Imputación de Datos Incompletos y Clasificación de Patrones mediante Aprendizaje Multitarea

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

***Abstract*— Almost all research on supervised learning is based on the assumption that training data are completely observable, but it is not a common situation because real world databases are rarely complete. The ability of handling missing data has become a fundamental requirement for machine learning. Up to now, proposed methods consider the problem as two separated tasks, main task and imputation task, and solve them separately (Single Task Learning, STL). In this paper, a new effective method is proposed to handle missing features in incomplete databases with Multitask Learning (MTL). This approach uses the imputation task as extra task and learning in parallel with the main task. Thus, imputation is guided and oriented by the learning process, i.e., imputed values are those that contribute to improve the learning. In this paper we use the advantages of MTL to handling missing data and analyze its robustness for handling different missing variables in real an artificial data sets.**

I. Introducción

En reconocimiento estadístico de patrones, se conoce como *aprendizaje* al modelado de un conjunto de datos conocidos (conocido como conjunto de entrenamiento), asociados a una determinada tarea, mediante la búsqueda de los parámetros que determinan dicho modelo. El objetivo es conseguir una elevada capacidad de generalización, es decir, el modelo estima nuevos valores para la tarea aprendida ante entradas futuras (conocido como conjunto de test) que no han participado en el aprendizaje de dicha tarea. El término tarea se refiere a una función objetivo que es aprendida a partir del conjunto de entrenamiento. La mayoría de los trabajos en aprendizaje supervisado parten de la suposición de que los conjuntos de entrenamiento y test están completos, es decir, no presentan valores perdidos en alguna de sus características. Esta suposición es erronea, ya que la mayoría de bases de datos que caracterizan problemas reales tienen datos incompletos. Las muestras que presentan valores perdidos se conocen como casos incompletos, [1][2]. Un claro ejemplo es la recogida de datos de cualquier encuesta, pues es frecuente que individuos encuestados no respondan a una o más preguntas del cuestionario.

En este artículo, se propone un nuevo método que combina la clasificación de patrones y la imputación (o asignación) de valores perdidos mediante una arquitectura neuronal basada en el aprendizaje multitarea (MTL, Multitask Learning). Un esquema MTL añade tareas extra relacionadas con una tarea principal y las aprende simultáneamente, [3]. La tarea que realmente se desea aprender se conoce como tarea principal, mientras que las tareas que se añaden durante el proceso de aprendizaje y son usadas como guías del aprendizaje por la tarea principal se conocen como tareas extra [4]. En el método propuesto, la imputación de datos es guiada y orientada por el proceso de aprendizaje de la tarea principal, es decir, los valores imputados son los que van a contribuir a mejorar el aprendizaje de la tarea principal de clasificación. Se ha analizado las prestaciones del método propuesto en distintos problemas reales y artificiales; los resultados obtenidos muestran la eficacia de usar esquemas MTL en imputación de valores perdidos. El resto del artículo se estructura de la siguiente forma: en la Sección 2, se da una breve introducción al problema de los valores perdidos y mostrando las soluciones comunes a este problema. La imputación de datos mediante redes neuronales es analizada en la Sección 3. En la Sección 4, se describe el MTL y se presenta el método propuesto. La Sección 5 describe los conjuntos de datos usados para evaluar las prestaciones del método propuesto y muestra los resultados obtenidos. Finalmente, la Sección 6 termina el artículo con las principales conclusiones y futuros trabajos relacionados.

II. Imputación de Valores Perdidos

En la Tabla I, se muestra la nomenclatura usada en este trabajo. Considérere un problema caracterizado por un conjunto de datos $\mathbf{D}$. Cada vector de entrada $\mathbf{x}^{(i)}$ puede tener valores perdidos en alguna de sus características. $\mathbf{D}$ puede dividirse en conjunto con datos completos $\mathbf{D}^{com}$ y en un conjunto con datos incompletos $\mathbf{D}^{mis}$.

$\mathbf{D} = \{\mathbf{X}, \mathbf{T}\}$	Conjunto de datos
$\mathbf{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(i)}, \ldots, \mathbf{x}^{(N)}\}$	Conjunto de vectores de entrada
$\mathbf{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots, x_j^{(i)}, \ldots, x_d^{(i)}]$	Vector de entrada
$\mathbf{T} = \{\mathbf{t}^{(1)}, \mathbf{t}^{(2)}, \ldots, \mathbf{t}^{(i)}, \ldots, \mathbf{t}^{(N)}\}$	Conjunto de salidas deseadas
$\mathbf{t}^{(i)} = [t_1^{(i)}, t_2^{(i)}, \ldots, t_j^{(i)}, \ldots, t_c^{(i)}]$	Vector de salidas deseadas
$\mathbf{D} = \mathbf{D}^{com} \cup \mathbf{D}^{mis}$	
$\mathbf{D}^{com} = \{\mathbf{X}^{com}, \mathbf{T}^{com}\}$	Conjunto de datos completos
$\mathbf{D}^{mis} = \{\mathbf{X}^{mis}, \mathbf{T}^{mis}\}$	Conjunto de datos incompletos

TABLA I
NOMENCLATURA.

La manera más sencilla de tratar con datos perdidos es simplemente eliminar dichos valores y trabajar sólo con los datos completos. Sin embargo, en las bases de datos reales, los valores perdidos pueden aparecer en cualquier componente de los vectores $\mathbf{x}^{(i)}$, pudiendo suponer una parte importante del conjunto de datos, por lo que eliminarlos implica una perdida sustancial de información, [1].

Una solución mucho más adecuada es la imputación. La imputación de datos es la etapa en la que los "huecos" correspondientes a los valores perdidos del conjunto de datos son rellenados por valores concretos. En nuestro caso, se pretende obtener un conjunto de datos completo y consistente que permita mejorar los resultados obtenidos durante el proceso de clasificación con datos incompletos. Un procedimiento muy habitual es imputar a la media de la clase, es decir, un "hueco" en la característica j-ésima de un patrón incompleto es rellenado con la media de los valores de dicha característica en los casos completos pertenecientes a la misma clase que el caso incompleto bajo estudio. Este método normalmente lleva a soluciones que están alejadas de la óptima, ya que la imputación se realiza sin tener en cuenta la naturaleza del problema, [2]. Otros métodos muy usados, como la imputación mediante redes neuronales, realizan la imputación sin tener en cuenta la resolución de la tarea principal que se desea resolver. Todo ello es solventado por el método propuesto.

III. Imputación con Redes Neuronales Artificiales

Las redes neuronales artificiales (Artificial Neural Networks, ANN), como aproximadores universales, han sido usadas con éxito en la resolución de multitud de problemas [5]. En concreto, se han empleado para imputación de datos incompletos en muchos trabajos, [6]-[8]. En todos ellos se trabaja de una manera similar, ya que dividen el problema en dos tareas: *tarea de imputación (Tarea-I)*, y *tarea principal, Tarea-P*. La Tarea-I está asociada a las características que presenta valores perdidos, mientras que la Tarea-P es la tarea que realmente se desea aprender. Este tipo de aprendizaje es conocido como STL (Single Task Learning) pues se aprende únicamente una tarea aislada en cada etapa.

La Figura 1 muestra el proceso de imputación en un problema con tres entradas y valores perdidos en el tercer atributo. Inicialmente, en el primer paso, una ANN se emplea para estimar los valores incompletos, aprende la Tarea-I, usando los datos completos $\mathbf{D}^{com}$, es decir, las salidas de la red están asociadas a los atributos con datos perdidos, mientras que las entradas son están asociadas al resto de atributos y la red es entrenada usando $\mathbf{D}^{com}$. En el paso 2, con la red entrenada se estiman los valores incompletos usando $\mathbf{D}^{mis}$. Por último, otra ANN es entrenada para aprender la Tarea-P con el conjunto de datos $\mathbf{D}'$, es decir, $\mathbf{D}^{com}$ y $\mathbf{D}^{mis}$ con los valores estimados en el paso 2.

Este método tiene dos inconvenientes o limitaciones. En el paso 1, la red aprende las características que presentan valores perdidos obviando la información de los casos incompletos, ya que sólo considera $\mathbf{D}^{com}$. Además, el problema es resuelto aprendiendo por separado cada tarea, omitiendo la relación entre ambas tareas.

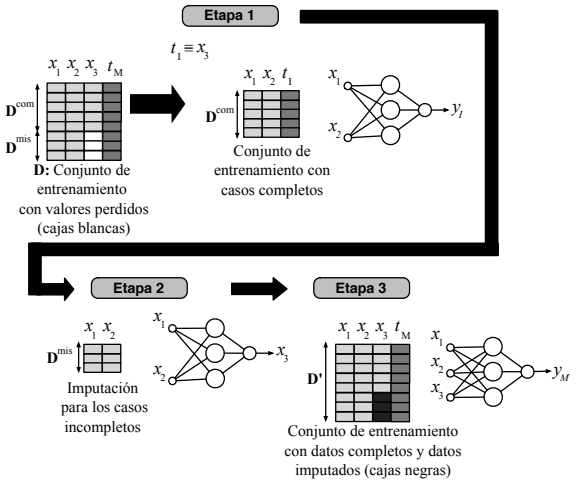

Fig. 1. Imputación mediante redes neuronales. En la primera etapa, la red aprende x_3 (atributo con valores perdidos) usando los casos completos. A continuación la red entrenada estima los valores perdidos. Por último, otra red distinta es entrenada para aprender la Tarea-P usando el conjunto $\mathbf{D}'$.

IV. Imputación basada en Aprendizaje Multitarea

A. Aprendizaje Multitarea

Rich Caruana introdujo el concepto de Aprendizaje Multitarea, [4]. En este tipo de aprendizaje, la capacidad de generalización de una red entrenada para aprender una determinada tarea (*tarea principal*) mejora al aprenderse simultaneamente junto con distintas tareas relacionadas (*tareas extra o secundarias*), es decir, dicha tarea se aprende mejor que en el caso de aprenderla aislada. 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.

B. 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. De esta manera, se tienen dos capas separadas de neuronas ocultas o dos subredes separadas. Una de ellas es una *subred privada* empleada solamente por la tarea principal, mientras que la otra es una *subred común* compartida por la tarea principal y las tareas secundarias. La 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.

C. Método de Imputación Propuesto

En un problema de reconocimiento de patrones con valores perdidos, es necesario resolver dos tareas: la tarea de imputación (problema de regresión), Tarea-I, y la tarea principal (problema de clasificación o de regresión, según la aplicación de que se trate), Tarea-P. En este artículo, se considera la Tarea-P como un problema de clasificación cuyos datos presentan valores perdidos.

Considérese un conjunto de entrada $\mathbf{X}$ con datos incompletos en la característica j-ésima. La arquitectura neuronal propuesta se muestra en la figura 2 (etapa 1 y 2). En este esquema hay dos salidas, y_I para la Tarea-I y y_P para la Tarea-P. La salida y_P está asociada a una *subred privada* que sólo aprende la Tarea-P, mientras que y_I está asociada a la *subred común* que aprende simultáneamente la Tarea-P y la Tarea-I. Por lo tanto, la arquitectura propuesta imputa datos incompletos aprendiendo la Tarea-P, es decir, la tarea principal guía el aprendizaje de la tarea de imputación.

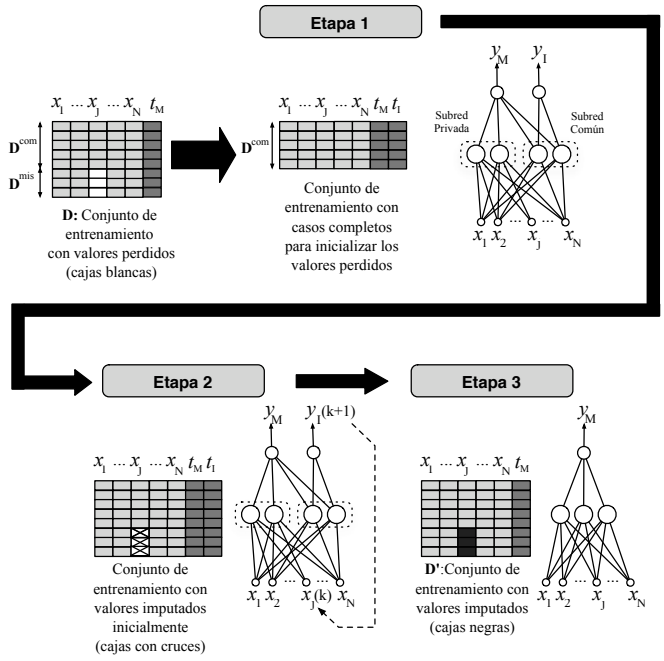

Fig. 2. Método propuesto que combina la imputación y la clasificación mediante una arquitectura neuronal basada en MTL. En una primera etapa, se inicializan los valores perdidos usando únicamente los datos completos. A continuación se aprenden en paralelo las tareas de imputación y la tarea principal, guiando esta última el proceso de imputación. Finalmente, mediante un esquema STL se aprende la tarea principal usando el conjunto de datos con los valores imputados en la etapa anterior.

Una cuestión importante es que las unidades de entrada no están completamente conectadas con todas las neuronas. La subred privada está conectada a todas las entradas, mientras que la subred común está únicamente conectada a las entradas que no presentan valores perdidos x_j. La subred común aprende la caracteristíca x_j, por lo que este atributo no puede ser una entrada en dicha subred. Otra cuestión a destacar es el hecho de que los patrones $\mathbf{x}$ con valores imputados no tienen ningún valor para $\mathbf{t_I}$ ya que es desconocido. Por esta razón, la componente del error $\mathbf{y_I} - \mathbf{t_I}$ se establece a cero para todos estos patrones incompletos durante el entrenamiento.

Atendiendo al esquema mostrado en la figura 2, el método propuesto tienen las siguientes fases:

1) En la primera etapa se inicializan los valores perdidos en x_j entrenando una ANN usando sólamente los casos completos, $\mathbf{D}^{com}$.
2) En esta etapa, la red es entrenada para imputar los valores finales de los datos incompletos en x_j. Para ello se aprende en paralelo la Tarea-P y la Tarea-I durante un número de épocas. Después, los casos incompletos son actualizados usando las salidas y_I. Este proceso se repite durante K iteraciones, con $k = 1, 2, \ldots, K$. Por ejemplo, en la iteración k-ésima, la red es entrenada usando los casos completos e incompletos, cuyos valores han sido rellenados usando y_I de la iteración anterior $k-1$. En nuestras simulaciones hemos usado 200 épocas y $K = 75$ iteraciones (200×75 épocas totales).
3) Finalmente, una red STL es entrenada para aprender específicamente la Tarea-P usando el conjunto de datos $\mathbf{D}'$, formado por los casos completos y los casos incompletos cuyos valores perdidos han sido imputados en la étapa anterior.

V. Resultados Experimentales

Tras una breve descripción de los problemas usados, se muestran los resultados obtenidos. Éstos prueban como el uso de MTL en imputación de valores perdidos mejora el porcentaje de acierto. En nuestras simulaciones se ha probado las prestaciones del método propuesto en dos problemas: Iris y Pima.

A. Bases de datos usadas

En el problema Iris, el objetivo es clasificar lirios basándose en cuatro características. Esta base de datos contiene 50 casos para cada uno de los tres tipos de lirios: setosa, versicolor y virginica. Hay 150 patrones, donde 50 son usados como conjunto de entrenamiento y los 100 restantes como conjunto de test. Como este conjunto de datos carece de valores perdidos, se elminaran aleatoriamente un porcentaje de datos, $p(\%)$, en el tercer atributo para probar las prestaciones del método propuesto. Se han evaluado dos situaciones: conjunto de entrenamiento incompleto y conjunto de test completo, y conjunto de entrenamiento y test incompletos.

En el problema Pima se pretende diagnosticar diabetes a partir de un conjunto de datos obtenido sobre una población de mujeres de la población de indios pima americanos. Cada

muestra tiene ocho entradas y una salida asociada a la diagnosis de la diabetes. El problema tiene 3 conjuntos de datos distintos. Uno se usa como conjunto de test y consiste en 332 casos completos. Los dos restantes se usan para el entrenamiento: uno con sólo 200 casos completos y el otro tiene 200 casos completos y 100 incompletos. Los valores perdidos están presentes en tres de las ocho características.

B. Resultados

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 termino de momento igual a 0.5 y tasa adaptativa. Los gradientes de los pesos son calculados usando el algoritmo de retropropagación. Los pesos han sido inicializados aleatoriamente con valores comprendidos en el intervalo $\left[-\frac{1}{2}\cdot\sqrt{\frac{3}{d}}, +\frac{1}{2}\cdot\sqrt{\frac{3}{d}}\right]$, siendo d el número de entradas. Las salidas deseadas para la Tarea-P de clasificación han sido codificadas en binario usando 0.8 y 0.2 en lugar de 1 y 0. En los dos problemas, se han usado 4 neuronas ocultas en cada subred.

p(%)	**Sólo casos completos**	**Imputación STL (Datos perdidos en entrenamiento)**	**Imputación STL (Datos perdidos en entrenamiento y test)**
8%	5.85 ± 1.61	2.17 ± 0.77	2.72 ± 0.73
16%	6.60 ± 2.25	3.04 ± 0.81	3.44 ± 0.63
24%	6.89 ± 2.61	3.39 ± 0.76	5.11 ± 0.44
32%	7.06 ± 2.33	3.97 ± 1.10	5.47 ± 0.87
40%	9.27 ± 2.53	4.65 ± 1.01	5.76 ± 0.65
48%	10.78 ± 1.49	4.92 ± 1.01	6.08 ± 0.75

p(%)	**Imputación MTL (Datos perdidos en entrenamiento)**	**Imputación MTL (Datos perdidos en entrenamiento y test)**
8%	1.58 ± 0.17	1.62 ± 0.16
16%	1.68 ± 0.22	1.72 ± 0.21
24%	2.25 ± 0.14	2.89 ± 0.11
32%	3.02 ± 0.53	4.12 ± 0.33
40%	3.53 ± 0.66	4.60 ± 0.56
48%	3.44 ± 0.33	4.14 ± 0.34

TABLA II
PROBABILIDAD DE ERROR (MEDIA (%) DE 15 SIMULACIONES Y DESVIACIONES ESTÁNDAR) PARA EL PROBLEMA IRIS

	Probabilidad de error(%)
Sólo casos completos	24.50 ± 2.49
Método de imputación STL	22.72 ± 3.93
Método de imputación MTL	20.12 ± 1.06

TABLA III
PROBABILIDAD DE ERROR (MEDIA DE (%) 15 SIMULACIONES Y DESVIACIONES ESTÁNDAR) PARA EL PROBLEMA PIMA

Todos los resultados son la media de 15 simulaciones. La tabla 2 muestra los resultados finales en el problema Iris para diferentes porcentajes de datos perdidos en la tercera característica. En este problema el error de clasificación sin eliminar datos es alrededor del 3%. Como se puede ver en la tabla 2, el método propuesto basado en MTL obtiene mejores resultados en la clasificación que el método basado en STL, siendo además la desviación éstandar de la probabilidad de error menor, es decir, el método propuesto también aporta robustez. En la Tabla 3 se muestran los resultados finales en el problema Pima. Como en el caso anterior, nuestro método consigue mejorar la probabilidad de error con respecto a los resultados obtenidos en el caso de usar solo los casos completos y en el caso de emplear imputación basada en STL.

VI. CONCLUSIONES

En este artículo, se ha presentado un nuevo método para abordar problemas de clasificación cuyos datos presentan valores perdidos. Usando un esquema neuronal del tipo MLP (Multilayer Perceptron), el método emplea el Aprendizaje Multitarea (MTL, Multitask Learning) como herramienta para mejorar tanto el aprendizaje como la imputación. La tarea de clasificación ayuda a la tarea de imputación y lo guía durante el aprendizaje de ambas. Los resultados obtenidos en problemas reales y artificiales prueban prueban las ventajas del esquema MTL propuesto frente a alternativas basadas en el aprendizaje STL, donde la imputación no es dirigida a la resolución de la tarea que realmente se desea aprender.

Son varias las lineas de trabajo futuras que se pretenden abordar. Las principales son un estudio exhaustivo del método propuesto en más problemas reales que presenten valores perdidos, y el uso de otras maquinas de aprendizaje para imputación basada en MTL, como por ejemplo, Máquinas de Vectores Soporte (SVM, Support Vector Machine).

AGRADECIMIENTOS

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

REFERENCIAS

[1] R. J. A. Little, D. B. Rubin, *Statistical Analysis with Missing Data*, Wiley, New Jersey, 2002.
[2] J. L. Schafer, *Analysis of Incomplete Multivariate Data*, Chapman & Hall, London, 1997.
[3] R. Caruana, Multitask learning: a knowledge-based source of inductive bias. *Proceedings of the 10th International Conference of Cognitive Science*, pp. 41-48, 1993.
[4] R. Caruana, *Multitask learning*. Ph. D. Thesis, Carnegie Mellon University, 1997.
[5] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, 1995.
[6] Z. Ghahramani, M. I. Jordan, Learning from incomplete data. Center for Biological and Computational Learning, Massachusetts Institute of Technology, Tech, Report 108, Cambridge, 1994.
[7] S. Nordbotten, Neural Network Imputation Applied to the Norwegian 1990 Census Data, *Journal of Official Statistics*, Svante Öberg, Vol. 12, No.4, pp. 385-401,1996.
[8] L. Kallin, Missing data and the preprocessing perceptron, UMINF-04.02, Dept. of Computing Science, Umeå University, 2004.
[9] S.Y. Yoon, S.Y. Lee, Training algorithm with incomplete data for feed-forward neural networks. *Neural Processing Letters*, Springer, No.10, pp. 171-179, 1999.