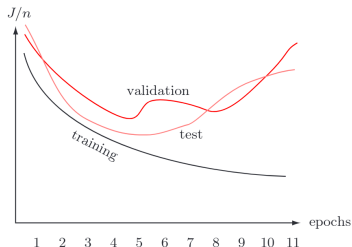


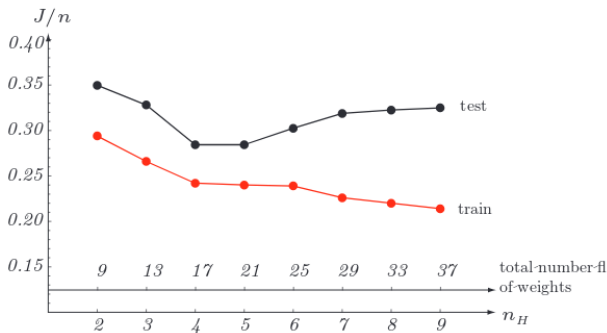
Generalizacja i regularyzacja sieci MLP

Generalizacja

- Na zbiorze treningowym błąd jest w stanie zawsze osiągnąć 0
- **Generalizacja** jest celem uczenia - uogólnienie reguły, która wytworzyła dane. Dążymy do minimalizacji błędu na danych, które nie zostały użyte w treningu.
- Zbiór **walidacyjny** - wyodrębniony fragment danych (np. ze zbioru uczącego) pozwala na ocenę błędu generalizacji - przydatny przy określaniu punktu zatrzymania treningu



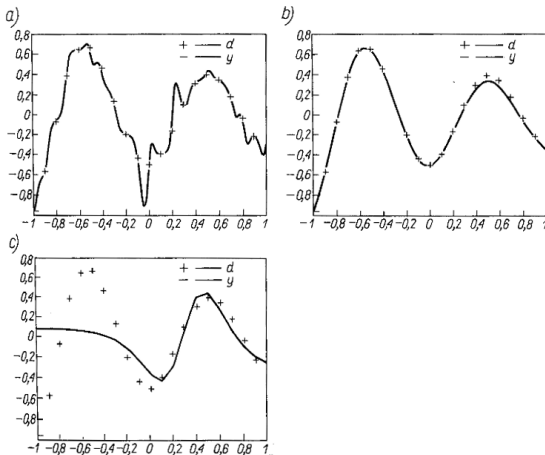
Rozmiar sieci



- ilość wag określa ilość stopni swobody - nie powinna przekraczać liczby przypadków uczących (np. $n/10$)
- za dużo wag - model uczy się na pamięć (przeuczenie)
- za mało wag - model zbyt prosty (niedouczony)

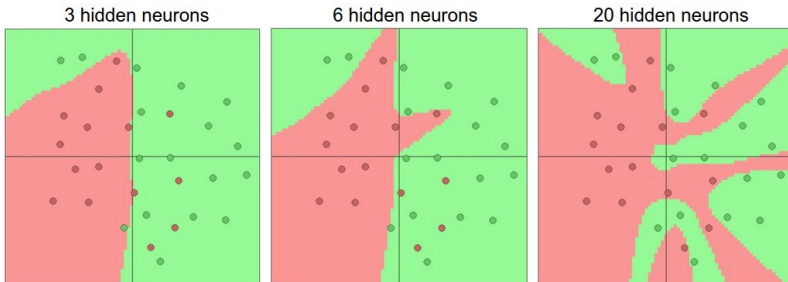
Rozmiar warstwy ukrytej

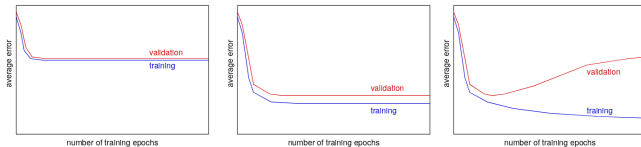
Zdolność uogólniania w problemie aproksymacji



Rozmiar warstwy ukrytej

Zdolność uogólniania w problemie klasyfikacji





- niedouczenie - błąd treningowy i walidacyjny pozostają duże
- przeuczenie - błąd walidacyjny rośnie a błąd treningowy maleje

Metody regularyzacji sieci neuronowych

Regularyzacja to metody, których celem jest poprawienie generalizacji

- dobór wielkości modelu (ilości neuronów i wag), preferowane są najprostsze rozwiązania (brzytwa Ockhama)
 - z pośród wielu modeli wybierz ten o najmniejszym błędzie walidacyjnym i najmniejszej liczbie parametrów (neuronów)
 - pruning - usuwanie nadmiarowych połączeń lub neuronów z wytrenowanej sieci
 - metody konstrukcyjne (rosnące) - zacznij od najprostszego modelu i zwiększaj jego złożoność w kolejnych iteracjach
 - sieci ontogeniczne - rozrastające się i kurczące się w czasie treningu
- małe wartości wag są preferowane
 - wagi zerowe oznaczają brak połączenia
 - neurony sigmoidalne z małymi wagami są w przybliżeniu liniowe

Regularyzacja

- modyfikacje funkcji kosztu wymuszające odpowiednią strukturę sieci, np. wymuszające minimalizację wartości wag

$$\bar{E}(\mathbf{w}) = E(\mathbf{w}) + \lambda \Omega(\mathbf{w})$$

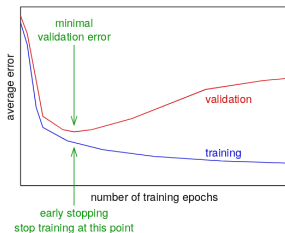
gdzie $\lambda > 0$ steruje wielkością regularyzacji

- wczesne zatrzymanie treningu, zanim błąd walidacyjny wzrośnie
- zwiększenie liczby danych lub przetransformowanie cech do „przyjemniejszej” postaci
- dodanie szumu do wag lub danych treningowych
- selekcja cech - wybór tylko istotnych cech do treningu
- uśrednianie wyników z wielu modeli, boosting, baging

Trening i walidacja

- uczenie sieci minimalizuje błąd treningowy a nie błąd generalizacji
- zbiór walidacyjny pozwala oszacować błąd generalizacji w trakcie treningu
- zbiór testowy - używany wyłącznie do ewaluacji nauczonego modelu
- kroswalidacja - metoda pozwalająca na oszacowanie błędu generalizacji i wariancji modelu poprzez powtarzanie treningu i ewaluacji na kolejnych podziałach zbioru uczącego
- k krotna kroswalidacja - dzieli zbiór na k części, gdzie $k - 1$ jest używanych do treningu a reszta do testu
- wyznaczamy średni błąd z k treningów
- kroswalidacja leave-one-out - gdy N różne liczbie wektorów uczących

Wczesne zatrzymanie



- przerwij trening, gdy błąd walidacyjny zacznie rosnąć, często wystarczy tylko kilka iteracji
- zapamiętaj model o najmniejszym błędzie walidacyjnym
- wymaga wydzielenia części zbioru treningowego, więc zmniejszenie liczby próbek uczących

Rys: Riedmiller, Machine Learning

Dobór liczby neuronów

Probabilistyczne oszacowanie błędu generalizacji sieci MLP klasyfikującej

$$E_G(\mathbf{w}) \leq E_T(\mathbf{w}) + \epsilon \left(\frac{N}{D}, E_T \right)$$

gdzie

N - liczba wzorców uczących,

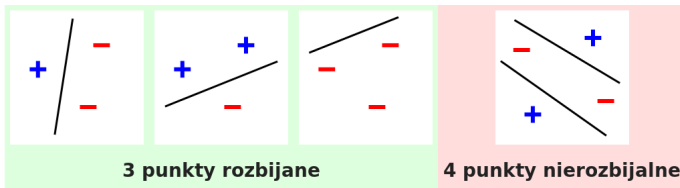
D - miara złożoności Vapnika-Chervonenkisa (VC dimension)

ϵ - przedział ufności maleje ze wzrostem $\frac{N}{D}$

Przeuczenie może zajść gdy $D > N$

Wymiar VC

Wymiar VC klasyfikatora to największa liczba punktów, którą klasyfikator jest w stanie rozbić, tj. podzielić przy dowolnej kombinacji etykiet (2^N możliwości)



Dla perceptronu $D = 3$

Wymiar VC dla MLP

Dla sieci MLP

$$2 \left\lceil \frac{N_h}{2} \right\rceil d \leq D \leq 2N_w(1 + \log N_n)$$

N_h - ilość neuronów ukrytych

d - ilość cech (wymiar przestrzeni wejściowej)

N_w - liczba wag

N_n - liczba wszystkich neuronów

- w praktyce $D \approx N_w$
- liczba próbek powinna być kilkakrotnie (np. 10 razy) większa od wymiaru VC

Regularyzacja L_2

Regularyzacja L_2 (weight decay, regularyzacja Tichonova) - czynnik regularyzacyjny dążący do zmniejszenia wartości wag

$$\hat{E}(\mathbf{w}) = E(\mathbf{w}) + \frac{1}{2}\lambda \sum_{ij} w_{ij}^2$$

gdzie $\lambda > 0$ decyduje o sile regularyzacji
(typowo $\lambda = 0.01$ lub 0.001)

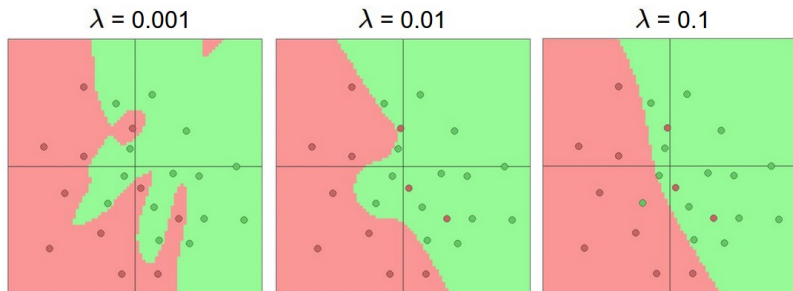
Zmniejszone są wszystkie wagi w trakcie uczenia

$$\hat{w}_{ij} = (1 - \lambda\eta)w_{ij}$$

Możliwe modyfikacje:

- wagi, które zmały poniżej pewnego progu mogą być usunięte
- usuwamy neurony dla których $\sum |w_i|$ jest bliskie zeru

Regularyzacja



Rys: <http://cs231n.github.io/neural-networks-1/>

Zmodyfikowany człon kary:

$$\hat{E}(\mathbf{w}) = E(\mathbf{w}) + \frac{1}{2}\lambda \sum_{ij} \frac{w_{ij}^2}{1 + \sum_k w_{ik}^2}$$

Minimalizacja E powoduje:

- zmniejszenie wartości wag w_{ij}
- eliminację neuronów dla których $\sum_k |w_{ik}|$ jest bliskie zera

Regularyzowana zmiana wag zależna jest od wartości w_{ij}

$$\hat{w}_{ij} = \left(1 - \lambda \eta \frac{1 + 2 \sum_{k \neq j} w_{ij}^2}{\left(1 + \sum_k w_{ik}^2 \right)^2} \right) w_{ij}$$

Dla neuronów z dużym $\sum_k |w_{ik}|$ czynnik regularyzacyjny zanika

Regularyzacja L_1

$$\hat{E}(\mathbf{w}) = E(\mathbf{w}) + \lambda \sum_{ij} |w_{ij}|$$

$$\nabla \hat{E}(\mathbf{w}) = \nabla E(\mathbf{w}) + \lambda \operatorname{sgn} \mathbf{w}$$

regularyzacja wpływa na gradient ze stałą wartością (zależy tylko od znaku w_{ij})

$$\hat{w}_{ij} = \begin{cases} w_{ij} - \lambda \eta & \text{dla } w_{ij} > 0 \\ w_{ij} + \lambda \eta & \text{dla } w_{ij} < 0 \end{cases}$$

Regularyzacja L_1 daje rzadsze rozwiązanie od L_2

Metody wrażliwościowe redukcji sieci

- Eliminacja wag na podstawie jej wpływu na wartość błędu
- Wagi o małej wartości $|w_i|$ są mniej istotne od dużych wag
- Wrażliwość połączenia

$$S_{ij} = E - E(w_{ij} = 0)$$

- Połączenia o najmniejszej wrażliwości usuwa się po czym sieć jest douczana
- Usuwamy neuron dla którego wszystkie dochodzące (lub wychodzące) połączenia są wyzerowane

Optimal Brain Damage (LeCun)

Założenie: hesjan $\mathbf{H}$ jest diagonalnie dominujący, więc uwzględniamy tylko składową diagonalną.

Z rozwinięcia funkcji błędu E w szereg Taylora i przyjmując $\nabla E = 0$ (punkt zbieżności dla wytrenowanej sieci)

$$\Delta E = \frac{1}{2} \Delta \mathbf{w}^T \mathbf{H} \Delta \mathbf{w} = \frac{1}{2} \sum_i H_{ii} \Delta w_i^2$$

Miara ważności wagi

$$S_i = \frac{1}{2} \frac{\partial^2 E}{\partial^2 w_i} w_i^2$$

Wagi o najmniejszym S_i mogą być usunięte z wytrenowanej sieci.

Optimal brain Damage

1. Wytrenuj sieć dowolnym algorytmem uczenia
2. Wyznacz diagonalne elementy hesjanu oraz wartości wrażliwości S_i
3. Posortuj wagi zgodnie z rosnącymi wartościami S_i i obetnij te o najmniejszych wartościach.
4. Wróć do punktu 1

Redukcja neuronów o małej aktywności

- Neurony o niewielkiej aktywności (których sygnał wyjściowy nie zmienia się dla zbioru treningowego) można usunąć bez szkody dla generalizacji
- duża aktywność neuronu świadczy o jego przydatności
- modyfikacja funkcji kosztu z czynnikiem kary za zbyt małą aktywność neuronów (Y. Chaurin)

$$\hat{E} = E + \lambda \sum_{i=1}^k \sum_{j=1}^n e(\Delta_{ij}^2)$$

gdzie Δ_{ij} oznacza zmianę sygnału wyjściowego i -tego neuronu dla j -tego wektora treningowego

- przykładowy czynnik korelacyjny

$$e = \frac{1}{1 + \Delta_{ij}^2}$$

Architektury konstruktywne (rozzrastające się)

- automatycznie dobierają złożoność modelu do złożoności problemu, rozrastając się od prostych do złożonych modeli
- strategia zachłanna - dodanie neuronu maksymalizuje zysk w pojedynczym kroku rozbudowy
- przedwczesne przerwanie uczenia nie musi być katastrofą
- (zazwyczaj) niski koszt obliczeniowy, np. w każdym cyklu douczamy tylko dodatkowy neuron, reszta połączeń jest „zamrożona”
- możliwość budowania sieci o różnorodnych funkcjach aktywacji
- przykład: algorytm wieżowy i piramidalny, korelacja kaskadowa

Korelacja kaskadowa (Fahlman i Labiere, 1989)

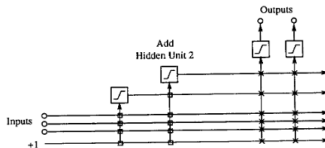
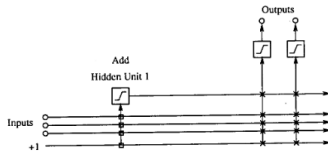
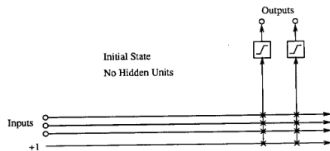
Sieć rozrastająca się:

- zaczynamy od treningu sieci bez warstwy ukrytej
- w kolejnych krokach dodawany jest neuron do warstwy ukrytej, wejścia neuronu połączone są ze wszystkimi wejściami sieci i wyjściami poprzednio dodanych neuronów ukrytych
- wagi neuronu dobierane są w procesie maksymalizacji korelacji nowego neuronu k z błędem wykazywanym przez neurony wyjściowe

$$C = \sum_{j=1}^M \left| \sum_{i=1}^N (o(\mathbf{x}_i) - \bar{o}) ((y_{ij} - f_j(\mathbf{x}_i)) - (\bar{y}_{ij} - \bar{f}_j(\mathbf{x}_i))) \right|$$

gdzie N - liczba wzorców, M - liczba wyjść sieci

Korelacja kaskadowa



Kaskadowa korelacja

- wagi kandydata maksymalizujące korelację są „zamrażane” – douczaniu podlegają wyłącznie wagi wyjściowe sieci
- jeśli wyjścia neuronu kandydata są skorelowane dodatnio z błędem to w wyniku treningu wykształci wagi, które mają szansę zniwelować ten błąd
- w każdym kroku rozpatruje się zbiór kandydatów (od 5 do 10), mogą posiadać różne funkcje aktywacji (sigmoidalna, gaussowska, itp.),
- kandydaci trenowani są równolegle konkurując ze sobą, wybiera się najlepszego (o największej korelacji)

Algorytmy rozrastające się - podsumowanie

- Problem z przeuczeniem, kiedy zaprzestać rozrost?
- Za dużą sieć można przyciąć ograniczając wpływ przeuczenia
- Uczenie zachłanne - dokładany pojedynczy neuron, koszt obliczeń zależy od czasu uczenia nowego neuronu, złożone struktury danych mogą nie zostać wykryte przez dodanie pojedynczego neuronu, zachłanna strategia nie zawsze jest najlepsza
- Algorytmy rosnące nie gwarantują najprostszyc sieci
- Niektóre tworzą specyficzne architektury
- Sieci ontogeniczne - rosną i kurczą się w trakcie treningu
 - wyszukiwanie najlepszej architektury, stosuje się np. alg. genetyczne, ewolucyjne
 - zazwyczaj wymagające obliczeniowe

Rozmiar zbioru treningowego

- większa liczba danych zmniejsza ryzyko przeuczenia
- „There's is no data like more data”
- dodanie szumu do danych uodparnia sieć na drobne zmiany w przestrzeni wejściowej
- sztuczne zwielokrotnianie danych (*data augmentation*), często stosowane z dobrymi rezultatami
 - w klasyfikacji obrazów: zmiany kontrastów i nasycenia barw, przesunięcia, obroty, itp.
 - w analizie mowy: pogłos, szum otoczenia, zmiana głośności, przyspieszenie i zwolnienie mowy, itp..

Selekcja i ekstrakcja cech

- zmienne, które nie zawierają istotnych informacji mogą wpływać negatywnie na trening
- metody selekcji cech usuwają nieistotne zmienne zmniejszając tym samym rozmiar zbioru treningowego
- dodanie cech, które niosą wartościową informację poprawia generalizację
- istnieje wiele metod, najczęściej dobierane są do specyfiki problemu
- niezbalansowane klasy - częsty problem w klasyfikacji