DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Regularización

Fernando Berzal, berzal@acm.org

Regularización

- El problema del sobreaprendizaje
- Técnicas de regularización
  - Weight decay (regularización L1/L2)
  - Restricciones sobre los pesos
  - Introducción de ruido
  - Early stopping
  - Dropout
- Ajuste de hiperparámetros

En la práctica…

Algoritmo de aprendizaje de redes multicapa

Aspectos que debemos considerar en su diseño:

- **Parámetros:** ¿Qué topología de red utilizamos?...
- **Optimización:** ¿Cómo obtenemos los pesos?
- **Generalización:** ¿Cómo conseguimos que la red funcione bien con datos distintos a los del conjunto de entrenamiento?
- **Invarianza:** ¿Cómo conseguimos que la red sea robusta frente a transformaciones comunes en los datos?

DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

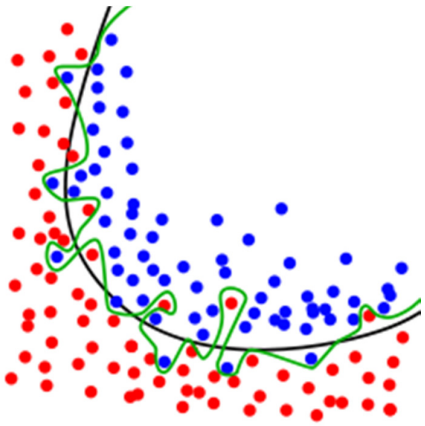
Regularización

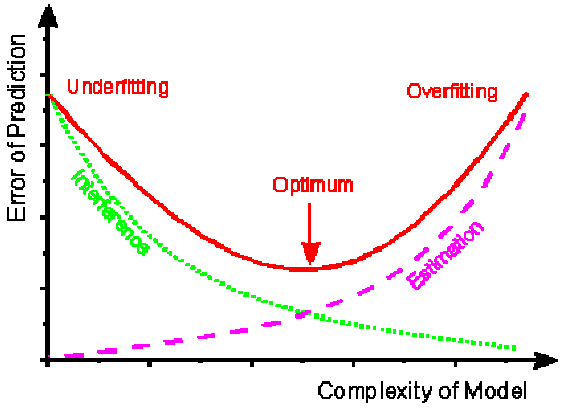
El problema del sobreaprendizaje

El problema del sobreaprendizaje

- El conjunto de entrenamiento contiene patrones útiles, pero también ruido: regularidades accidentales debidas al conjunto de datos particular utilizado (error de muestreo).
- Cuando ajustamos la red a los datos del conjunto de entrenamiento, no podemos diferenciar las regularidades reales de las debidas al conjunto de entrenamiento utilizado, por lo que corremos el riesgo de "sobreaprender" [overfitting].

El problema del sobreaprendizaje

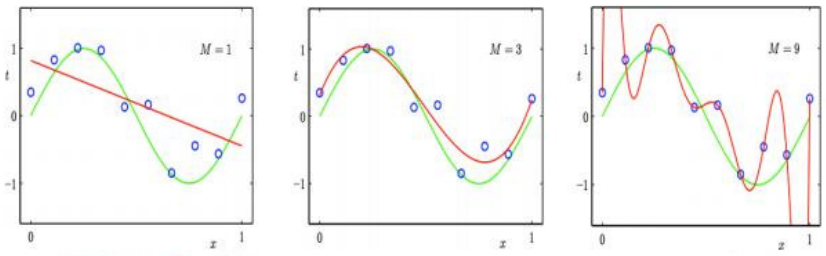




Sobreaprendizaje [overfitting]

El problema del sobreaprendizaje

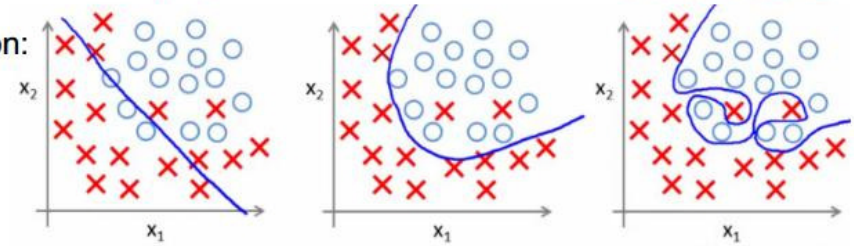
Regression:



predictor too inflexible:
cannot capture pattern

predictor too flexible:
fits noise in the data

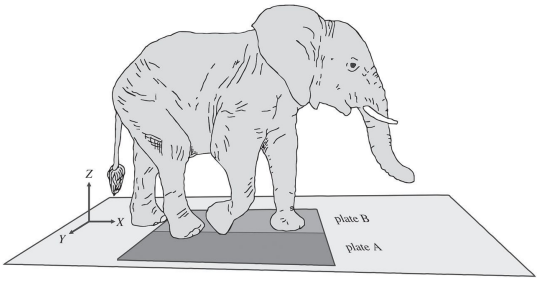
Classification:

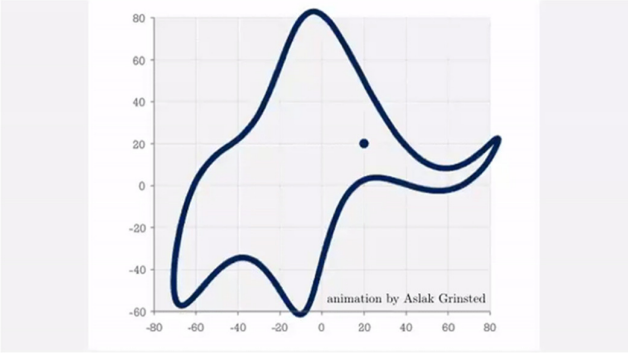


Underfitting & overfitting

El problema del sobreaprendizaje

"Con cuatro parámetros puedo ajustar un elefante,
con cinco puedo hacer que menee su trompa."
-- John von Neumann





Jürgen Mayer, Khaled Khairy, y Jonathan Howard.
Drawing an elephant with four complex parameters.
American Journal of Physics, 78(6):648–649, 2010.
DOI 10.1119/1.3254017

DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Regularización
Técnicas de regularización

Técnicas de regularización

Estrategias para evitar el sobreaprendizaje:

- Obtener más datos (la mejor opción si tenemos capacidad para entrenar la red usando más datos).
- Ajustar los parámetros de la red para que tenga la capacidad adecuada (suficiente para identificar las regularidades en los datos, pero no demasiada para ajustarse a las espúreas, suponiendo que sean más débiles que las auténticas).

Técnicas de regularización

Una tercera estrategia: Combinar modelos

- **"Model averaging"** (a.k.a. "ensembles"):
  Muchos modelos diferentes con distintos parámetros o el mismo tipo de modelo utilizando distintos subconjuntos del conjunto de entrenamiento [bagging].

- **"Bayesian fitting"** (enfoque bayesiano):
  Utilizando una única arquitectura de red, se combinan las predicciones realizadas por muchos vectores de pesos diferentes.

Técnicas de regularización

Capacidad de la red: Topología

Algunas formas de limitar la capacidad de la red actuando sobre su topología:

- **Arquitectura de la red**:
  Se limita el número de capas ocultas y/o el número de unidades por capa.

- **Weight sharing**:
  Se reduce el número de parámetros de la red haciendo que distintas neuronas compartan los mismos pesos (p.ej. redes convolutivas).

Técnicas de regularización

Capacidad de la red: Entrenamiento

Algunas formas de limitar la capacidad de la red actuando sobre su algoritmo de entrenamiento:

- **Early stopping**: Se comienza a entrenar la red con pesos pequeños y se para el entrenamiento antes de que sobreaprenda.
- **Weight decay**: Se penalizan los pesos grandes en función de sus valores al cuadrado (penalización L2) o absolutos (penalización L1).
- **Ruido**: Se añade ruido a los pesos o actividades de las neuronas de la red que se está entrenando.

Técnicas de regularización

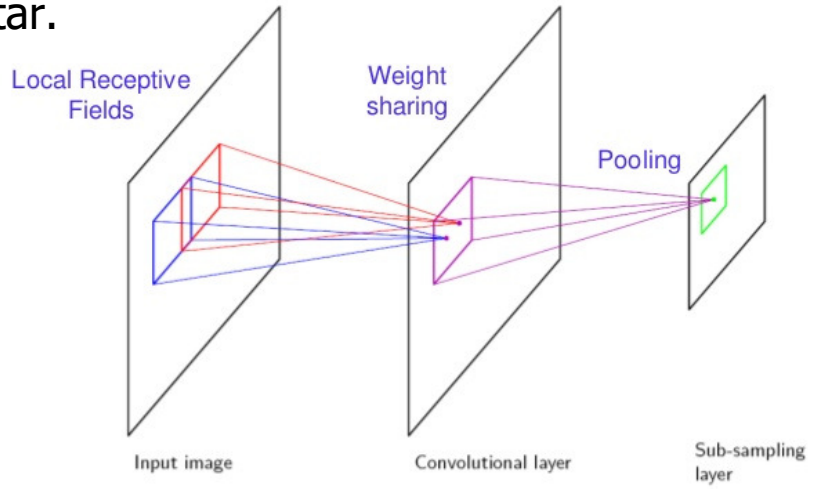
En la práctica, se pueden combinar varias técnicas concretas para prevenir el sobreaprendizaje:

- Weight sharing
- Weight decay
- Ruido (sobre pesos, entradas o actividades)
- Early stopping
- Dropout
- Generative pre-training → Deep Learning

Técnicas de regularización

Weight sharing

Se usan los mismos pesos para diferentes neuronas, reduciendo así el número de parámetros que hay que ajustar.



Ejemplo: Redes convolutivas

DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Regularización

Regularización L1 & L2 (a.k.a. weight decay)

Regularización L1 & L2

Weight decay

Incorporación de una penalización en la función de coste (como la regularización en las técnicas de regresión)

$$E = \frac{1}{2}\sum_i \| t_i - y_i \|^2 = \frac{1}{2}\sum_i \sum_j (t_{ij} - y_{ij})^2$$

$$J = \frac{1}{2m}\sum_i \| t_i - y_i \|^2 + \frac{\lambda}{2}\sum_l \sum_i \sum_j (w_{ij}^{(l)})^2$$

donde m es el número de ejemplos del conjunto de entrenamiento y λ un factor de regularización.

Regularización L1 & L2

Weight decay

La penalización mantiene los pesos pequeños, evitando que la red utilice pesos que no necesite realmente.

$$J = E + \frac{\lambda}{2}\sum_i w_i^2$$

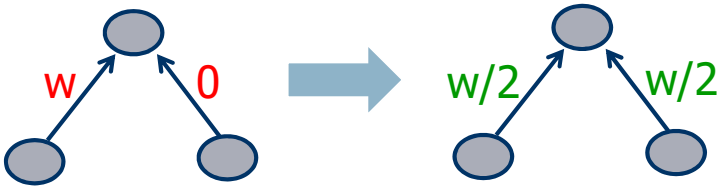
$$\frac{\partial J}{\partial w_i} = \frac{\partial E}{\partial w_i} + \lambda w_i$$

$$cuando \quad \frac{\partial J}{\partial w_i} = 0, \quad w_i = -\frac{1}{\lambda}\frac{\partial E}{\partial w_i}$$

Regularización L1 & L2

Weight decay

- Se reduce el sobreaprendizaje evitando que la red se ajuste demasiado al conjunto de entrenamiento.
- Se consigue un modelo más "suave" en el que las salidas cambian más lentamente cuando cambian las entradas.

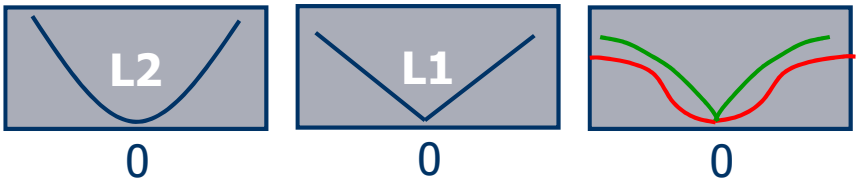


Si la red tiene dos entradas similares, preferirá poner la mitad del peso en cada una, en vez de todo en una sola.

Regularización L1 & L2

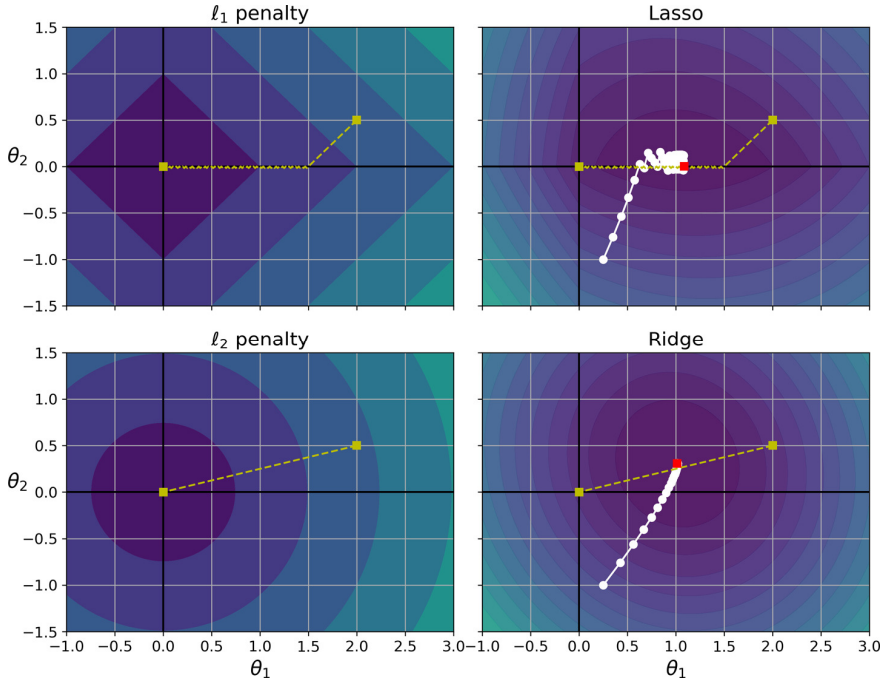
Weight decay: Tipos de penalización

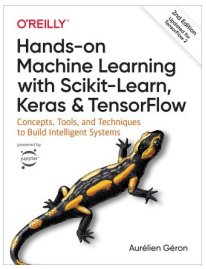
- La penalización estándar añade un término que incluye los pesos al cuadrado (regularización L2).
- En ocasiones, funciona mejor penalizar usando el valor absoluto de los pesos (regularización L1), con lo que muchos pesos acaban siendo exactamente 0.
- En otras ocasiones, puede ser recomendable utilizar una penalización que no tenga apenas efecto sobre pesos grandes (para permitir algunos pesos grandes en la red).



Regularización L1 & L2

Weight decay: Tipos de penalización





Regularización L1 & L2

Weight decay: Tipos de penalización

- **L2** (a.k.a. ridge regression / Tikhonov regularization)

$$J = Error + \frac{\lambda}{2}\sum_k\sum_i\sum_j \left(w_{ij}{}^{(k)}\right)^2$$

- **L1** (a.k.a. Lasso regression)

$$J = Error + \sum_k\sum_i\sum_j \left|w_{ij}{}^{(k)}\right|$$

- **L1 + L2** (a.k.a. elastic net)

$$J = Error + r\lambda\sum_k\sum_i\sum_j \left|w_{ij}{}^{(k)}\right| + (1-r)\frac{\lambda}{2}\sum_k\sum_i\sum_j \left(w_{ij}{}^{(k)}\right)^2$$

Regularización
Restricciones sobre los pesos

Restricciones sobre los pesos

En vez de incluir una penalización sobre los pesos de la red en la función de coste, podemos establecer directamente restricciones sobre los pesos.

p.ej. Establecemos un máximo sobre la magnitud del vector de pesos de entrada de cada neurona (si una actualización viola esta restricción, el vector de pesos se reescala).

a.k.a. **max-norm regularization**

$$\| \mathbf{w} \|_2 \leq r$$

Restricciones sobre los pesos

Ventajas de las restricciones sobre las penalizaciones:

- Obtienen valores razonables para los pesos.
- Evitan que la actividad de las neuronas ocultas se quede atascada en torno a 0.
- Evitan que los pesos "exploten".

Suelen ser más efectivas que una penalización fija a la hora de conseguir que los pesos irrelevantes tiendan hacia cero.

DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Regularización
Introducción de ruido

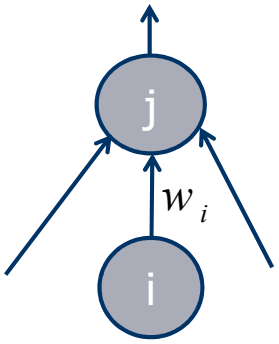
Introducción de ruido

Ruido sobre las entradas

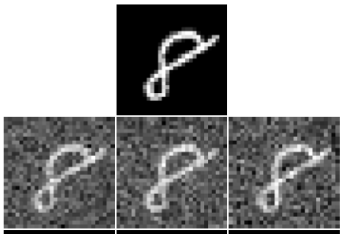
Regularización L2 añadiendo ruido a las entradas

Si añadimos ruido gaussiano a las entradas de una neurona lineal, la varianza del ruido se amplifica antes de llegar a la siguiente capa:

$$y_j + N(0, w_i^2 \sigma_i^2)$$



$$x_i + N(0, \sigma_i^2)$$



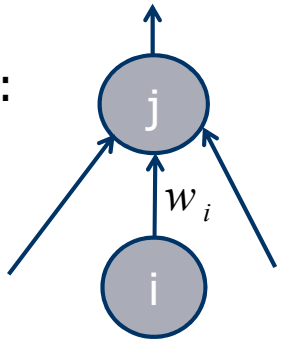
Introducción de ruido

Ruido sobre las entradas

Regularización L2 añadiendo ruido a las entradas

El ruido amplificado se añade a la salida (realiza una contribución al error):

$$y_j + N(0, w_i^2 \sigma_i^2)$$



Minimizar el error cuando se introduce ruido en las entradas regulariza los pesos.

$$x_i + N(0, \sigma_i^2)$$

Introducción de ruido

Ruido sobre las entradas

Regularización L2 añadiendo ruido a las entradas

- Salida: $y^{noisy} = \sum_i w_i x_i + \sum_i w_i \varepsilon_i \;\; donde \; \varepsilon_i \; se \; muestrea \; de \; N(0, \sigma_i^2)$
- Error:

$$E\left[(y^{noisy} - t)^2\right] = E\left[\left(y + \sum_i w_i \varepsilon_i - t\right)^2\right] = E\left[\left((y-t) + \sum_i w_i \varepsilon_i\right)^2\right]$$

$$= (y-t)^2 + E\left[2(y-t)\sum_i w_i \varepsilon_i\right] + E\left[\left(\sum_i w_i \varepsilon_i\right)^2\right]$$

El error gaussiano en las entradas equivale a una penalización L2

$$= (y-t)^2 + E\left[\sum_i w_i^2 \varepsilon_i^2\right]$$

$$= (y-t)^2 + \sum_i w_i^2 \sigma_i^2$$

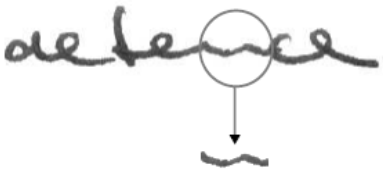
Introducción de ruido

Ruido sobre los pesos

Añadir ruido a los pesos de una red neuronal multicapa no lineal también puede ayudar (aunque no sea exactamente equivalente a una penalización L2 sobre los pesos).

EJEMPLO

Redes recurrentes para el reconocimiento de textos manuscritos



Alex Graves:
Supervised Sequence Labelling with Recurrent Neural Networks
Springer, 2012. ISBN 978-3-642-24797-2

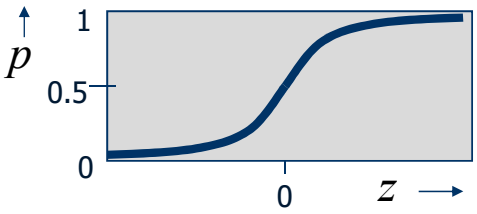
Introducción de ruido

Ruido sobre las actividades

$$p = \frac{1}{1+e^{-z}}$$

Otra alternativa: Se usa backpropagation para entrenar una red multicapa con unidades logísticas,

- como si las neuronas fuesen unidades binarias estocásticas en el paso hacia adelante,
- pero se comportan de la forma normal en la propagación de errores hacia atrás



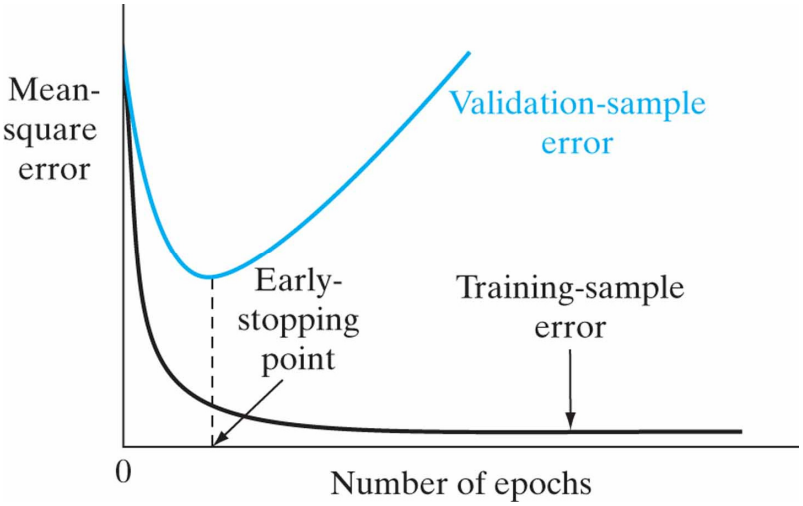
Entrenamiento más lento, con peores resultados sobre el conjunto de entrenamiento pero mejores resultados sobre el conjunto de prueba :-)

DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Regularización

Parada temprana [early stopping]

Parada temprana



Early stopping

[Haykin: "Neural Networks and Learning Machines", 3rd edition]

Parada temprana

Early stopping

- Puede resultar demasiado costoso probar con diferentes penalizaciones sobre los pesos de la red, por lo que puede ser aconsejable empezar el entrenamiento con pesos muy pequeños y dejar que crezcan hasta que el rendimiento sobre el conjunto de validación comience a empeorar.

- Se limita la capacidad de la red al impedir que los pesos crezcan demasiado.

Parada temprana

Early stopping

¿Por qué funciona?

- Cuando los pesos son muy pequeños, las neuronas de las capas ocultas se mantienen en su rango lineal: Una red con capas ocultas de neuronas lineales no tiene más capacidad que una red lineal sin capas ocultas.

- Conforme crecen los pesos, las neuronas ocultas comienzan a comportarse de forma no lineal (y la capacidad de la red aumenta).

DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

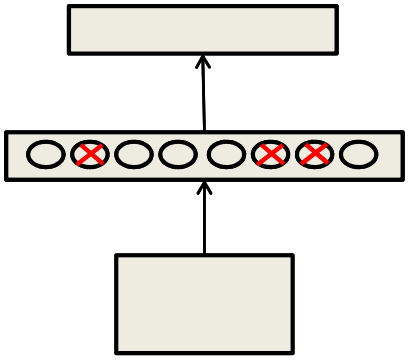
Regularización

Dropout

Dropout

Dropout

Técnica de regularización



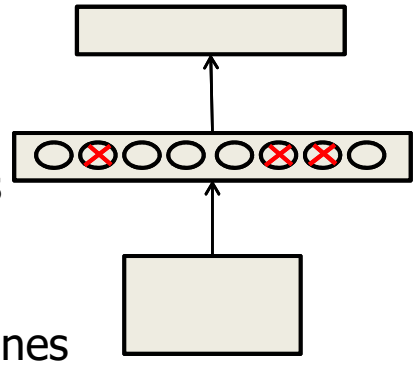
- Para cada caso de entrenamiento, se omite aleatoriamente cada neurona oculta con probabilidad 0.5 (la mitad de las neuronas de las capas ocultas no se utilizan para cada caso del conjunto de entrenamiento).

- Se evita que las neuronas ocultas dependan/confíen demasiado en el trabajo de otras neuronas ocultas de su misma capa.

Dropout

Dropout

Técnica de regularización



- Si una neurona oculta sabe qué otras neuronas ocultas existen, pueden co-adaptarse sobre el conjunto de entrenamiento, pero las co-adaptaciones no funcionarán bien sobre el conjunto de prueba: ¡las conspiraciones complejas no son robustas! [Hinton].

- Si tiene que trabajar correctamente con muchos conjuntos de neuronas, es más probable que haga algo que sea individualmente útil.

Dropout

Dropout & model averaging

Una forma de combinar múltiples modelos

- Para cada caso de entrenamiento, estamos muestreando de una familia de 2^H arquitecturas diferentes (redes que comparten los mismos pesos).
- Sólo se entrenan algunos de los posibles modelos y cada modelo sólo recibe un caso de entrenamiento (versión extrema de bagging).
- Al compartir pesos, se regularizan todos los modelos (mucho mejor que las penalizaciones L2 o L1).

Dropout

Dropout

- A la hora de utilizar la red, podríamos muestrear múltiples modelos y calcular la media geométrica de sus salidas…

Modelo A:	.3	.2	.5
Modelo B:	.1	.8	.1
Combinado	$\sqrt{.03}$	$\sqrt{.16}$	$\sqrt{.05}$ /sum

- … pero es mejor utilizar todas las neuronas ocultas, dejando en la mitad sus pesos de salida (w/2 equivale a calcular la media geométrica de las predicciones de los 2^H modelos posibles).

Dropout

Dropout

¿Y si tenemos varias capas ocultas?

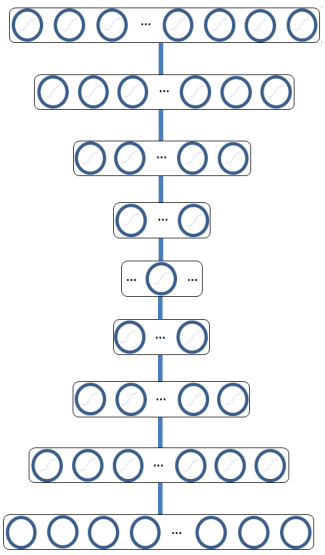
- Usamos la red "promedio" con todos los pesos de salida de las neuronas ocultas divididos por 2. Matemáticamente, no es equivalente a promediar todos los modelos posibles, pero es una buena aproximación (y muy rápida).
- La alternativa es aplicar de forma estocástica el modelo varias veces con la misma entrada (lo que nos permite hacernos una idea de la incertidumbre de la respuesta proporcionada por la red neuronal).

Dropout

Dropout

¿Algo que decir sobre la capa de entrada?

Se puede utilizar dropout sobre la capa de entrada, siempre que usemos una probabilidad mayor para mantener las unidades de entrada.



NOTA:

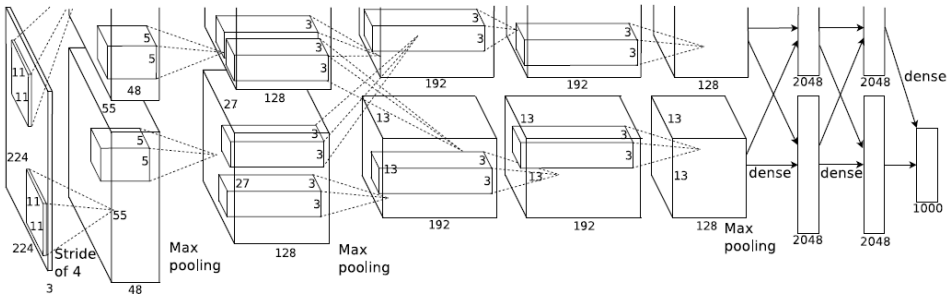
Los "denoising autoencoders" de Bengio et al. utilizan este truco ➔ Deep Learning

Dropout

Dropout

En deep learning, dropout reduce significativamente el error, evitando el sobreaprendizaje.

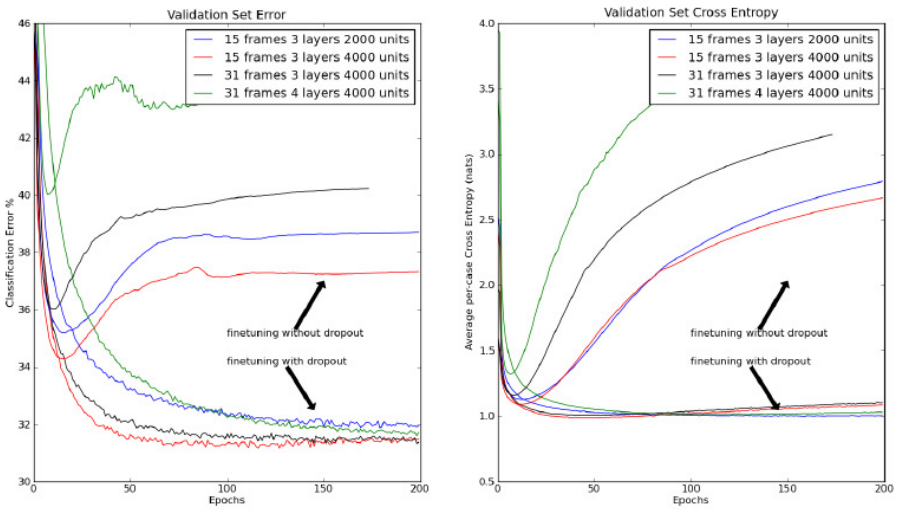
IMAGENET Alex Krizhevsky (NIPS 2012)



[Hinton] En deep learning, si tu red no sobreaprende, entonces es que deberías usar una red más grande.

Dropout

Dropout

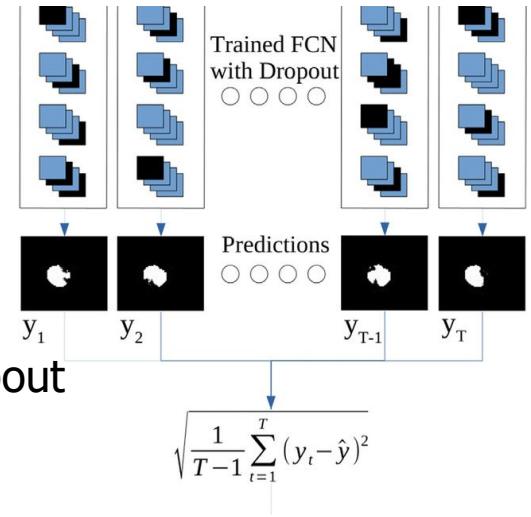


Cualquier red que se entrene con "early stopping" puede hacerlo mejor si se utiliza dropout, aunque costará más tiempo entrenarla.

Dropout

Monte Carlo Dropout [MC Dropout]

Promediar predicciones
usando dropout (p.ej. 100)
nos permite obtener
una estimación que suele ser
más fiable que
el resultado de una predicción
en la que hayamos eliminado dropout
(y estimar la incertidumbre
de nuestra predicción).



Yarin Gal and Zoubin Ghahramani,
"Dropout as a Bayesian Approximation:
Representing Model Uncertainty in Deep Learning,"
*Proceedings of the 33rd International Conference on Machine Learning ICML'*2016: 1050–1059

DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Regularización
Ajuste de hiperparámetros

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

Una de las principales dificultades prácticas del uso de redes neuronales es la destreza que requiere establecer todos sus parámetros ("arte" más que ciencia)

p.ej. Número de capas
Número de neuronas por capa
Tipo de neuronas
Penalización de los pesos
Tasas de aprendizaje
…

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

¿Cómo elegir los hiperparámetros de una red neuronal?

Método incorrecto: Se prueban montones de alternativas para ver cuál funciona mejor en el conjunto de test.

- Fácil de hacer, pero nos da una impresión engañosa de lo bien que funcionará la red en la práctica:
- La configuración que funcione mejor sobre el conjunto de prueba puede que no sea la que funcione mejor en otros conjuntos de prueba (o los nuevos casos sobre los que queramos aplicar la red neuronal).

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

¿Cómo elegir los hiperparámetros de una red neuronal?

Un método mejor: **Conjunto de validación.**

Se divide el conjunto de datos disponible en tres partes:

- Conjunto de **entrenamiento** (para aprender los parámetros del modelo, i.e. los pesos de la red).
- Conjunto de **validación** (no se utiliza en el entrenamiento, sino para decidir qué hiper-parámetros resultan más adecuados)
- Conjunto de **prueba** (para obtener una estimación no sesgada de lo bien que funciona la red).

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

¿Cómo elegir los hiperparámetros de una red neuronal?

Validación cruzada

- Dividimos el conjunto de datos en N subconjuntos.
- Utilizamos N-1 subconjuntos de conjunto de entrenamiento y el subconjunto restante de conjunto de prueba para obtener N estimaciones del error.

Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

¿Cómo elegir los hiperparámetros de una red neuronal?

Aprendizaje automático [Machine Learning]

En vez de probar todas las combinaciones posibles de parámetros, podemos muestrear el espacio de posibles combinaciones.

p.ej. Metaheurísticas (algoritmos genéticos)

Optimización bayesiana (procesos gaussianos)

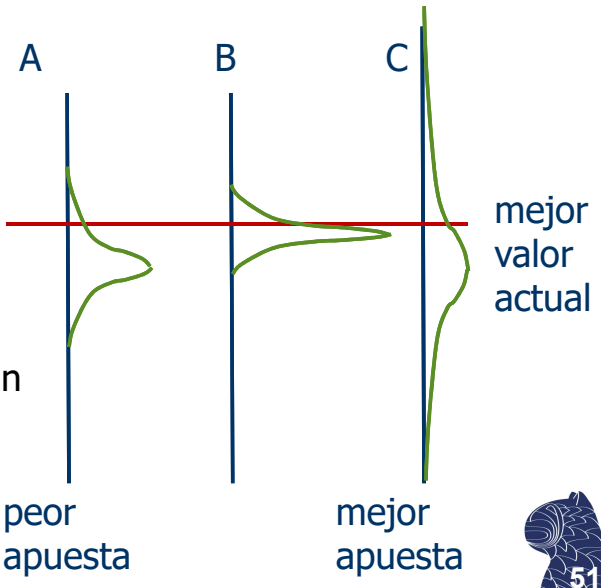
Ajuste de hiperparámetros

Hiperparámetros (o meta-parámetros)

¿Cómo elegir los hiperparámetros de una red neuronal?

Modelo de procesos gaussianos:
A partir de la mejor configuración conocida, se elige una combinación de hiperparámetros tal que la mejora esperada sea grande (sin preocuparse por la posibilidad de empeorar).

[Snoek, Larochelle & Adams, NIPS 2012]



Ajuste de hiperparámetros

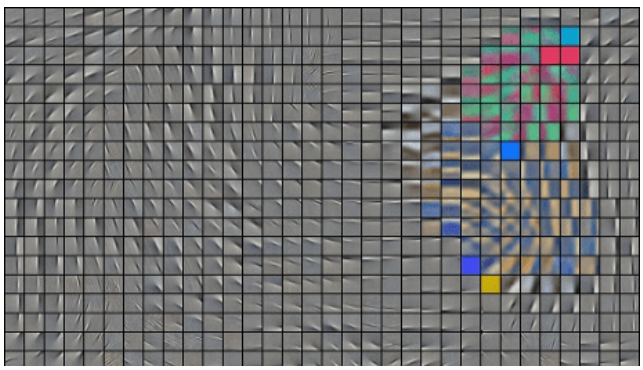
Hiperparámetros (o meta-parámetros)
¿Cómo elegir los hiperparámetros de una red neuronal?

Aprendizaje automático [Machine Learning]

- Mucho mejor que ir haciendo pruebas manualmente (no es el tipo de tarea que los humanos hacemos bien).
- Evita sesgos psicológicos no deseados: método menos propenso a funcionar mejor con el método que nos gusta y peor con el que no (las personas no podemos evitarlo ;-)

Cursos

Neural Networks for Machine Learning
by Geoffrey Hinton
(University of Toronto & Google)
https://www.coursera.org/course/neuralnets



UNIVERSITY OF TORONTO

Cursos

Deep Learning Specialization

by Andrew Ng, 2017

- Neural Networks and Deep Learning
- Improving Deep Neural Networks: Hyperparameter tuning, Regularization and Optimization
- Structuring Machine Learning Projects
- Convolutional Neural Networks
- Sequence Models

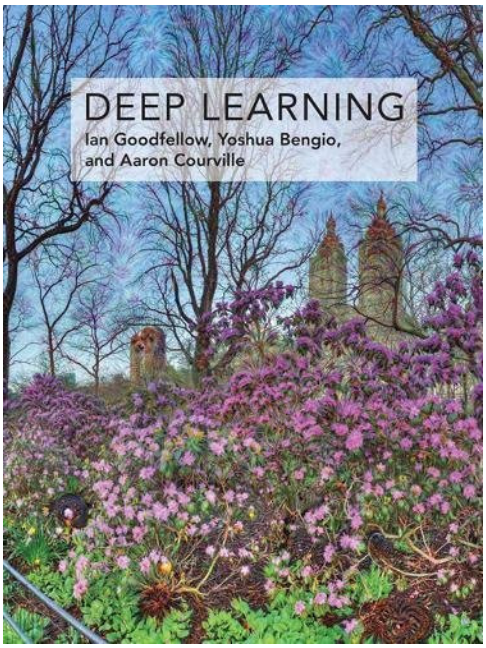
deeplearning.ai

https://www.coursera.org/specializations/deep-learning

Bibliografía

Lecturas recomendadas

Ian Goodfellow,
Yoshua Bengio
& Aaron Courville:
Deep Learning
MIT Press, 2016
ISBN 0262035618



http://www.deeplearningbook.org

Bibliografía

Lecturas recomendadas

Fernando Berzal:
Redes Neuronales & Deep Learning

CAPÍTULO 10
Prevención del sobreaprendizaje