Diseño Óptimo de Comités de Redes Neuronales Artificiales

Pedro J. García Laencina*, José Luis Sancho Gómez.
Departamento de Tecnologías de la Información y las Comunicaciones
Universidad Politécnica de Cartagena. Plaza del Hospital 1, 30202 Cartagena
Teléfono: (+34) 968326542. E-mail: pedroj.garcia@upct.es

Resumen. *Las máquinas de aprendizaje, y particularmente, las redes neuronales artificiales (RNA), tienen aplicación en multitud de problemas reales: control automático, detección de señales, estimación de variables financieras, filtros 'antispam', etc. Una manera eficiente para mejorar la capacidad de generalización de una RNA es diseñar un conjunto de máquinas ('committee of machines' o 'network ensembles'), cuya solución global es el resultado de combinar la estimación proporcionada por cada máquina. Este artículo propone un novedoso, rápido y eficiente método para el entrenamiento de comités de máquinas basado en el algoritmo 'Extreme Learning Machine'.*

1. Introducción

Las Redes Neuronales Artificiales (RNAs) han sido empleadas con éxito en multitud de aplicaciones [1]. Sin embargo, presentan diversos inconvenientes durante el proceso de optimización de los hiper-parámetros (i.e., pesos y número de neuronas) como son el elevado tiempo computacional requerido y la convergencia a soluciones sub-optimas (mínimos locales) para un determinado problema al usar técnicas de optimización basados en gradiente [1]. Recientemente, el algoritmo de entrenamiento conocido como *Extreme Learning Machine* [2], [3] ha permitido solventar estos claros inconvenientes que presentan las técnicas tradicionales de entrenamiento de RNA. Destaca el trabajo dirigido por Lendasse [3], donde se propone *Optimal Prunned-Extreme Learning Machine* (OP-ELM), que constituye un eficiente método para la selección óptima del número de neuronas en RNAs basadas en ELM.

Sin embargo, en lugar de emplear una única máquina para modelar un determinado problema, un procedimiento muy recomendable es modelar dicho problema mediante un conjunto de máquinas, cuya solución viene dada por la combinación de la salida obtenida por cada una de las máquinas [4]. Siguiendo esta filosofía, este trabajo extiende el algoritmo OP-ELM para obtener el conjunto óptimo de RNAs, proponiendo una nueva técnica para el entrenamiento de comités de RNAs: *Optimal Committee of Extreme Learning Machines*, OCoELM. Los resultados obtenidos en problemas de regresión muestran las ventajas de las máquinas OP-ELM y el algoritmo propuesto OCoELM.

2. Perceptrones Multicapa

Las RNAs del tipo "feed-forward", o Perceptrones Multicapa (Multi-Layer Perceptron, MLP), constan de múltiples capas de unidades computacionales (neuronas) interconectadas, pudiendo aproximar cualquier función continua seleccionando los hiper-parámetros (pesos y número de neuronas) adecuadamente [1]. Una arquitectura MLP estándar está compuesta de una capa oculta de h neuronas. Los pesos de la capa de entrada conectan las n variables de entrada con las h neuronas, mientras que una segunda capa de pesos conecta las salidas de las h neuronas con las m unidades de salida del MLP. Considerar un vector de entrada $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}^d$, con "target" $\mathbf{t} = [t_1, t_2, \ldots, t_m]^T \in \mathbb{R}^m$. De esta forma, la salida del MLP viene dada por

$$\mathbf{y} = \sum_{j=1}^{h} \mathbf{v}_j z_j = \sum_{j=1}^{h} \mathbf{v}_j f\left(\mathbf{w}_j^T \mathbf{x} + b_j\right), \qquad (1)$$

donde $\mathbf{y} = \{y_k\}_{k-1}^m$, $f(\cdot)$ son las funciones de activación de las neuronas ocultas, $\mathbf{v}_j = [v_{j1}, v_{j2}, \ldots, v_{jm}]^T$ son los pesos de salida asociados a la neurona oculta j-ésima, mientras que $\mathbf{w}_j = [w_{j1}, w_{j2}, \ldots, w_{jn}]^T$ y b_j representan respectivamente el vector de pesos de entrada y el sesgo correspondientes a dicha neurona. z_j es la salida de la neurona j-ésima de la capa oculta. Las unidades de salida tienen funciones de activación lineales, mientras que para $f(\cdot)$ se suelen emplear funciones de tipo sigmoide. La Figura 1 muestra un MLP estándar.

Los hiper-parámetros a optimizar durante el entrenamiento son los pesos y el número de neuronas. Generalmente, se establece un número de neuronas h, despues $\mathbf{w}$ y $\mathbf{v}$ son inicializados aleatoriamente, y a partir de un conjunto de N patrones de entrenamiento asociado al problema a resolver, la red es entrenada mediante el algoritmo "back-propagation", junto con métodos de optimización basados en gradiente. El número óptimo de neuronas es seleccionado

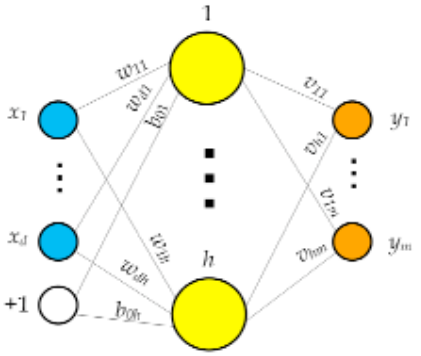

Fig. 1. Arquitectura estándar MLP.

por validación cruzada [1]. Este entrenamiento conlleva un gran número de iteraciones, y una búsqueda exhaustiva del parámetro h, lo que puede suponer un elevado coste computacional en determinadas aplicaciones. Para más detalles de implementación/diseño, ver [1].

3. Extreme Learning Machine

El algoritmo *Extreme Learning Machine* (ELM) está fundamentado en que un MLP compuesto por h neuronas, cuyos pesos de entrada están inicializados aleatoriamente, pueden "aprender" N distintos casos de entrenamiento produciendo un error cero, siendo $N \geq h$, y aproximar cualquier tipo de función continua [2]. Tras inicializar de manera aleatoria los pesos de entrada, un MLP puede ser considerado como un sistema lineal y los pesos de salida pueden obtenerse de manera analítica mediante un simple cálculo de la pseudo-inversa de la matriz de las salidas de las h neuronas ocultas para un determinado conjunto de entrenamiento. De esta forma, dado un conjunto de N vectores de entrada, un MLP con h neuronas ocultas puede aproximar estos N casos con error nulo ($\sum_{i=1}^{N} \|\mathbf{y}_i - \mathbf{t}_i\| - 0$), i.e., existen $\mathbf{v}_j$, $\mathbf{w}_j$ y b_j tal que,

$$\sum_{j=1}^{h} \mathbf{v}_j z_{ij} - \mathbf{t}_i, \quad i - 1, \ldots, N, \tag{2}$$

siendo $z_{ij} - f\left(\mathbf{w}_j^T \mathbf{x}_i + b_j\right)$. Las anteriores N ecuaciones se pueden expresar:

$$\mathbf{ZV} = \mathbf{T}, \tag{3}$$

donde $\mathbf{Z} \in \mathbb{R}^{N \times h}$ es la matriz de salidas de la capa oculta de neuronas del MLP, $\mathbf{V} \in \mathbb{R}^{h \times m}$ es la matriz de pesos de salida, y $\mathbf{T} \in \mathbb{R}^{N \times m}$ es la matriz de "targets" de los N casos de entrenamiento. De esta forma, el entrenamiento del MLP viene dado por la solución del problema de mínimos cuadrados establecido en (3), es decir, los pesos óptimos de la capa de salida son

$$\hat{\mathbf{V}} = \mathbf{Z}^{\dagger}\mathbf{T}, \tag{4}$$

donde $\mathbf{Z}^{\dagger}$ es la pseudo-inversa de Moore-Penrose [5]. ELM proporciona un entrenamiento rápido y eficiente para MLPs [2], aunque es necesario pre-establecer el número de neuronas ocultas. A continuación se describe el algoritmo OP-ELM que permite seleccionar, de manera unívoca, las neuronas que constituyen la capa oculta del MLP [3].

3.1. Selección Óptima de Neuronas

El método *Optimal Prunned-ELM* (OP-ELM) [3] establece un número inicial muy elevado de neuronas ocultas ($h >> N$), y mediante el algoritmo *Least Angle Regression* (LARS) [6] elimina/poda aquellas variables (neuronas) que no son útiles para resolver el problema de mínimos cuadrados. Para ello ordena el conjunto posible de neuronas, conforme a su importancia para resolver (3). La solución obtenida por LARS es única cuando el problema es lineal [6]. La poda de neuronas es realizada mediante validación cruzada del tipo Leave-One-Out (LOO), escogiendo aquella combinación de neuronas (que han sido previamente ordenadas mediante el algoritmo LARS) que proporciona menor error de validación [3]. Por tanto, únicamente son seleccionadas las h^* neuronas más importantes según LARS (con $h^* < h$), y junto con el entrenamiento ELM, se obtiene una solución única y óptima para el diseño de un MLP.

3.2. Experimentos para OP-ELM

A continuación, se emplea un problema sencillo de regresión unidimensional para mostrar las capacidades de OP-ELM. Se han generado artificialmente N=1000 casos de entrenamiento, según la suma de dos señales sinusoidales y ruido aleatorio gaussiano. La Figura 2 muestra con puntos azules el conjunto de datos de entrenamiento. A continuación se ha entrenado un MLP mediante OP-ELM, empleando un tiempo de cálculo de **5,47** s (segundos). La Figura 2 muestra el modelo obtenido por OP-ELM en rojo. Podemos comprobar que el resultado obtenido es muy preciso, y más teniendo en cuenta que únicamente han sido necesarios poco más de cinco segundos para el modelado final.

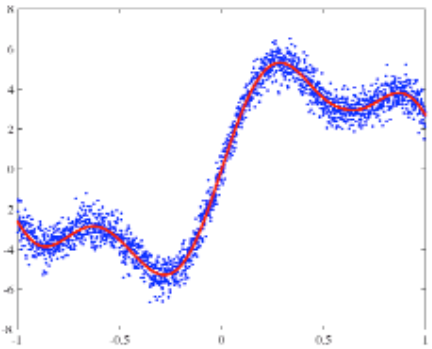

Fig. 2. Regresión unidimensional mediante OP-ELM.

A continuación, se resuelve el conocido problema de regresión *Abalone* [7], que consta de N=4177 casos y n=8 características de entrada. Los hiper-parámetros del MLP han sido seleccionados mediante validación cruzada (10 "folds"), y para el entrenamiento se ha

usado el algoritmo de gradiente conjugado escalado [1]. Para solventar el problema de los mínimos locales, se han realizado 10 inicializaciones aleatorias de pesos, y se ha seleccionado el número de neuronas mediante búsqueda exhaustiva entre 1 y 20 neuronas. El entrenamiento estándar del MLP obtiene un error de test igual a **4,34** tras **2640** s, mientras que OP-ELM obtiene un error de test igual a **4,50** tras emplear **25** s. Podemos comprobar como OP-ELM obtiene resultados prácticamente similares al MLP estándar, siendo el tiempo de cálculo **100** veces más rápido.

4. Método Propuesto: OCoELM

El método OP-ELM sigue siendo sensible a la inicialización de los pesos del MLP, por lo que es necesario realizar múltiples inicializaciones y escoger aquella máquina entrenada que proporciona mejor capacidad de generalización según un criterio de validación cruzada, descartando los restantes MLPs.

Para cada inicialización, OP-ELM proporciona una arquitectura con diferente número de neuronas (una solución distinta). Según esta metodología, se está desperdiciando la gran cantidad de información presente en cada uno de los MLPs entrenados. A pesar de que las máquinas descartadas proporcionan modelos sub-óptimos para el problema, la combinación de las predicciones obtenidas por el conjunto de MLPs puede proporcionar una solución más solida y precisa que la obtenida por un único MLP. Las técnicas de combinación de RNAs se conocen como *Committe Machines* o *Network Ensembles* [4]. La Figura 3 muestra un comité de máquinas, donde se aprovecha la diversidad existente en distintos MLPs combinando sus correspondientes estimaciones para obtener una mejor solución.

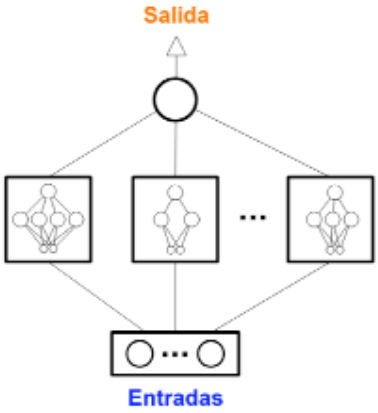

Fig. 3. Comité de MLPs.

Considerar un conjunto de L máquinas diferentes que han sido entrenadas para modelar un determinado problema (con target t), siendo $y_1, y_2, \ldots y_L$ las salidas obtenidas por cada una de las L máquinas. Una primera aproximación es obtener la salida global (o) como el promedio de las L salidas:

$$o = \frac{1}{L}\sum_{l=1}^{L} y_l. \tag{5}$$

Sin embargo, una solución más eficiente es una combinación lineal de los L modelos:

$$o - \sum_{l=1}^{L} \lambda_l y_l, \tag{6}$$

por lo que es necesario resolver el siguiente problema de mínimos cuadrados:

$$\sum_{l-1}^{L} \lambda_l y_{il} - t_i, \quad i - 1, \ldots, N, \tag{7}$$

donde y_{il} es la salida del l-ésimo MLP para el vector $\mathbf{x}_i$. Cada MLP ha sido entrenado mediante OP-ELM. El método propuesto extiende OP-ELM combinando las estimaciones proporcionadas por L^* máquinas (con $L^* < L$), que han sido seleccionadas, de manera automática y óptima, como aquellas que obtienen una mejor aproximación global a t, es decir, una mejor solución global al problema objetivo. Para ello, al igual que OP-ELM, se utiliza el algoritmo LARS (ver Sección 3, [6]) para resolver (7), obteniendo aquellos L^* valores de λ_l que proporcionan una solución global óptima para el problema definido en (7). El método propuesto se denomina *Optimal Committee of Extreme Learning Machines* (OCoELM).

5. Simulaciones para OCoELM

Se han modelado cuatro problemas de regresión: *Friedman*, *Elevators*, *Stocks* y *Banks*, para mostrar las ventajas de OCoELM sobre OP-ELM. Los conjuntos de datos están disponibles en [7]. A continuación, se describen brevemente cada uno de estos problemas:

- *Friedman*. Este problema ha sido generado artificialmente a partir de 10 variables $(x_1, x_2, \ldots, x_{10})$ uniformemente distribuidas en el intervalo $[0, 1]$. Consta de 27179 patrones de entrenamiento, mientras que otros 13589 patrones componen el conjunto de test. La función objetivo a aproximar es: $t = 10\sin(\pi x_1 x_2) + 20(x_3 - 0{,}5)^2 + 10x_4 + 5x_5 + \sigma(0, 1)$, donde $\sigma(0, 1)$ representa ruido blanco gaussiano con media cero y varianza unidad.
- *Elevators*. Este conjunto de datos se obtiene a partir de la tarea de controlar un avión F16. Se pretende modelar la variable relacionada con una acción realizada en los elevadores de la aeronave a partir de 18 variables continuas. Se dispone de 8752 vectores de entrenamiento y 7847 vectores de test.
- *Stocks*. Este conjunto de datos contiene los valores diarios de las acciones para diez empresas del sector aeroespacial, a partir de Enero de 1988 hasta Octubre de 1991. El objetivo es predecir el valor de la acción de una de las empresas a partir

de los valores correspondientes a las restantes empresas, por lo que está compuesto por 9 variables continuas de entrada. El conjunto de datos de entrenamiento consta de 634 patrones, mientras que el conjunto de test consta de 316 vectores de entrada.

- *Bank*. Este problema ha sido generado sintéticamente para modelar cómo los clientes eligen sus bancos. El conjunto de datos consta de 4499 patrones para la fase de entrenamiento y se dispone de 3693 patrones para la fase de test. Cada vector de entrada está definido a partir de 8 variables continuas.

Cada problema es modelado mediante 20 MLPs distintos entrenados mediante OP-ELM, considerando un número máximo de 100 neuronas. Posteriormente, los MLPs entrenados se combinan de manera óptima empleando el método propuesto. La Tabla 1 muestra los resultados obtenidos en términos de error cuadrático medio (MSE, Mean Square Error) y tiempo de cálculo:

- E_{OP-ELM}, MSE obtenido por el mejor MLP entrenado con OP-ELM;
- E_{COoELM}, MSE obtenido por COoELM;
- T, tiempo total empleado para entrenamiento de los 100 MLPs con OP-ELM;
- ΔT, incremento de tiempo para COoELM.

Junto a E_{OP-ELM} y E_{COoELM} se muestra entre paréntesis el número de neuronas óptimo según OP-ELM (h^*) y el número de MLPs seleccionados por COoELM (L^*).

	Friedman	*Elevators*
E_{OP-ELM}	7,12 (64)	14,08E-6 (76)
E_{COoELM}	6,19 (16)	9,39E-6 (16)
T	2464 s.	2246 s.
ΔT	65,59 s.	30,12 s.

	Stocks	*Bank*
E_{OP-ELM}	1,33 (82)	2,10E-3 (80)
E_{COoELM}	0,78 (12)	1,40E-3 (14)
T	69,29 s.	1836 s.
ΔT	0,88 s.	18,15 s.

TABLA I
RESULTADOS OBTENIDOS POR OP-ELM Y COOELM.

Las ventajas de COoELM sobre OP-ELM son evidentes, sobre todo en los problemas *Elevators*, *Stocks* y *Banks*. Por ejemplo, el problema *Stocks* es resuelto en **69,29** s mediante OP-ELM, obteniendo un MLP óptimo con **82** neuronas ocultas que proporciona un MSE igual a **1,33**. Sin embargo, el método COoELM emplea **0,88** s adicionales para diseñar un comité de 12 MLPs que proporciona un MSE igual a **0,78**. Así, COoELM consigue reducir el error MSE en un **41,35** % incrementando únicamente el tiempo de entrenamiento en un **0,98** %, con respecto a OP-ELM. Según los resultados obtenidos, COoELM mejora claramente la capacidad de generalización obtenida por OP-ELM en todos los problemas simulados, empleando un escaso tiempo adicional de cómputo.

6. Conclusiones

Este trabajo propone un nuevo método de entrenamiento óptimo para comités de máquinas basado en *Extreme Learning Machine* (ELM). ELM proporciona un entrenamiento rápido y eficiente para MLPs. El reciente algoritmo *Optimal Prunned-ELM* (OP-ELM) permite seleccionar, de manera automática y óptima, las neuronas de la capa oculta de un MLP entrenado mediante ELM. A partir de OP-ELM, este artículo describe *Optimal Committee of Extreme Learning Machines* (OCoELM) que permite diseñar una combinación óptima de MLPs entrenados con OP-ELM. OCoELM mejora notablemente las prestaciones obtenidas por un solo MLP basado en OP-ELM, ya que salida global del comité aprovecha la diversidad proporcionada por cada una de las máquinas incluidas en el mismo.

Referencias

[1] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York, USA: Oxford University Press (1995).

[2] G.-B. Huang, Q.-Y. Zhu, C.-k. Siew, "Extreme Learning Machine: Theory and Applications", *Neurocomputing*, 70(1-3), pp. 489-501, (2006).

[3] Y. Miche, P. Bias, C. Jutten, O. Simula, A. Lendasse, "A methodology for Building Regression Models using Extreme Learning Machine", *16th European Symposium in Artificial Neural Networks*, pp. 247-252, Bruges, Belgium, 2008.

[4] L. Hansen, P. Salamon, "Neural Network Ensembles", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12 (10), pp. 93-1001, 1990.

[5] A. Bjork. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia, 1996.

[6] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, "Least Angle Regression", *Annal of Statistics*, vol. 32, pp. 407-499, (2004).

[7] http://www.liaad.up.pt/~ltorgo/Regression/DataSets.html