але це може свідчити не про хорошу якість моделі, а про перенавчання.

Відповідь полягає в тому, щоб використовувати механізм *контрольної крос-перевірки*, при якій частина навчальних спостережень резервується і в навчанні по алгоритму зворотного розповсюдження не використовується. Натомість, по мірі роботи алгоритму, вона використовується для незалежного контролю результату.

На самому початку роботи помилки мережі на навчальній і контрольній множинах будуть однаковими (якщо вони істотно відрізняються, то, ймовірно, розбиття всіх прикладів на дві множини було неоднорідне). У міру того як мережа навчається, помилки навчання, поволі зменшуються, і поки навчання зменшує дійсну функцію помилок, помилка на контрольній множині також буде зменшуватись. Якщо ж контрольна помилка перестала зменшуватись або навіть стала зростати, це вказує на те, що мережа почала дуже близько апроксимувати дані і навчання слід зупинити. Це явище занадто точної апроксимації у процесі

навчання і *називається перенавчанням*. Якщо таке трапилося, то звичайно радять зменшити число прихованих елементів або шарів, бо мережа є дуже потужною для даної задачі. Якщо ж мережа, навпаки, була взята недостатньо складна для того, щоб моделювати дану залежність, то перенавчання, швидше за все, не відбудеться, і обидві помилки – навчання і перевірки – не досягнуть достатнього рівня малості.

Описані проблеми з локальними мінімумами і вибором розміру мережі призводять до того, що при практичній роботі з нейронними мережами, як правило, доводиться експериментувати з великою кількістю різних мереж, деколи навчаючи кожну з них по кілька разів (щоб не бути введеним в оману локальними мінімумами) і порівнювати одержані результати. Головним показником якості результату є *контрольна помилка*. При цьому відповідно до загальнонаукового принципу, згідно якому при інших рівних слід віддати перевагу більш простій моделі, з двох мереж з приблизно рівними помилками контролю має сенс вибрати ту, яка менше.

Класичний метод зворотного розповсюдження відноситься до алгоритмів з лінійною збіжністю. Для збільшення швидкості збіжності необхідно використовувати матриці других похідних функції помилки.

Були запропоновані численні модифікації алгоритму зворотного розповсюдження, які зв'язані з використанням різних функцій помилки, різних процедур визначення напряму і величини кроку.

1. *Функції помилки:*
   - інтегральні функцій помилки на всій сукупності навчальних прикладів;
   - функції помилки цілих і дробових ступенів.
2. *Процедури визначення величини кроку на кожній ітерації:*
   - дихотомія;
   - інерційні співвідношення;
   - відпал.
3. *Процедури визначення напряму кроку:*
   - з використанням матриці похідних другого;
   - з використанням напрямів на декількох кроках.