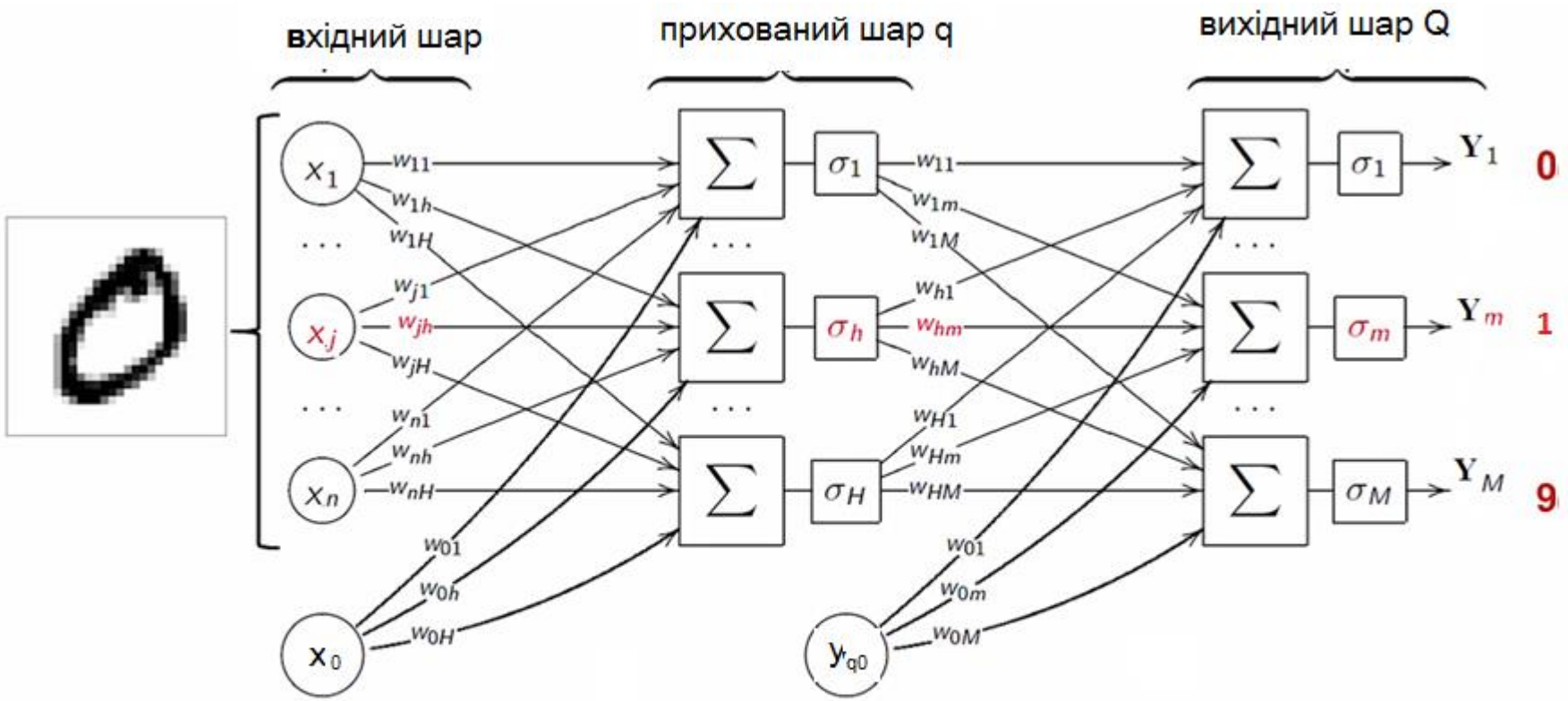
Лекція 2 до модуля 3

Багатошарові нейронні мережі

Багатошарова НМ



Градієнтний спуск

Цільова функція помилки

$$E(w_{ij}^{(q)}) = \frac{1}{2}\sum_{p}\sum_{j=1}^{M} (y_{j,p}^{(Q)} - d_{j,p})^2 \to \min$$

Мінімізуємо функцію помилки для кожного прикладу по черзі

Тоді $$E_p(w_{ij}^{(q)}) = \frac{1}{2}\sum_{j=1}^{M} (y_{j,p}^{(Q)} - d_{j,p})^2 = \frac{1}{2}\sum_{j=1}^{M} \Delta_{j,p}^2 \to \min$$

$$E(w_{ij}^{(q)}) = \sum_{p=1}^{P} E_p(w_{ij}^{(q)})$$

Градієнтний спуск

Згідно з методом градієнтного спуску - крок η

$$w_{ij}^{(q)}(t+1) = w_{ij}^{(q)}(t) + \Delta w_{ij}^{(q)} = w_{ij}^{(q)}(t) - \eta \cdot \frac{\partial E_p(w_{ij}^{(q)}(t))}{\partial w_{ij}^{(q)}}$$

Так як $$\Delta_{j,p} = (y_{j,p}^{(Q)} - d_{j,p})$$

То при *q=Q* $$\frac{\partial E_p}{\partial w_{ij}^{(Q)}} = \frac{\partial E_p}{\partial \Delta_{j,p}} \cdot \frac{\partial \Delta_{j,p}}{\partial y_{j,p}^{(Q)}} \cdot \frac{\partial y_{j,p}^{(Q)}}{\partial w_{ij}^{(Q)}}$$

$$\frac{\partial E_p}{\partial \Delta_{j,p}} = \Delta_{j,p}; \frac{\partial \Delta_{j,p}}{\partial y_{j,p}^{(Q)}} = 1;$$

Оскільки $$y_{j,p}^{(Q)} = F(\sum_{i=0}^{n_{Q-1}} w_{ij}^{(Q)} y_{i,p}^{(Q-1)}) = F(s_{j,p}^{(Q)})$$

Градієнтний спуск

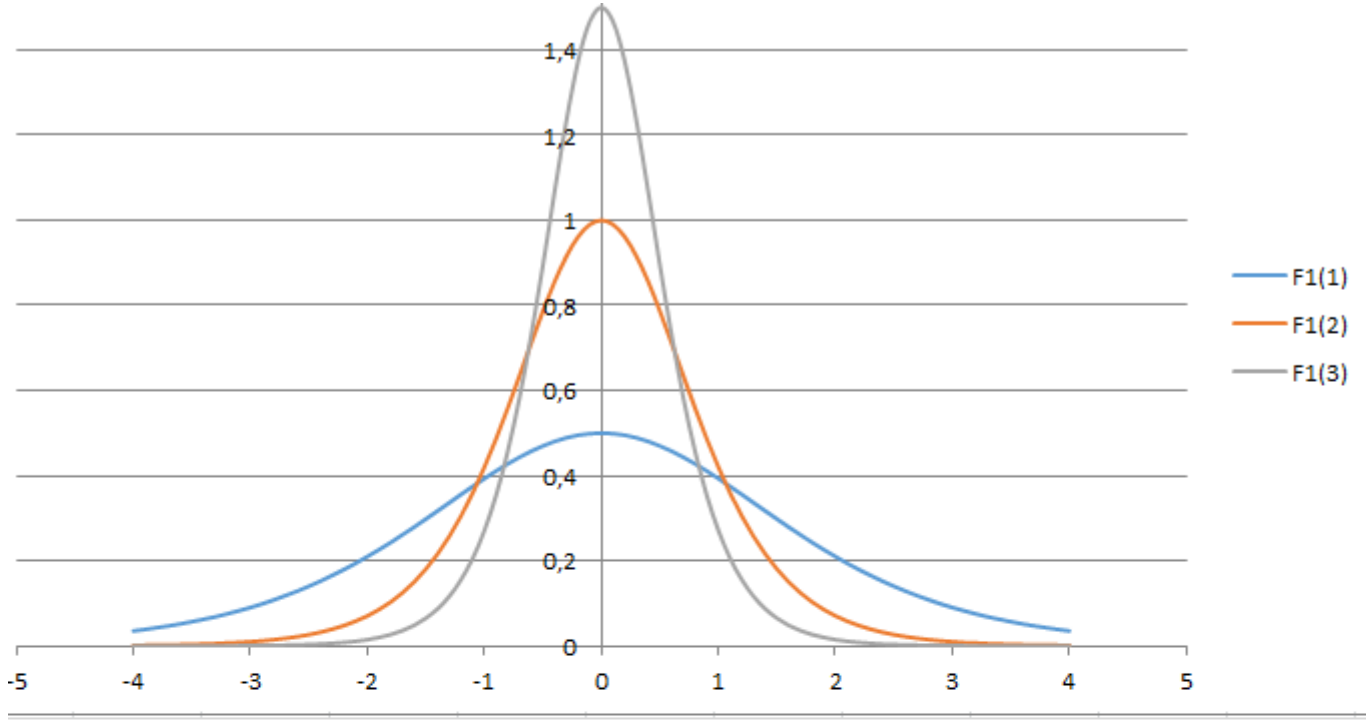
то

$$\frac{\partial y_{j,p}^{(Q)}}{\partial w_{ij}^{(Q)}} = F'(s_{j,p}^{(Q)}) \cdot \frac{\partial s_{j,p}^{(Q)}}{\partial w_{ij}^{(Q)}} = F'(s_{j,p}^{(Q)}) \cdot y_{i,p}^{(Q-1)}$$

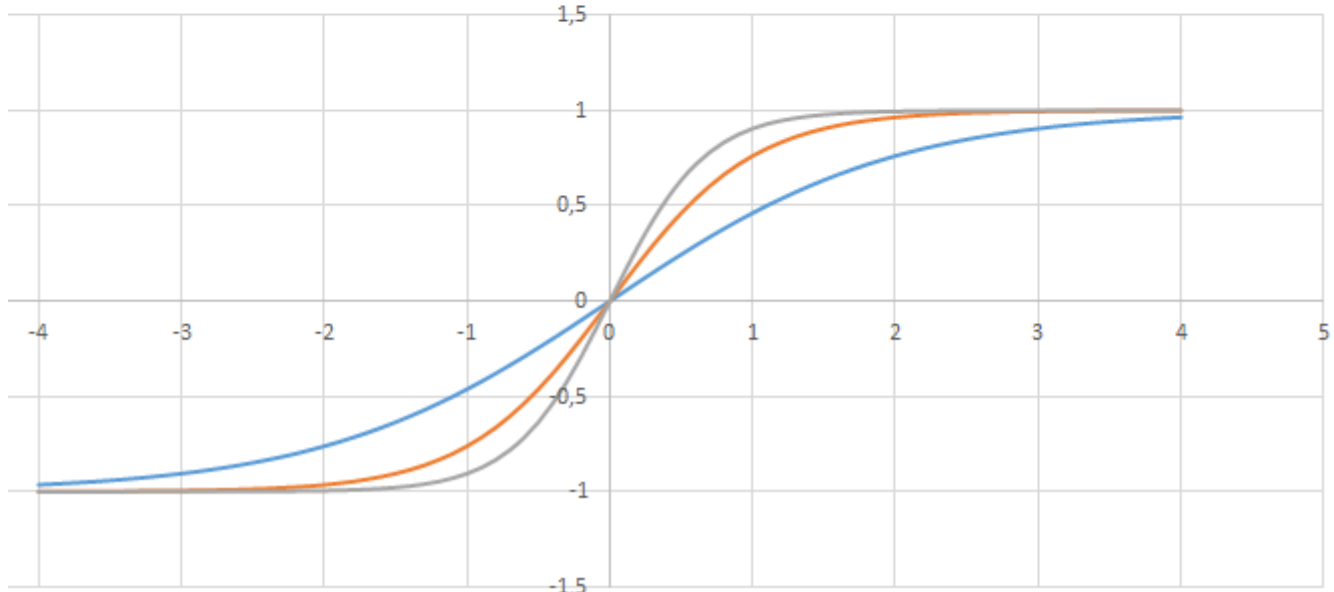
Оскільки множник $\frac{\partial y_{j,p}^{(Q)}}{\partial w_{ij}^{(Q)}}$ є похідною функції по її аргументу, з цього виходить, що похідна активаційної функція має бути визначена на усій осі абсцис, тому застосовуються такі функції, як гіперболічний тангенс або класичний сигмоїд з експонентою.

$$F(s) = \frac{2}{1+e^{-a \cdot s}} - 1 = th(\frac{a \cdot s}{2})$$

$$F'(s) = \frac{a}{2 \cdot ch^2(\frac{a \cdot s}{2})} = \frac{a}{2}(1 - th^2(\frac{a \cdot s}{2})) = \frac{a}{2}(1 - F^2(s))$$


1,4
1,2
1
0,8
0,6
0,4
0,2
0
-5
-4
-3
-2
-1
0
1
2
3
4
5
F1(1)
F1(2)
F1(3)

Сігмоїд


1,5
1
0,5
0
-0,5
-1
-1,5
-4
-3
-2
-1
0
1
2
3
4
5

Градієнтний спуск

При одношаровій мережі Q=1, $y^{(0)}=x_i$

$$\frac{\partial E_p}{\partial w_{ij}^{(Q)}} = \Delta_{j,p} \cdot F'(s_{j,p}^{(Q)}) \cdot x_{i,p}$$

Формула для уточнення $w_{ij}^{(Q)}$ матиме вигляд

$$w_{ij}^{(Q)}(t+1) = w_{ij}^{(Q)}(t) - \eta \cdot (\Delta_{j,p} \cdot F'(s_{j,p}^{(Q)})) \cdot x_{i,p}$$

І процес навчання буде повторювати процедуру Відроу-Хебба, тобто уточнення

Для багатошарових НМ невідомі помилки у проміжних шарах

Алгоритм Back propagation error

Запропонований Дэвидом И. Румельхартом, Дж. Е. Хинтоном и Рональдом Дж. Вильямсом в работе

***Rumelhart D.E., Hinton G.E., Williams R.J.*, Learning Internal Representations by Error Propagation. In: Parallel Distributed Processing, vol. 1, pp. 318—362. Cambridge, MA, MIT Press. 1986.**

Згідно з методом найменших квадратів, цільовою функцією помилки НС, що мінімізується, є величина:

$$E(w) = \frac{1}{2}\sum_{j,p}\left(y_{j,p}^{(Q)} - d_{j,p}\right)^2 \qquad (2.1)$$

де $y_{j,p}^{(N)}$ - реальний вихідний стан нейрона j вихідного шару N нейронної мережі при подачі на її входи p -го образу;

d_{jp} - ідеальний (бажане) вихідний стан цього нейрона.

Підсумовування ведеться по усіх нейронах вихідного шару і по усіх оброблюваних мережею образах.

Алгоритм Back propagation error

Мінімізація ведеться методом градієнтного спуску, що означає підстроювання вагових коефіцієнтів таким чином:

$$w_{ij}^{(q)}(t+1) = w_{ij}^{(q)}(t) + \Delta w_{ij}^{(q)} = w_{ij}^{(q)}(t) - \eta \cdot \frac{\partial E_p}{\partial w_{ij}^{(q)}} \qquad (2)$$

Тут $\boldsymbol{w_{ij}}$ - ваговий коефіцієнт синаптичного зв'язку, що сполучає і –й нейрон шару n – 1 з j –м нейроном шару n,

η - коефіцієнт швидкості навчання, 0< η ≤1,

Очевидно що,

$$\frac{\partial E}{\partial w_j^{(q)}} = \frac{\partial E}{\partial y_j^{(q)}} \cdot \frac{dy_j^{(q)}}{ds_j^{(q)}} \cdot \frac{\partial s_j^{(q)}}{\partial w_j^{(q)}} \qquad (3)$$

Тут під y_j, як і раніше, мається на увазі вихід нейрона j, а під s_j – зважена сума його вхідних сигналів, тобто аргумент активаційної функції.

$$\frac{dy}{ds} = F'(s) = \frac{1}{2}(1 - y^2) \qquad (4)$$

Алгоритм Back propagation error

Третій множник ds_j/dw_{ij}), очевидно, дорівнює виходу нейрона попереднього шару: $y_i^{(q-1)}$.

$$s_j^{(q)} = \sum_{i=0}^{n_q} y_i^{(q-1)} \cdot w_{ij}^{(q)} \qquad s_j^{(q+1)} = \sum_{i=0}^{n_q} y_i^{(q)} \cdot w_{ij}^{(q+1)}$$

Що стосується першого множника в (3), він легко розкладається таким чином:

$$\frac{\partial E}{\partial y_j^{(q)}} = \sum_{i=0}^{n_{q+1}} \frac{\partial E}{\partial y_i^{(q+1)}} \cdot \frac{dy_i^{(q+1)}}{ds_i^{(q+1)}} \cdot \frac{\partial s_i^{(q+1)}}{\partial y_i^{(q+1)}} = \sum_{i=0}^{n_{q+1}} \left(\frac{\partial E}{\partial y_i^{(q+1)}} \cdot \frac{dy_i^{(q+1)}}{ds_i^{(q+1)}}\right) \cdot w_{ji}^{(q+1)} \tag{5}$$

Тут підсумовування по і виконується серед нейронів шару *n+1*.

Ввівши нову змінну

$$\delta_j^{(q)} = \frac{\partial E}{\partial y_j^{(q)}} \cdot \frac{dy_j^{(q)}}{ds_j^{(q)}} \tag{6}$$

отримаємо рекурсивну формулу для розрахунків величин (j(n) шару n з величин (k(n+1) більше старшого шару n+1.

$$\delta_j^{(q)} = \left[\sum_{i=0}^{n_{q+1}} \delta_k^{(q+1)} \cdot w_{jk}^{(q+1)}\right] \cdot F'(s_j^{(q)}) \tag{7}$$

Для вихідного ж шару

$$\delta_j^{(Q)} = (y_j^{(Q)} - d_j) \cdot F'(s_j^{(Q)})$$

Алгоритм Back propagation error

(8)

Тепер ми можемо записати (2) в розкритому виді:

$$\delta_j^{(q)} = \left[\sum_{i=0}^{n_{q+1}} \delta_k^{(q+1)} \cdot w_{jk}^{(n+1)} \right] \cdot F'(s_j^{(q)}) \qquad (9)$$

Іноді для надання процесу корекції ваг деякої інерційності, що згладжує різкі скачки при переміщенні по поверхні цільової функції, (9) доповнюється значенням зміни ваги на попередній ітерації

$$\Delta w_{ij}^{(q)} = -\eta \cdot \delta_j^{(q)} \cdot y_i^{(q-1)},$$

$$\Delta w_{ij}^{(q)}(t+1) = -\eta \cdot (\mu \cdot \Delta w_{ij}^{(q)}(t) + (1-\mu) \cdot \delta_j^{(q)} \cdot y_i^{(q-1)}) \qquad (10)$$

де μ - коефіцієнт інерційності, t - номер поточної ітерації.

Реалізація Backpropogation ERROR

Алгоритм навчання багатошарової НМ за допомогою процедури зворотного поширення помилки :

0. Зформувати навчальну вибірку {$\mathbf{x}_p$, $\mathbf{d}_p$}, p=1…P

ініціалізувати початкові значення вагових параметрів

$$w_{ij} = 0.01*rand(-1,+1).$$

1. Подати на входи мережі один прикладів навчальної вибірки і в режимі звичайного функціонування НМ, коли сигнали поширюються від входів до виходів, розрахувати значення останніх. Нагадаємо, що

$$s_j^{(q)} = \sum_{i=0}^{n_{q-1}} y_i^{(q-1)} \cdot w_{ij}^{(q)}$$

де n_{q-1} - число нейронів в шарі q-1 з урахуванням нейрона з постійним вхідним станом +1, що задає зміщення;

$$y_j^{(q)} = F(s_j^{(q)}),$$

де F() – функція активації

$$y_i^{(0)} = x_i$$

де x_i - і -та компонента вектору вхідного образу.

2. Розрахувати для вихідного шару по формулі $\delta_j^{(Q)} = (y_j^{(Q)} - d_j) \cdot F'(s_j^{(Q)})$

Реалізація Backpropogation

3. Розрахувати по формулах відповідні для усіх інших шарів,

q=(Q – 1),..1.

$$\delta_j^{(q)} = \left[\sum_{i=0}^{n_{q+1}} \delta_k^{(q+1)} \cdot w_{jk}^{(q+1)} \right] \cdot F'(s_j^{(q)})$$

4. Розрахувати по формулі

$$\Delta w_{ij}^{(q)} = -\eta \cdot \delta_j^{(q)} \cdot y_i^{(q-1)}$$

зміни ваг шару q.

5. Скоригувати усі ваги в НМ

$$w_{ij}^{(q)}(t) = w_{ij}^{(q)}(t-1) + \Delta w_{ij}^{(q)}(t)$$

6. Перейти на крок 1 до вичерпання всіх прикладів навчальної вибірки.

7. Якщо функція помилки мережі E для всіх M прикладів навчальної вибірки істотна, перейти на крок 1. Інакше - кінець.

$$E = \frac{1}{2} \sum_{p=1}^{P} \sum_{j=1}^{n_Q} \Delta_{jp}^2$$

де $\Delta_{jp} = (y_j^{(Q)} - d_j)$

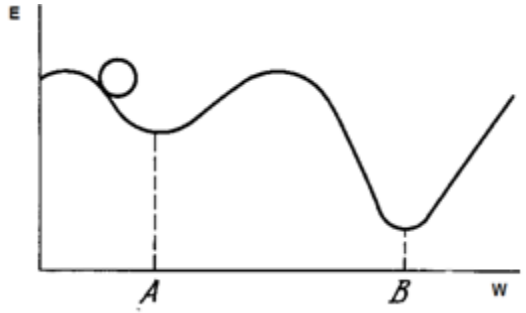
Мережі на кроці 1 поперемінно у випадковому порядку пред'являються усі тренувальні приклади.

Налаштування НМ для вирішення задач

- Локальні мінімуми.
- Параліч НМ
- Розмір кроку.
- Тимчасова нестійкість.
- Попередня обробка вхідних даних.
- Згасання помилки у алгоритмі Backprop.

Локальні мінімуми

BackProp використовує різновид градієнтного спуску, тобто здійснює спуск вниз по поверхні помилки, безперервно налаштовуючи ваги у напрямі до мінімуму. Поверхня помилки складної мережі дуже порізана і складається з пагорбів, долин, складок і ярів в просторі високої розмірності.



Мережа може потрапити в локальний мінімум (неглибоку долину), коли поруч є набагато глибший мінімум. У точці локального мінімуму усі напрями ведуть вгору, і мережа нездатна з нього вибратися.

Статистичні методи навчання можуть допомогти уникнути цієї пастки, але вони повільні. Застосовується також метод «струшування», тобто зміна деяких вагових параметрів для виходу з локального мінімуму і продовження процесу навчання.

Параліч мережі

В процесі навчання мережі значення ваг можуть в результаті корекції стати дуже великими величинами. Це може привести до того, що усі або більшість нейронів функціонуватимуть при дуже великих значеннях S, в області, де похідна функції дуже мала. Оскільки помилка пропорційна цій похідній, то процес навчання може практично завмерти.

Різні евристики використовувалися для оберігання від паралічу або для відновлення після нього, наприклад тимчасове зменшення параметра сигмоїду **a**, оскільки причиною паралічу НМ часто є нульове значення похідної від активаційної функції $F'(S) = \boldsymbol{\alpha}(1-F^2(S))$ для великих значень S і це зупиняє процес навчання. Після відновлення навчання значення параметра $\boldsymbol{\alpha}$ повертається.

Розмір кроку

Уважний розбір збіжності показує, що корекції ваг передбачаються нескінченно малими. Ясно, що це нездійсненно на практиці, оскільки веде до нескінченного часу навчання.

Розмір кроку **η** повинен братися скінченним, і в цьому питанні доводиться спиратися тільки на досвід.

Якщо розмір кроку **η** дуже малий, то збіжність занадто повільна, якщо ж дуже великий, то може виникнути параліч або постійна нестійкість.

Автоматичне налаштування кроку може здійснюватись по формулі: $\eta = \eta_0/(1+b*t)$

t - номер епохи навчання,
η_0 і **b –** обираються експериментально.

Тимчасова нестійкість

Якщо мережа вчиться розпізнавати букви, то немає сенсу учити "Б", якщо при цьому забувається "А". Процес навчання має бути таким, щоб мережа навчалася на усій навчальній множині без пропусків того, що вже вивчене.

Необхідні зміни ваг повинні обчислюватися **на усій множині прикладів,** а це вимагає додаткової пам'яті; після ряду таких навчальних циклів вони зійдуться до мінімальної помилки.

Цей метод може виявитися даремним, якщо мережа знаходиться в зовнішньому середовищі, що постійно міняється, так що другий раз один і той же вектор може вже не повторитися. В цьому випадку процес навчання може ніколи не зійтися, безцільно блукаючи або сильно осцилюючи. У цьому сенсі зворотне поширення не схоже на біологічні системи.

Попередня обробка вхідних даних.

- Процес навчання можна суттєво покращити попередньою обробкою навчальної вибірки.
- Приклади для кожного класу повинні бути типовими проте різними. При майже однакових прикладах спостерігається синдром «перенавчання», коли нейронна мережа у процесі функціонування добре розпізнає образи близькі до прикладів і не розпізнають ся віддалені образи.
- Разом з тим, дуже «зашумлені» приклади у навчальній вибірці не дозволяють навчити НС взагалі.
- Суттєво може прискорити навчання попередня нормалізація навчальної вибірки, наприклад методом природної чи стандартної нормалізації.

Функції активації

- Логістичний сигмоїд $F(s)=\frac{1}{1+e^{-a\cdot s}}$ $F'(s)=a\cdot F(s)\cdot(1-F^2(s))$
- Гіперболічний тангенс $F(s)=th(as)$ $F'(s)=a(1-F^2(s))$
- SoftSign $F(s)=\frac{a\cdot s}{1+|a\cdot s|}$ $F'(s)=\frac{1}{(1+|a\cdot s|)^2}$
- SoftPlus $F(s)=\ln(1+e^{a\cdot s})$ $F'(s)=\frac{a}{1+e^{-a\cdot s}}$
- ReLU(rectified linear units) $F(s)=\max(0,s)$

$F'(s)=\max(0,1)$

Згасання помилки у алгоритмі BackProp.

Спостерігається згасання помилки при зворотному розповсюдженні, що зупиняє процес навчання.

Виходом можуть бути застосування інших методів навчання, зокрема, чисельних методів оптимізації чи обмеженої машини Больцмана.

Лабораторна робота № 6

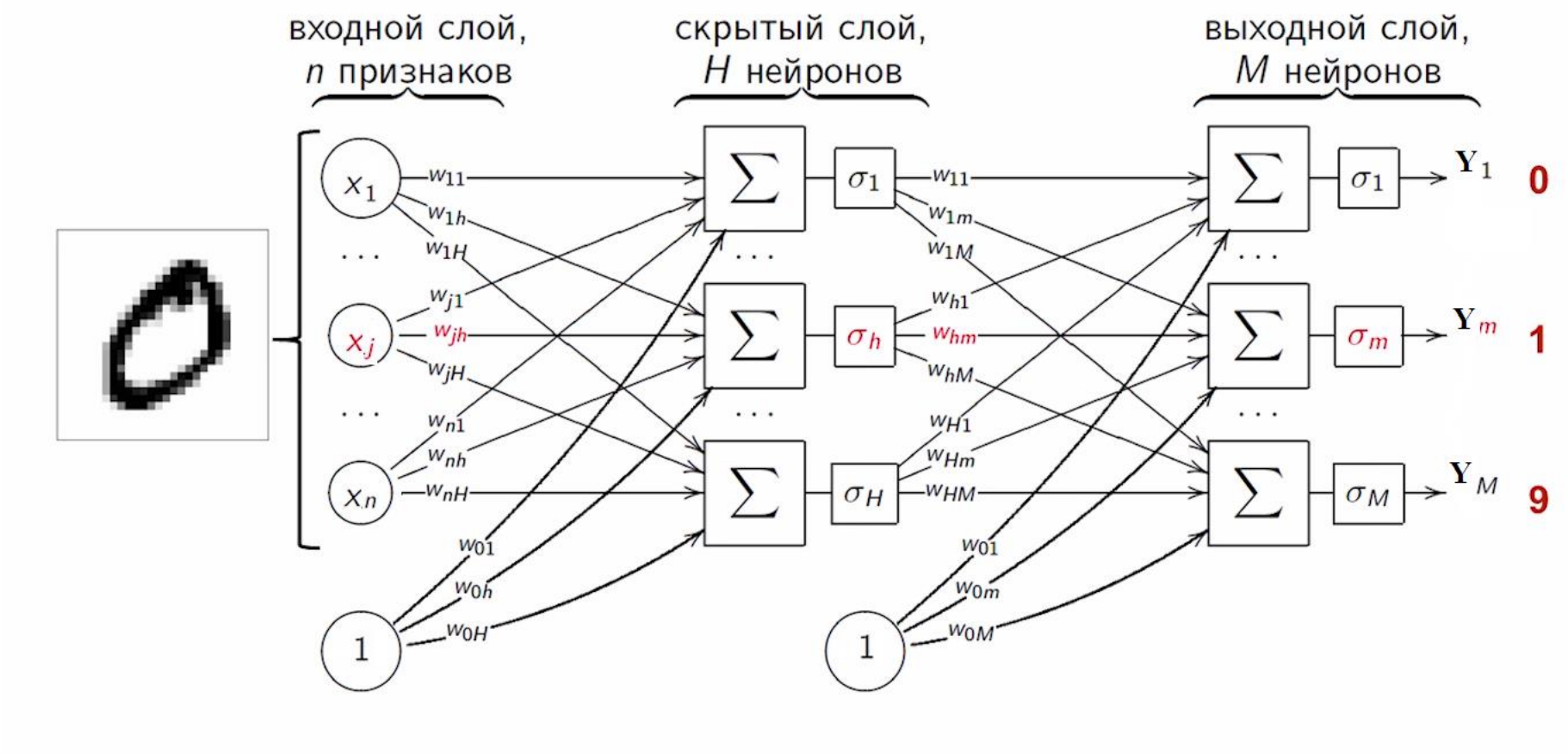
Створити нейро комп’ютерну систему розпізнавання рукописних символів, використавши алгоритм зворотного поширення помилки для навчання багатошарової НМ.

Алгоритм лабораторної роботи №3

1. Створити графічний інтерфейс для введення прикладів і розпізнавання символів.

2. Створити навчальну вибірку рукописних символів або використати стандартний набір MNIST з Internet

Лабораторна робота № 2



Лабораторна робота №6

3.Спроектувати нейронну мережу, задаючись на вході достатньою кількістю рецепторів для відображення символів через мега пікселі двовимірної сітки, а на виході НС достатньою кількістю нейронів для кодування символів (кількість різних символів <=2^m,

де m – кількість нейронів на виході).

4.Кількість нейронів у проміжних шарах визначити експериментально.

5.Навчити НМ згідно з алгоритмом BackProp.

6. Протестувати роботу НС на контрольній частині навчальної вибірки і при необхідності донавчити НС.