Árboles de redes neuronales autoorganizativas

Diego H. Milone, José C. Sáez, Gonzalo Simón, Hugo L. Rufiner.

Laboratorio de Cibernética, Departamento de Bioingeniería, Facultad de Ingeniería, Universidad Nacional de Entre Ríos, Ruta 11 Km. 10, Oro Verde (CP: 3101), Entre Ríos, Argentina.
E-Mail: cyberlab@fi.uner.edu.ar

Resumen: Un campo muy importante de la inteligencia artificial es el de la clasificación automática de patrones. Para este tipo de tareas se han utilizado diferentes técnicas. En este trabajo se presenta una combinación de árboles de decisión y redes neuronales autoorganizativas como una alternativa para atacar el problema. Para la construcción de estos árboles se recurre a un proceso de crecimiento. En este proceso es necesario evaluar el rendimiento en la clasificación de uno o varios nodos ante distintas configuraciones para tomar decisiones que optimicen la estructura y el desempeño del árbol de redes neuronales autoorganizativas. Para ello se define un grupo de coeficientes que cuantifican el rendimiento y se desarrolla un algoritmo de crecimiento basado en estos coeficientes. En las pruebas se realiza una comparación con otros métodos de clasificación utilizando métodos de validación cruzada y bases de datos reales y artificiales.

Palabras clave: Árboles de decisión. Redes neuronales autoorganizativas. Clasificación automática.

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
D. H. Milone, J. C. Sáez, G. Simón & H. L. Rufiner; "Arboles de redes neuronales autoorganizativas"
Anales del 1er Congreso Latinoamericano de Ingeniería Biomédica, Mazatlán 98, Vol. 1, pp. 779-784, Nov, 1998.

Introducción y antecedentes

La clasificación de patrones es una importante herramienta utilizada en el análisis de datos experimentales, reconocimiento automático, visión por computadora y otras disciplinas ingenieriles y científicas. Los algoritmos de aprendizaje maquinal forman parte de lo que se denomina clasificación automática y están encargados de extraer la información relevante de un conjunto de patrones que permita su clasificación. Estos algoritmos pueden ser supervisados y no supervisados, mientras que desde otro punto de vista pueden agruparse en simples y jerárquicos [1].

Los árboles de decisión (DT) y las redes neuronales artificiales (ANN) son dos técnicas ampliamente utilizadas para la implementación de clasificadores. Los DT generan un conjunto de particiones de los datos basándose en una estructura jerárquica de nodos en los que se realizan comparaciones sobre alguna componente del vector de características. Los DT pueden ser binarios o *n*-arios, de acuerdo a la cantidad de particiones realizadas en cada nodo. Dos de los algoritmos más utilizados son ID3 y CART [9].

Las redes neuronales [2] están formadas por un conjunto de unidades de procesamiento no lineal altamente interconectadas, que procesan en paralelo un conjunto de datos para extraer información. Kohonen introdujo en 1982 [1,6] un algoritmo de autoorganización que produce mapas ordenados con el objeto de resolver problemas prácticos de clasificación y reconocimiento de patrones. Estos mapas fueron denominados mapas autoorganizativos (SOM).

Una alternativa para solucionar algunas limitaciones de los DT, consiste en la implementación híbrida de DT y ANN. Sankar y Mammone presentaron en 1991 [7] los denominados árboles de redes neuronales (NTN). Este tipo de enfoque permite aprovechar las ventajas de la clasificación jerárquica y crear fronteras de decisión más complejas con menos nodos. El problema con las particiones generadas mediante hiperplanos paralelos u ortogonales a los ejes coordenados es que cuando las fronteras son complejas se requieren gran cantidad de nodos o reglas para aproximarlas. Los NTN son DT que implementan la tarea de decisión en los nodos mediante una red neuronal. De esta manera la decisión que se toma en cada nodo se basa en reglas más complejas, lo que permite aproximar mejor las fronteras. Se han utilizado diversas ANN en los nodos, entre ellas se pueden citar: perceptrones simples [7], perceptrones multicapa (MLP) [8,9] y SOMs [10]. La cantidad de particiones que se producen en cada nodo puede ser fija [7,8] o variable [9]. Cuando la cantidad de clases generadas puede variar para cada nodo, el NTN tiene la posibilidad de adoptar una configuración más adecuada para cada problema.

En este trabajo se presenta un árbol de redes neuronales autoorganizativas (ARNA) basado en clasificadores SOMs. Esta combinación permite que las redes no supervisadas separen los patrones de acuerdo a la distribución natural de los mismos y se aprovechen las características jerárquicas de los árboles. El algoritmo permite que en las primeras capas o nodos se separen los grupos de patrones más alejados entre sí (o más fácilmente separables) y en las capas finales se separen los patrones de manera más fina (es decir, los más difícilmente separables). Un problema importante es como decidir acerca de la cantidad de particiones a realizar en cada nodo en el caso de árboles n-arios. Para atacar este problema se establecieron criterios basados en los coeficientes de clasificación que se describen en la sección siguiente.

Este artículo se organizará de la siguiente forma. A continuación se presentan los coeficientes utilizados en el crecimiento de los ARNA. En la sección de Métodos se explica el algoritmo de entrenamiento y se detallan los datos utilizados en las pruebas. Seguidamente se presentan los resultados correspondientes y se discuten los mismos, finalizando con las conclusiones del trabajo.

Coeficientes de clasificación

Dado un clasificador general se define $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_P\}$ donde $\boldsymbol{x}_i \in \Re^D$, como el conjunto de patrones de entrada. Estos patrones de entrada corresponden a *M* clases que se denominan clases de entrada $C^I = \{C_1^I, C_2^I, ..., C_M^I\}$.

De la misma forma en que los patrones de entrada se agrupan según las clases a las que pertenecen realmente, también se pueden agrupar de acuerdo a las clases C_j^O en que son separados por el clasificador. Éstas últimas forman el conjunto de clases de salida $C^O = \{C_1^O, C_2^O, ..., C_N^O\}$.

Un elemento importante para el desarrollo posterior es la matriz de intersección de entrada - salida definida como:

$$N_{i,j}^{IO} = \mathrm{n}\left(C_i^I \cap C_j^O\right) \text{ where } 1 \le i \le M \text{ and } 1 \le j \le N \qquad (1)$$

donde n es el operador de cardinalidad. Esta matriz contiene en su *i,j*-ésima celda la cantidad de patrones de la clase de entrada C_i^I clasificados como pertenecientes a la clase de salida C_j^O.

Para medir en que grado un clasificador agrupa patrones pertenecientes a una clase de entrada en una misma clase de salida se define, en primer lugar, el coeficiente de concentración interclase para la clase de entrada C_i^I en las *N* clases de salida C_j^O como:

$$cc_i = \frac{N \max_{j=1}^{N}\left(N_{i,j}^{IO}\right) - \sum_{j=1}^{N} N_{i,j}^{IO}}{(N-1)\sum_{j=1}^{N} N_{i,j}^{IO}} \qquad (2)$$

Se define el *coeficiente de concentración interclase* para un clasificador como el promedio de los cc_i ponderados por la cantidad de patrones de la clase de entrada correspondiente:

$$cc = \frac{\sum_{i=1}^{M} N \max_{j=1}^{N}\left(N_{i,j}^{IO}\right) - \sum_{i,j=1}^{i=M,\,j=N} N_{i,j}^{IO}}{(N-1)\sum_{i,j=1}^{i=M,\,j=N} N_{i,j}^{IO}} \qquad (3)$$

Para medir la capacidad que posee un clasificador para llevar patrones de distintas clases de entrada a distintas clases de salida se define, en primer lugar, el coeficiente de dispersión intraclase para la clase de salida C_j^O en las *M* clases de entrada C_i^I como:

$$cd_j = \begin{cases} \dfrac{M \max_{i=1}^{M}\left(N_{i,j}^{IO}\right) - \sum_{i=1}^{M} N_{i,j}^{IO}}{(M-1)\sum_{i=1}^{M} N_{i,j}^{IO}} & \text{if} \quad \mathrm{n}\left(C_j^O\right) \neq 0 \\ 0 & \text{if} \quad \mathrm{n}\left(C_j^O\right) = 0 \end{cases} \qquad (4)$$

De forma similar que para el coeficiente de concentración, se define el *coeficiente de dispersión intraclase* para un clasificador como el promedio de los cd_j ponderados por la cantidad de patrones en cada clase de salida:

$$cd = \frac{\sum_{j=1}^{N} M \max_{i=1}^{M}\left(N_{i,j}^{IO}\right) - \sum_{j,i=1}^{j=N,\,i=M} N_{i,j}^{IO}}{(M-1)\sum_{j,i=1}^{j=N,\,i=M} N_{i,j}^{IO}} \qquad (5)$$

Métodos

Algoritmo de entrenamiento del ARNA

La totalidad de los patrones de entrenamiento es presentada inicialmente al nodo raíz. A los nodos de los niveles siguientes les llega un subconjunto de patrones que ha sido derivado jerárquicamente desde los niveles anteriores del árbol.

Considerando un nodo en particular se debe decidir si se justifica o no realizar una tarea de clasificación. Así, se distingue entre dos tipos de nodos: nodos clasificadores y nodos terminales. Para declarar que un nodo es terminal o clasificador se deben tener en cuenta dos características de su conjunto de patrones de entrada: el grado de homogeneidad en clases y el número de patrones que posee. Por esta razón se define el coeficiente de concentración para el conjunto de patrones de entrada como:

$$pc = \frac{M \max_{i=1}^{M}\left(\mathrm{n}(C_i)\right) - \mathrm{n}(X)}{(M-1)\,\mathrm{n}(X)} \qquad (6)$$

Para determinar el tipo de nodo en base a las características mencionadas se comparan sus medidas con dos umbrales: el umbral de concentración mínima de patrones de entrada (*upc*) y el umbral de cantidad mínima de patrones de entrada (*unX*). De esta forma se dice que un nodo es terminal cuando la concentración de patrones de entrada supera el umbral *upc* o cuando la cantidad de patrones es menor que el umbral *unX*.

Si se encuentra un nodo clasificador entonces éste debe entrenarse. La red neuronal que debe entrenarse para implementar la tarea de clasificación en el nodo es un SOM. La dimensión de entrada en esta red está determinada por la dimensión de los patrones y es la misma para todo el árbol. La dimensión o cantidad de clases de salida junto con los nodos terminales definen la topología final del árbol.

Para determinar la cantidad apropiada de clases de salida se utiliza un proceso de crecimiento de nodo basado en los coeficientes *cc* y *cd* y dos umbrales de capacidad de clasificación mínima *ucc* y *ucd* respectivamente. Se adopta

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
D. H. Milone, J. C. Sáez, G. Simón & H. L. Rufiner; "Arboles de redes neuronales autoorganizativas"
Anales del 1er Congreso Latinoamericano de Ingeniería Biomédica, Mazatlán 98, Vol. 1, pp. 779-784, Nov, 1998.

inicialmente una configuración con dos clases de salida (*N*=2), se entrena la red y se evalúa su desempeño en la clasificación. En el caso en que no se supere alguno de los umbrales se incrementa *N* en uno y se repite el entrenamiento y prueba. Este proceso culmina cuando ambos coeficientes superan sus correspondientes umbrales o cuando *N* alcanza el máximo permitido *maxN*. En este último caso, se elige la mejor de todas las configuraciones entre 2 y *maxN* y se considera concluido el entrenamiento de ese nodo. Este algoritmo de crecimiento de nodo se repite para todos los nodos de cada nivel del árbol y puede verse en la figura 1.

```
Para cada nivel del árbol
  Para cada nodo del nivel
    Nodo Terminal = (pc>upc) OR (n(X)<unX)
    Si no(Nodo Terminal) entonces
      N=minN
      Mientras no(Nodo Entrenado) hacer
        Crear Nodo
        Entrenar Nodo
        Probar Nodo

        Si N = maxN entonces
          Nodo Entrenado = verdadero
          Buscar Mejor N
          Si Mejor N <> maxN entonces
            Destruir Nodo
            N = Mejor N
            Crear Nodo
            Entrenar Nodo
          Si No
            Nodo Entrenado = (cc>ucc) AND (cd>ucd)

        Si no(Nodo Entrenado) entonces
          Destruir nodo
          N=N+1
  Actualizar los umbrales
```

Figura 1: Algoritmo para el crecimiento de ARNA.

Las exigencias en cuanto a concentración y dispersión varían de acuerdo al nivel de profundidad en el proceso de clasificación. En las primeras etapas de la clasificación se propone una mayor exigencia en cuanto a la concentración, mientras que la separación basada en detalles más finos se realiza progresivamente en niveles posteriores, en los que se exige mejor dispersión en la clasificación. Así se pasa gradualmente desde la no supervisión a la supervisión y se logra progresivamente la concordancia entre las clases de salida y las de entrada.

Cuando el árbol ha sido entrenado se procede al etiquetado de los nodos terminales. La elección de la etiqueta asignada a cada nodo terminal se realiza en base a la siguiente ecuación:

$$J_i = \arg\left[\max_{j=1}^{N}\left\{P(x_i \in C_j^O | x_i)\right\}\right] \qquad (7)$$

Los nodos terminales se unen, de acuerdo a su etiqueta, en otro nivel de nodos artificiales que poseen las etiquetas de todas las clases y de esta forma en el ARNA completo se cumple *M = N*.

Datos de entrenamiento

Las bases de datos empleadas en las pruebas del algoritmo fueron tomadas en su mayoría de [11]. En el trabajo citado se presenta una comparación entre varios paradigmas. Estos resultados se utilizarán para contrastar con los obtenidos en el presente trabajo. Se emplearon dos tipos de bases de datos: artificiales y reales. Dentro de las bases artificiales se uso la denominada **Clouds**. Entre las bases reales se empleo la base **IRIS** por su gran difusión [12]. Además se eligieron dos relacionadas con la clasificación de fonemas (**Peterson** [13] y **Phoneme**) debido a que esta aplicación es de interés central para nuestro grupo.

Resultados

Se presentan los resultados finales de las pruebas de validación para un conjunto de problemas de clasificación bien conocidos. El método de validación utilizado fue *hold out* promediado [11]. Para contrastar los resultados obtenidos con ARNA se muestran los resultados obtenidos con otras técnicas de clasificación aplicadas al mismo problema y una estimación del máximo desempeño de Bayes mediante el método *k-nearest neighbour* [11]. En las tablas 1 a 3 se compara el coeficiente de reconocimiento del ARNA con el de MLP, *learning vector quantization* (LVQ) y árboles de perceptrones simples (APS) extraídos del trabajo comparativo de [11]. La tabla 4 compara los resultados con MLP y LVQ a partir de los patrones de **Peterson**. Para tener una referencia de la velocidad de entrenamiento se entrenaron un MLP y un ARNA con la base de datos **Peterson**. El resultado final indica que el ARNA es 8.4 veces más rápido que el MLP, para un desempeño similar.

cr %	μ	Min	Max
MLP	95.78	93.33	98.61
LVQ	93.83	89.44	98.67
APS	93.33	86.61	98.61
ARNA	97.78	96.67	98.33
Bayes	96.66	94.72	97.27

Tabla 1: Resultados para **IRIS**.

cr %	μ	Min	Max
MLP	87.66	86.77	88.33
LVQ	87.66	85.66	88.55
APS	85.66	85.00	86.33
ARNA	83.93	82.20	87.20
Bayes	88.11	87.44	88.77

Tabla 2: Resultados para **Clouds**.

cr %	μ	Min	Max
MLP	83.63	82.36	84.69
LVQ	83.00	82.20	83.76
APS	83.47	82.04	85.06
ARNA	82.99	81.87	84.84
Bayes	87.7	86.81	88.02

Tabla 3: Resultados para **Phoneme**.

cr %	μ	Min	Max
MLP	83.08	79.55	84.85
LVQ	84.36	74.81	87.57
ARNA	86.36	84.85	87.88

Tabla 4: Resultados para **Peterson**.

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
D. H. Milone, J. C. Sáez, G. Simón & H. L. Rufiner; "Arboles de redes neuronales autoorganizativas"
Anales del 1er Congreso Latinoamericano de Ingeniería Biomédica, Mazatlán 98, Vol. 1, pp. 779-784, Nov, 1998.

Discusión

Si se comparan los resultados obtenidos con el ARNA y otros clasificadores se observa que el desempeño comparativo varía para las diferentes bases de datos. El ARNA tuvo un desempeño superior a todas las otras arquitecturas cuando se utilizó la base **IRIS** para el entrenamiento. El desempeño del ARNA para la base **Phoneme** fue prácticamente el mismo que el desempeño obtenido por el resto de las arquitecturas. Sin embargo, para la base **Clouds** el rendimiento del ARNA fue 3.72% inferior al obtenido por un MLP, que fue el clasificador que obtuvo el mejor resultado. En el caso de los patrones de **Peterson** los resultados favorecen al ARNA frente a LVQ y MLP.

La razón de la disminución comparativa de los resultados del ARNA se debe a una limitación técnica de la implementación del algoritmo. La máxima cantidad de SOMs que se pueden generar está limitada a 128 en la versión actual. Esta cantidad parece ser suficiente para bases simples como **IRIS** y **Peterson**, sin embargo no lo es para bases más complejas como **Clouds** y **Phoneme**.

Una consideración muy importante a la hora de realizar comparaciones entre diferentes arquitecturas es el hecho de que mientras el ARNA adapta su topología al problema en cuestión, otros métodos como LVQ y MLP necesitan que se especifique una configuración inicial, generalmente basada en la experiencia del usuario y refinada mediante prueba y error. De hecho los resultados para LVQ, MLP y APS que se encuentran en las tablas 1, 2, 3 corresponden a resultados alcanzados con la mejor configuración hallada después de numerosas pruebas.

Dado que los cómputos realizados para la generación de un ARNA son sencillos, esta arquitectura es considerablemente más veloz que otras estructuras con algoritmos más complejos. Sin embargo puede requerir más operaciones para solucionar un mismo problema que los métodos clásicos de generación de árboles como ID3. Su principal ventaja frente a éstos últimos la posibilidad de generar fronteras mucho más complejas.

Conclusiones

En el presente trabajo se presenta un algoritmo para el crecimiento de árboles de redes neuronales autoorganizativas y se compara su desempeño frente a otros métodos de clasificación con distintas bases de datos. La principal fuente de las ventajas de este método está en la combinación de diferentes paradigmas de clasificación, lo que permite aprovechar las ventajas de cada uno. El algoritmo planteado combina las ventajas del aprendizaje supervisado con las del aprendizaje no supervisado. Para el crecimiento y definición de la topología del árbol se utiliza información acerca de la identidad de los patrones. En cambio, para la tarea de clasificación en cada nodo se utiliza un extractor de características basado en un SOM que no usa información acerca de identidad de los patrones de entrenamiento. Otra de las combinaciones de paradigmas de clasificación que se encuentran en este algoritmo es la de los clasificadores simples y los jerarquizados. Mientras que la estructura general responde a los métodos de clasificación jerarquizada, en cada nodo se utiliza un típico clasificador simple como lo es una red neuronal. El ARNA es un clasificador muy flexible que no necesita la definición a priori de la topología (como se requiere en el caso del MLP). La velocidad del método es otra característica que hace del ARNA una configuración adecuada para la implementación de tareas de clasificación.

Actualmente se ha resuelto el problema generado por la limitación en el número de redes neuronales y los resultados serán objeto de un trabajo futuro, junto con los derivados de la implementación de nuevos clasificadores para reforzar los nodos con menor desempeño.

Referencias

[1] T. Kohonen, "*The Self-Organizing Map*", New York: Springer-Verlag, 1995.

[2] R. P. Lippmann, "*An introduction to computing with neural nets*", IEEE ASSP Mag., vol.4, no.2, 1987, pp 4-22.

[3] J.C. Goddard, F.M. Martínez, A.E. Martínez, J.M. Cornejo, H.L. Rufiner, R.C. Acevedo, "*Aprendizaje Maquinal en Medicina: Diabetes, un caso de Estudio";* Revista Mexicana de Ingeniería Biomédica vol.16, no.2, Octubre de 1995.

[4] L. Breiman, J.H.Friedman, R. A. Olshen, C. J. Stone, "*Classification and Regression Trees*", Wadsworth. Int. 1984.

[5] Sestito, Dillon, "*Automated Knowledge Acquisition*", Prentice Hall 1994.

[6] T. Kohonen, "*The Self-Organizing Map*", Proc.IEEE, vol.78, no.9, Sept.1990, pp 1464-1480

[7] A. Sankar, R. J. Mammone, "*Neural Tree Networks*", en "Neural Networks: Theory and Applications", R. Mammone and Y. Zeevi Eds., New York: Academic Press, 1991, pp 281-302.

[8] I. K. Sethi, "*Decision tree performance enhancement using an artificial neural network implementation"*, Artificial Neural Networks and Statitical Patter Recognition. Old and New Conecctions, Ed. I.K.Sethi, A.K.Jain, Elsevier Science Publishers B.V., 1991.

[9] M. G. Rahim, "*A self-learning neural tree network for phone recognition*", Artificial Neural Networks for Speech and Vision, Ed.R. J. Mammone, Chapman & Hall, 1993.

[10] C. Ke, Y. Xiang, CH. Huisheng, Y. Liping, "*Modular-Tree: A Self-Architecture Neural Network Architecture*", Report, Vol. 32, Nº 1 pp 110-119.

[11] Guerin-Dugue, A. and others, Deliverable R3-B4-P - Task B4: Benchmarks, Technical report, Elena-Nerves II "Enhanced Learning for Evolutive Neural Architecture", ESPRIT-Basic Research Project Number 6891, June 1995.

[12] UCI Repository of Machine Learning Databases and Domain Theories (ics.uci.edu: pub/machine-learning - databases).

[13] G. E. Peterson, H. L. Barney, "Control methods used in a study of the vowels", J. Acoust. Soc. Am. 24, 175-184, 1952.

sinc(*i*) Research Center for Signals, Systems and Computational Intelligence (fich.unl.edu.ar/sinc)
D. H. Milone, J. C. Sáez, G. Simón & H. L. Rufiner; "Arboles de redes neuronales autoorganizativas"
Anales del 1er Congreso Latinoamericano de Ingeniería Biomédica, Mazatlán 98, Vol. 1, pp. 779-784, Nov, 1998.