Aprendizaje Multitarea mediante Arquitecturas Neuronales basadas en Subredes Privadas

Pedro J. García-Laencina, Rafael Verdú-Monedero, José-Luis Sancho Gómez
Dpto. Tecnologías de la Información y las Comunicaciones
Universidad Politécnica de Cartagena
e-mail: pedroj.garcia@upct.es, rafael.verdu@upct.es, josel.sancho@upct.es

***Abstract*— Human learning frequently involves learning several tasks simultaneously; in particular, humans compare and contrast similar tasks for solving a problem. Nevertheless, most approaches to machine learning focus on the learning of a single isolated task, *Single Task Learning* (STL). Most of them can be formulated from learning several tasks related to the main task at the same time while using a shared representation, *Multitask Learning* (MTL). This type of learning improves generalization performance for a main task by using the information contained in other related tasks. In this article, we examine distinct schemes used in MTL, propose an new network architecture and test each scheme in two different problems. The proposed scheme makes use of private subnetworks to improve the performance of MTL.**

I. Introducción

La mayoría de los problemas en reconocimiento de patrones son resueltos mediante el aprendizaje de una única tarea. Este aprendizaje (conocido como STL, *Single Task Learning*), obvia las ventajas y el funcionamiento real del aprendizaje humano. En éste, el aprendizaje de una tarea se ve mejorado y completado por el aprendizaje de otras tareas relacionadas con la primera. Así por ejemplo, un persona que durante su formación ha aprendido simultáneamente recursos humanos y administración de empresas realizará más eficientemente la selección del personal de una empresa que otra que únicamente tiene conocimientos de administración de empresas o recursos humanos. Recientemente, muchos trabajos han aplicado estas ventajas al aprendizaje máquina [1]-[4]. Para ello durante el aprendizaje de una tarea principal se añaden tareas extra relacionadas. Este tipo de aprendizaje se conoce como *Aprendizaje Multitarea* (MTL, *Multitask Learning*) [2].

En este artículo se presentan distintas arquitecturas neuronales empleadas en MTL y se propone una nuevo esquema MTL. Para evaluar las prestaciones se han empleado varios problemas artificiales, mostrando los resultados obtenidos la eficiencia del esquema propuesto. El resto del artículo se estructura de la siguiente forma: en la sección 2, se analiza el aprendizaje multitarea y se describen distintos trabajos relacionados. La sección 3 muestra los distintos esquemas neuronales para MTL y se propone una nueva arquitectura. Este esquema emplea subredes privadas para mejorar las prestaciones y la capacidad de generalización del aprendizaje multitarea. En la sección 4 se describen los problemas que hemos usado para analizar las distintas arquitecturas multitarea y se muestran los resultados obtenidos. Finalmente, se exponen las conclusiones principales.

II. Aprendizaje Multitarea

En [5], Baxter introdujo el concepto de *sesgo inductivo* como todo aquello que induce a una máquina a preferir unas hipótesis sobre otras. Esta idea fue desarrollada por Rich Caruana en su tesis, donde acuñó el término de Aprendizaje Multitarea [6]. En este tipo de aprendizaje, la capacidad de generalización de una red entrenada para aprender una determinada tarea mejora al aprenderse simultaneamente junto con distintas tareas relacionadas, es decir, dicha tarea se aprende mejor que en el caso de aprenderla aislada [7]. La tarea que se debe aprender mejor se conoce como la *tarea principal*, mientras que las tareas que ayudan a mejorar la capacidad de generalización de la tarea principal se denominan *tareas extra o secundarias*. Se entiende por dominio a la unión de la tarea principal y las tareas extra.

R. Caruana analiza MTL en perceptrones multicapa (MLP, *Multilayer Perceptron*). La manera más sencilla de implementarlo consiste en añadir salidas extra para aprender las distintas tareas secundarias, junto con la principal, compartiendo todas las tareas la única capa oculta de la red. El hecho de que un conjunto de neuronas ocultas estén conectadas a las salidas asociadas a las distintas tareas permite que lo que se aprenda en una salida contribuya al aprendizaje del resto, mejorando así la capacidad de generalización de la red neuronal.

III. Arquitecturas Neuronales usadas en Aprendizaje Multitarea

Para una mejor comprensión del contenido del artículo, en la tabla I se muestra la nomenclatura que se va a emplear en las siguientes secciones.

A. Arquitecturas Estándar para STL y MTL

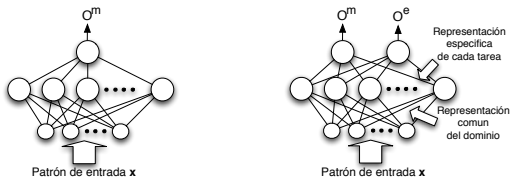
Antes de analizar el aprendizaje multitarea, se describe el esquema STL. Consideresé un conjunto $\mathcal{M}$, asociado a una única tarea, formado por un conjunto de patrones de entrada $\mathbf{X}^m$ y un conjunto de salidas deseadas $\mathbf{T}^m$. Una arquitectura neuronal para STL consta de una capa oculta de neuronas

$\mathbf{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(i)}, \ldots, \mathbf{x}^{(N)}\}$	Conjunto de vectores de entrada
$\mathbf{T} = \{\mathbf{t}^{(1)}, \mathbf{t}^{(2)}, \ldots, \mathbf{t}^{(i)}, \ldots, \mathbf{t}^{(N)}\}$	Conjunto de salidas deseadas
$\mathcal{M} = \{\mathbf{X}^m, \mathbf{T}^m\}$	Conjunto asociado a la tarea principal
$\mathcal{E} = \{\mathbf{X}^e, \mathbf{T}^e\}$	Conjunto para aprender las tareas extra
$\mathbf{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots, x_j^{(i)}, \ldots, x_d^{(i)}]$	Vector de entrada
$\mathbf{t}^{(i)} = [t_1^{(i)}, t_2^{(i)}, \ldots, t_j^{(i)}, \ldots, t_p^{(i)}]$	Vector de salidas deseadas
$\mathbf{o}^m, \mathbf{o}^e$	Salidas asociadas a las tareas

TABLA I
NOMENCLATURA.

completamente interconectadas y una salida asociada a la tarea que se desea aprender (ver Figura 1(a)). Esta red puede ser entrenada para minimizar una función de error entre las salidas de esta red y las salidas deseadas. Por lo tanto, aprende únicamente las salidas deseadas $\mathbf{T}^m$ usando $\mathbf{X}^m$.

En una primera aproximación al MTL, la tarea principal y una o varias tareas extra son aprendidas conjuntamente mediante una misma red con una capa oculta neuronas completamente interconectadas y tantas salidas como tareas a aprender. Consideresé un conjunto $\mathcal{M}$, asociado a una tarea principal, formado por un conjunto de patrones de entrada $\mathbf{X}^m$ y un conjunto de salidas deseadas $\mathbf{T}^m$, y un conjunto $\mathcal{E}$, asociado a las tareas secundarias, formado por $\mathbf{X}^e$ y $\mathbf{T}^e$. En este caso, la red será entrenada para minimizar una función de error entre las salidas de la red y las salidas deseadas para cada tarea. El objetivo es que la red obtenga con exactitud nuevos valores para la tarea principal ante futuras entradas (o conjunto de test). En general, nos interesa aprender con precisión la tarea principal, mientras que el aprendizaje de las tareas extra nos interesa sólo como guía para la tarea principal, mejorando así la capacidad de generalización de está última.



(a) Arquitectura STL estándar.

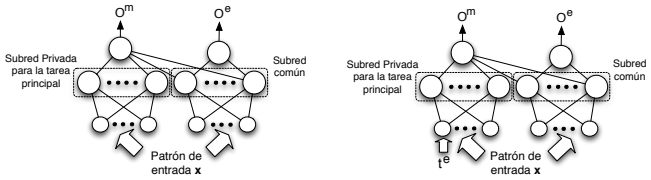
(b) Arquitectura MTL estándar.

Fig. 1. Arquitecturas neuronales estándar para STL y MTL.

La Figura 1(b) muestra la arquitectura MTL estándar con una capa oculta de neuronas y una salida asociada a cada tarea. Se observa como la capa oculta de neuronas está compartida por todas las tareas. Esta es la idea base del aprendizaje multitarea: compartir la información aprendida mientras las tareas se aprenden en paralelo. En cuanto a los pesos de la red, los pesos de la primera capa son actualizados según el error total de todas las tareas, mientras que los pesos de segunda capa son actualizados en función del error de cada tarea a la que se encuentran asociados.

B. Uso de Subredes Privadas en MTL

Es posible mejorar el rendimiento y las prestaciones del MTL usando arquitecturas neuronales más complicadas que el esquema estándar MTL. Para ello, Caruana propuso en su tesis añadir una subred específica o privada que aprendiese únicamente la tarea principal. La Figura 2(a) muestra este esquema donde se tiene dos capas separadas de neuronas ocultas o dos subredes separadas. Una de ellas es una *subred privada* empleada sólamente por la tarea principal, mientras que la otra es la *subred común* compartida por la tarea principal y las tareas secundarias. Esta subred común soporta el aprendizaje multitarea. Esta arquitectura es asimétrica ya que la tarea principal afecta al aprendizaje de la tarea extra mediante la subred común, mientras que la tarea extra no puede afectar a la subred privada reservada para la tarea principal.



(a) Arquitectura MTL con una subred privada para la tarea principal.

(b) Arquitectura MTL con una subred privada para la tarea principal y entradas extra.

Fig. 2. Arquitecturas neuronales MTL con una subred privada.

Hasta ahora se ha supuesto que, durante el aprendizaje de la tarea principal y las tareas secundarias, los vectores de entrada $\mathbf{x}$ son los mismos para todas las neuronas ocultas, según nuestra notación, $\mathbf{X}^m = \mathbf{X}^e = \mathbf{X}$. Usando *entradas extra* es posible aumentar las prestaciones de la arquitectura. Para ello, las salidas deseadas t^e junto con los vectores de entrada $\mathbf{x}$ se usan como entradas de la subred privada para aprender la tarea principal, $\mathbf{X}^m = \{\mathbf{X}, \mathbf{T}^e\}$. La Figura 2(b) muestra esta arquitectura. De esta forma, se añade información a priori acerca del dominio en la subred privada, siendo mejor la generalización de la tarea principal.

C. Arquitectura Multitarea Propuesta

Finalmente, proponemos un nuevo esquema MTL que emplea los conceptos de subredes privadas y entradas extra. En la Figura 3 se muestra el esquema MTL propuesto. Esta arquitectura tiene una subred común que aprende todas las tareas, una subred privada asociada a la tarea principal y otra subred privada asociada a las tareas extra. Cada subred privada está únicamente conectada a la unidad de salida asociada a cada tarea, por lo tanto, estas subredes aprenderán específicamente cada tarea. La subred común está completamente conectada con todas las unidades de salida, por tanto, esta subred aprenderá el dominio completo, tarea principal más tareas

extra. Luego, podemos decir que las subredes privadas trabajan como un esquema STL, mientras que la subred común trabaja como un esquema MTL.

La arquitectura propuesta además hace uso de entradas extra en cada subred privada. La subred privada que aprende específicamente la tarea principal tiene como entradas los vectores $\mathbf{x}$ y $\mathbf{t}^e$, y viceversa para la otra subred privada, siendo sus entradas $\mathbf{x}$ y $\mathbf{t}^m$.

Este esquema MTL presenta muchas ventajas frente al resto de esquemas MTL propuestos anteriormente, mejorando así la capacidad de generalización y las prestaciones.

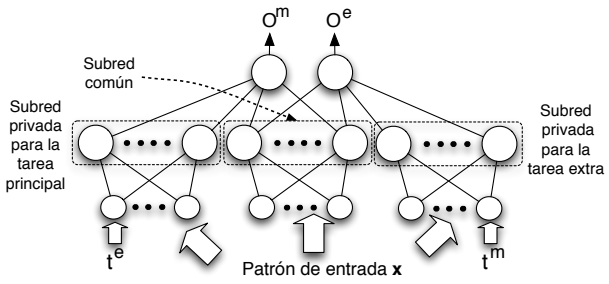


Fig. 3. Arquitectura MTL propuesta con dos subredes privadas y entradas extra para cada una de las tareas.

IV. Resultados Experimentales

Tras una breve descripción de los problemas simulados, se presentan los resultados obtenidos en cada uno de los problemas. Con estos resultados, se muestra el beneficio que aporta añadir las subredes privadas y entradas extra durante el proceso de aprendizaje, así como también se muestra la gran capacidad de generalización de la arquitectura propuesta.

El objetivo es mejorar el aprendizaje de una tarea principal usando esquemas multitarea, por tanto, los resultados presentados muestran el rendimiento del aprendizaje de la tarea principal.

A. Problema Binario

Este problema procede de la tesis de R. Caruana [6]. Las entradas son vectores binarios de 8 bits, $\mathbf{x} = [x_1, \ldots, x_8]$, por tanto hay 256 casos. Consideraremos dos tareas relacionadas:

$$TareaA : x_1 \vee (2 \cdot Nbits(x_2, x_3, x_4) < Nbits(x_5, x_6, x_7, x_8))$$
$$TareaB : x_1 \wedge (2 \cdot Nbits(x_2, x_3, x_4) < Nbits(x_5, x_6, x_7, x_8))$$

donde $Nbits()$ devuelve el número de bits iguales a 1 en su argumento, $\vee$ es la operación lógica OR y $\wedge$ es la operación lógica AND. Se ha usado la tarea A como tarea principal, y la tarea B como extra. De los 256 casos, se han seleccionado aleatoriamente 128 casos para el conjunto de entrenamiento y el resto para el conjunto de test.

B. Problema de Monk

En este problema [8], cada vector de entrada tiene 6 componentes, $\mathbf{x} = [x_1, \ldots, x_6]$, que describen los seis atributos que caracterizan a un robot artificial. Cada uno de estos atributos están codificados con valores enteros entre 1 y 4.

Este problema consta de 3 tareas, de las cuales en este trabajo consideraremos dos de ellas. En la tarea A, se pretende identificar aquellos patrones con $(x_1 == x_2) \vee (x_5 = 1)$, mientras que en la tarea B se pretende identificar aquellos patrones donde al menos dos de sus seis atributos tomen el valor codificado como 1. Se ha empleado la tarea A como tarea principal y la tarea B como tarea secundaria. Hay 432 casos, 124 casos como conjunto de entrenamiento y los 432 casos como conjunto de test.

C. Resultados Obtenidos

La función de error que se minimiza durante el proceso de entrenamiento es la raiz del error cuadrático medio de las salidas de la red y las salidas deseadas. En todas las neuronas ocultas, la función de activación es la tangente hiperbólica. El entrenamiento de los pesos de la red se ha realizado mediante el método por descenso de gradiente en modo bloque, con término de momento igual a 0.5 y tasa adaptativa. Los gradientes de los pesos son calculados usando el algoritmo de retropropagación [9]. Los pesos han sido inicializados aleatoriamente con valores comprendidos en el intervalo [-0.5, 0.5]. El conjunto de test se emplea para evaluar el porcentaje de patrones clasificados correctamente, una vez la red ha sido entrenada.

En el problema binario, hemos usado los siguientes números de neuronas: 100 neuronas en los esquemas estándar MTL y STL; 40 neuronas, en cada subred, en el esquema MTL con una subred privada; y 20 neuronas, en cada subred, en el esquema MTL propuesto. La Figura 4 muestra las curvas asociadas a la probabilidad de acierto para la tarea A en el conjunto de test con respecto de las iteraciones. Cada curva es el promedio de 25 simulaciones. La tabla II resume los resultados obtenidos en el problema binario.

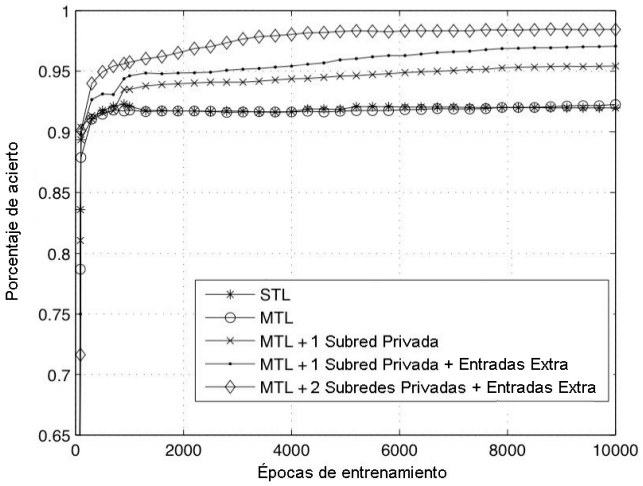


Fig. 4. Porcentaje de acierto para la tarea A en el conjunto de test del problema binario.

Arquitectura	% Acierto	Desv. estándar
STL	91.97	0.171
MTL	92.25	0.164
MTL + 1 subred privada	95.42	0.017
MTL + 1 subred privada + Entrada extra	97.10	0.015
MTL + 2 subredes privadas + Entradas extra	98.44	0.022

TABLA II

PROBABILIDAD DE ACIERTO Y DESVIACIONES ÉSTANDAR EN EL PROBLEMA BINARIO

En el problema de Monk, se ha usado los siguientes números de neuronas: 6 neuronas en los esquemas estándar MTL y STL; 3 neuronas, en cada subred, en el esquema MTL con una subred privada; y 2 neuronas, en cada subred, en el esquema MTL propuesto. La Figura 5 muestra las curvas asociadas a la probabilidad de acierto para la tarea A en el conjunto de test con respecto de las iteraciones. Cada curva es el promedio de 25 simulaciones. La tabla III resume los resultados obtenidos en el problema binario.

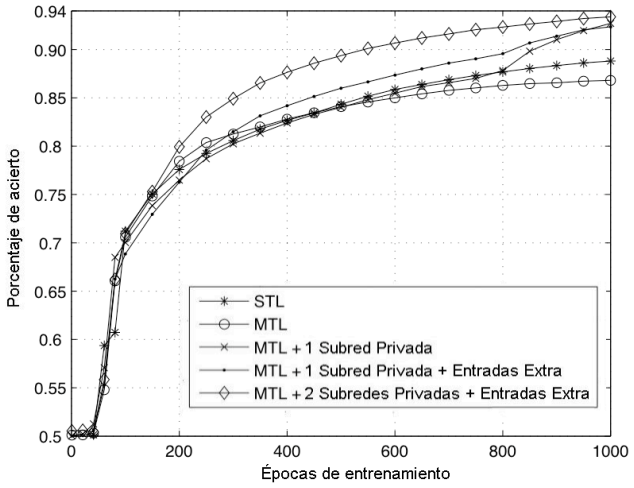


Fig. 5. Porcentaje de acierto para la tarea A en el conjunto de test del problema de Monk.

Arquitectura	% Acierto	Desv. estándar
STL	88.81	0.151
MTL	86.81	0.131
MTL + 1 subred privada	92.71	0.075
MTL + 1 subred privada + Entrada extra	92.35	0.089
MTL + 2 subredes privadas + Entradas extra	93.37	0.069

TABLA III

PROBABILIDAD DE ACIERTO Y DESVIACIONES ESTÁNDAR EN EL PROBLEMA DE MONK

V. CONCLUSIONES

Los resultados obtenidos muestran como MTL puede mejorar el aprendizaje de la tarea principal, aumentando el rendimiento y la capacidad de generalización. Usando subredes privadas para cada tarea, las prestaciones del esquema estándar MTL son claramente mejorados. Las subredes privadas aprenden específicamente una tarea, mientras que la subred común aprende todas las tareas simultáneamente. Las simulaciones sobre dos problemas muestran como el esquema propuesto obtiene mejor generalización y mayor velocidad de convergencia que el resto de alternativas consideradas. Además, la nueva arquitectura no sólamente mejora el aprendizaje de la tarea principal, también mejora el aprendizaje de la tarea extra. Por tanto, el esquema propuesto es óptimo en problemas donde sea necesario aprender bien todas las tareas. Cabe destacar que estas ventajas son conseguidas sin aumentar la complejidad de la red, es decir, sin incrementar el número de neuronas usado en el esquema MTL estándar.

Este trabajo puede estimular trabajos futuros en distintas direcciones. Así por ejemplo, se puede realizar un estudio en problemas con más de dos clases y problemas de regresión y el uso de una tasa de aprendizaje distinta para cada tarea.

AGRADECIMIENTOS

Este trabajo está parcialmente financiado por el Ministerio de Educación y Ciencia a través del proyecto TIC2002-03033.

REFERENCIAS

[1] Y. S. Abu-Mostafa, Learning from hints in neural networks. *Journal of Complexity*, **6(2)**, pp.192-198, 1990

[2] R. Caruana, Multitask learning: a knowledge-based source of inductive bias. *Proceedings of the 10th International Conference of Cognitive Science*, pp. 41-48, 1993

[3] S. Thrun, Is learning the n-thing any easier than learning the first?. *Advances in Neural Information Processing Systems (NIPS)*. pp. 640-646, 1996

[4] S. C. Suddarth, Y. L. Kergosien, Rule-injection hints as a means of improving network performance and learning time.*Proceedings of the EURASIP Workshop on Neural Networks* (Lecture Notes in Computer Science, Vol. 412). pp 120-129, 1990

[5] J. Baxter, *Learning Internal Representations*. Ph. D. Thesis, The Flinders University of South Australia, 1994

[6] R. Caruana, *Multitask learning*. Ph. D. Thesis, Carnegie Mellon University, 1997

[7] J. Baxter, A model of inductive learning bias learning. *Journal of Artificial Intelligence Research*, **12**, pp. 149-98, 2000

[8] S. Thrun et al., The MONK´s problem: A performance comparison of different learning algorithms, Technical Report, CMU-CS-91-197, Carnegie Mellon University, Pittsburgh, PA, 1991.

[9] D.E. Rumelhart, G. E. Hinton, R. J. Williams, Learning Representations by Back-propagating Errors. *Nature* **323** pp. 533-536, 1986