Машинно обучение

Лабораторно упражнение 8

Обучение на невронна мрежа чрез Backpropagation алгоритъм

Упражнението демонстрира обучение на невронна мрежа чрез алгоритъм backpropagation. Целта е да се намерят такива стойности на параметрите (теглата на връзките) в мрежата, при които функцията на цената, оценяваща точността на мрежата, има минимална стойност.

Като пример се разглежда задачата за разпознаване на ръкописни цифри от 0 до 9.

1. Обучителни данни

Файлът data.mat съдържа 5000 форматирани изображения на ръкописни цифри от 0 до 9. Данните са взети от еталонната база данни MNIST (http://yann.lecun.com/exdb/mnist/), съдържаща множество примери за ръкописни цифри. Файлът е в .mat формат, т.е. данните са записани в MATLAB матрична форма вместо в текстов ASCII формат. Изображението на всяка цифра се представя с матрица, която се зарежда директно в програмата.

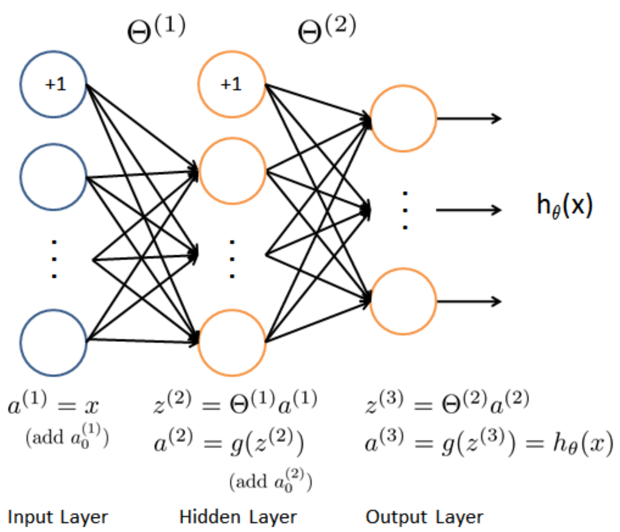
Във файла са дадени 5000 обучителни примери, всеки от които е изображение, представено с матрица от пиксели 20x20. Всеки пиксел е представен с числова стойност за неговия интензитет от сивата скала. Матрицата 20x20 се преобразува във вектор 400x1. Като резултат всеки обучителен пример става един ред в матрицата X. Тя има размер 5000 x 400 и всеки ред в нея и обучителен пример с изображение на ръкописна цифра.

Обучителните примери съдържат също вектор y (5000x1) с етикети labels) за всяка цифра.

2. Невронна мрежа

2.1. Модел на мрежата

Мрежата се състои от 3 слоя - входен слой (input layer), скрит слой (hidden layer) и изходен слой (output layer) (фиг. 1). Входните данни са стойностите на интензитетите на пикселите от изображението на цифра. Тъй като изображенията са с размерност 20x20, входният слой има 400 върха (+1 за допълнителен връх за т.нар. отклонение (bias unit)).



Фиг. 1. Модел на невронна мрежа за разпознаване на ръкописни цифри

$\theta^{(1)}$ - матрица със стойности на параметри в първия слой (input layer)

$\theta^{(2)}$ - матрица със стойности на параметри в скрития слой (hidden layer)

$\theta^{(1)}$ се зарежда в Theta1 с размерност 25x401, а $\theta^{(2)}$ се зарежда в Theta2 с размерност 10x26. Тези размерност са определени от това, че във втория (скрития слой) слой има 25 върха, а в изходния слой има 10 върха (по един за всеки възможен клас).

Файлът weights.mat съдържа стойности на предварително обучени параметри.

2.2. Feedforward Propagation и функция на цената (cost function)

Feedforward propagation е алгоритъм за изчисляване на стойностите във върховете от всеки слой на невронната мрежа. В дадената невронна мрежа стойностите във върховете се изчисляват по следния начин:

За първи слой (input layer):

Добавяме връх $a_0^{(1)} = 1.$

$$a^{(1)} = X$$

, където

$a^{(1)}$ - стойности на върховете на мрежата в първия слой

X - матрица със стойности на входните променливи

За втори слой (hidden layer)

Добавяме връх $a_0^{(2)} = 1.$

$$z^{(2)} = \theta^{(1)}a^{(1)}$$

$$a^{(2)} = g(z^{(2)})$$

$\theta^{(1)}$ - стойности на параметрите в първи слой на мрежата

$a^{(1)}$ - стойности на върховете в първи слой на мрежата

$g(z^{(2)})$ - сигмоидна функция с параметър $z^{(2)}$

За трети слой (output layer)

$$z^{(3)} = \theta^{(2)}a^{(2)}$$

$$a^{(3)} = g\left(z^{(3)}\right) = h_\theta(x)$$

$\theta^{(2)}$ - стойности на параметрите във втори слой на мрежата

$a^{(2)}$ - стойности на върховете във втори слой на мрежата

$g(z^{(3)})$ - сигмоидна функция с параметър $z^{(3)}$

Стойността на хипотезата $h_\theta(x)$ се получава в изходния слой на мрежата като стойност в интервала 0-1 за всеки възможен клас. Тази стойност определя вероятността за принадлежност на входните данни към дадения клас. Крайният резултат от невронната мрежа е класът с най-голяма стойност на хипотезата, т.е. класът с най-голяма вероятност.

Функцията на цената се изчислява по следния начин:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[-y_k^{(i)}\log((h_\theta(x^{(i)})_k\right) - \left(1 - y_k^{(i)}\right)\log(1 - \left(h_\theta(x^{(i)})\right)_k)] + \frac{\lambda}{2m}[\sum_{j=1}^{25}\sum_{k=1}^{400}(\theta_{j,k}^{(1)})^2 + \sum_{j=1}^{10}\sum_{k=1}^{25}(\theta_{j,k}^{(1)})^2]$$

, където $h_\theta(x)$ се изчислява, както е показано на фиг. 1, а K=10 - общ брой на възможните класове. Обърнете внимание, че:

- $h_\theta\left(x^{(i)}\right)_k = a_k^{(3)}$ е изходната стойност във връх k от изходния слой;
- Докато оригиналните класове имат стойности 1,2,3,…,10, записани в y, то за обучението на невронната мрежа е необходимо тези стойности да бъдат представени като вектори, съставени само от 0 и 1, т.е.

$$y = \begin{bmatrix}1\\0\\0\\\dots\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\\\dots\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\\\dots\\0\end{bmatrix} \dots \begin{bmatrix}0\\0\\0\\\dots\\1\end{bmatrix}$$

Напр., ако $x^{(i)}$ е изображение на цифрата 5, то съответстващият вектор $y^{(i)}$, използван във функцията на цената, е 10-разреден вектор с $y_5 = 1$ и 0 в останалите позиции.

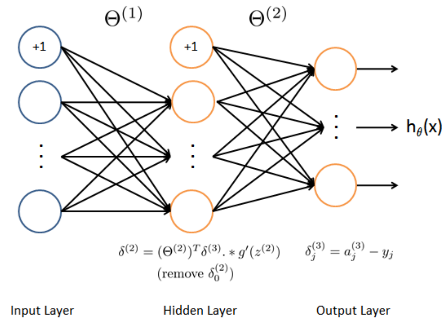
2. Алгоритъм Back propagation

2.1. Градиент на сигмоидната функция

Градиентът на сигмоидната функция се изчислява по следния начин:

$g'(z) = \frac{d}{dz}g(z) = g(z)(1 - g(z))$, където $sigmoid(z) = g(z) = \frac{1}{1+e^{-z}}$

2.2. Back propagation



Фиг. 2. Алгоритъм Back propagation

За всеки обучителен пример $(x^{(t)}, y^{(t)})$ изпълняваме forward propagation за изчисляване на стойностите във върховете до получаване на стойността на хипотезата $h_\theta(x)$ в изходния слой. След това за всеки връх *j* в слой *l* изчисляваме „грешка“ $\delta_j^{(l)}$, която измерва доколко даден връх е отговорен за крайната грешка в изхода на мрежата.

За върховете в изходния слой можем директно да изчислим грешката $\delta_j^{(3)}$ чрез стойността *y* от обучителните примери (слой 3 е изходен). За върховете в скрития слой грешката $\delta_j^{(l)}$ се изчислява чрез усреднената грешка в слой (*l+1*).

Стъпки на Back propagation алгоритъма:

1. На стойностите на върховете $a^{(1)}$ във входния слой присвояваме стойностите от $t^{\text{ти}}$ обучителен пример $x^{(t)}$. След това изпълняваме feedforward за изчисляване на стойностите $z^{(2)}, a^{(2)}, z^{(3)}, a^{(3)}$ в слоеве 2 и 3.

2. За всеки връх *k* в слой 3 (изходен слой) изчисляваме грешката

$$\delta_k^{(3)} = (a_k^{(3)} - y_k)$$

, където $y_k \in \{0,1\}$ определя дали поредният обучителен пример принадлежи на клас *k* $(y_k = 1)$ или не $(y_k = 0)$.

3. За скрития слой *l=2* изчисляваме грешката

$$\delta^{(2)} = (\theta^{(2)})^T \delta^{(3)} .* g'(z^{(2)})$$

4. Изчисляваме градиента за поредния обучителен пример

$$\Delta^{(l)} = \Delta^{(l)} + \delta^{(l+1)} (a^{(l)})^T$$

5. Изчисляваме регуляризирания градиент

$$\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta) = D_{i,j}^{(l)} \coloneqq \frac{1}{m} \Delta_{i,j}^{(l)} \text{ при } j = 0$$

$$\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta) = D_{i,j}^{(l)} \coloneqq \frac{1}{m} \left(\Delta_{i,j}^{(l)} + \lambda \theta_{i,j}^{(l)} \right) \text{ при } j \geq 1$$

Задача 1. Довършете функцията sigmoidGradient(z).

Задача 2. Във функцията nnCostFunction довършете изчисляването на a1, z2, z3, a3 (feedforward propagation).

Задача 3. Във функцията nnCostFunction довършете изчисляването на d3, delta1, delta2, theta1_grad, theta2_grad (back propagation).

Изпълнете скрипта. Би трябвало да получите, че стойността на цената без регуляризация е около 0.287629165161, а с регуляризация - около 0.383769859091.

Задача 5. Изпълнете функцията sigmoidGradient последователно с параметри -1, -0.5, 0, 0.5, 1.

Резултат:

Сигмоид градиент

0.196611933241

0.235003712202

0.25

0.235003712202

0.196611933241