Sistema automático de ayuda a la evaluación práctica docente

A. Soriano Payá, F. Flórez Revuelta, J. Mora Pascual, A. Grediaga Olivo, A. Fuster Guilló
Dpto. Tecnología Informática y Computación
Universidad de Alicante
e-mail:soriano@dtic.ua.es

Resumen

En este artículo proponemos un sistema automático de ayuda a la corrección de prácticas relacionadas con lenguajes de programación como son el lenguaje C, C++, Pascal, Prolog, Informix, etc. Este sistema está formado por dos fases: extracción de características de la práctica entregada por el alumno y clasificación del vector de características asociado a la práctica. El clasificador, implementado mediante mapas auto-organizativos de Fritzke[1], nos permite distinguir qué prácticas son iguales y cuales no. Presentamos los resultados obtenidos, así como las posteriores conclusiones a las que hemos llegado.

1 Introducción

Ante el incremento en los nuevos planes de estudio de asignaturas con un mayor contenido práctico se ha hecho necesario depurar y refinar el sistema de evaluación desde el punto de vista práctico. En concreto, uno de los puntos a tener en cuenta es el control de posibles copias de los trabajos prácticos.

La carga teórica y práctica de las principales asignaturas de nuestro entorno de trabajo es la siguiente:

- Fundamentos de los Sistemas Operativos (4.5 teóricos y 1.5 prácticos);
- Diseño de los Sistemas Operativos (4.5 prácticos y 1.5 teóricos);
- Administración de los Sistemas Operativos (1.5 teóricos y 3 prácticos).

Como podemos ver el contenido práctico de las dos primeras asignaturas representa el 30% del contenido teórico, mientras que en la última asignatura el contenido práctico es el doble que el contenido teórico. Es por ello, que no hay que dejar de lado los contenidos prácticos y mucho menos su evaluación.

Dependiendo del tipo de trabajo práctico, su evaluación o corrección nos llevará más o menos tiempo. En el caso de corrección de trabajos prácticos que sean programas, independientemente del lenguaje de programación, por regla general tenemos que dedicarle bastante tiempo sobre todo si queremos verificar su funcionamiento con toda la casuística posible. Este problema se agrava aún más cuando en la asignatura hay matriculados una gran cantidad de alumnos. Con el fin de reducir este problema, vamos a proponer un sistema automático basado en redes neuronales que nos servirá como herramienta de ayuda en la evaluación de las prácticas. Con este sistema podemos distinguir qué prácticas son iguales y cuales no. De esta forma ya no será necesario corregir todas las prácticas sino aquellas que sean distintas ya que las prácticas que son iguales tendrán la misma nota. Además de reducir el tiempo en la corrección de prácticas podemos ver los alumnos que se han copiado y tomar medidas al respecto si fuera necesario.

En este artículo, en primer lugar describiremos el funcionamiento del sistema automático y analizaremos sus fases. A continuación, describiremos como lo hemos implementado y examinaremos un ejemplo de aplicación del sistema. Por último, presentaremos

las ventajas del sistema y propondremos líneas de investigación futura.

2 Descripción del sistema automático

En este apartado vamos a describir el funcionamiento y estructura del sistema automático.

El sistema tiene como entrada las prácticas (ficheros fuentes) entregadas por los alumnos y como salida información de similitud de las prácticas donde se indica el parecido y discrepancia de las prácticas analizadas. Con el fin de poder diferenciar los nombres de fichero entregados por los alumnos, éstos deberán tener una codificación que los identifique unívocamente.

El sistema está formado por dos fases o bloques:

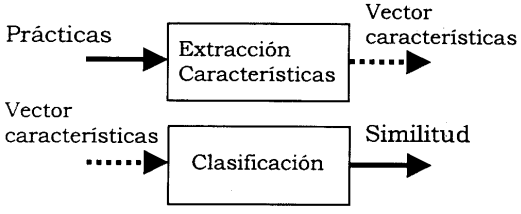

Figura 1. Sistema automático.

2.1 Extracción de características

La extracción de características consiste en dos subfases:

- Filtrado. Esta subfase elimina los comentarios introducidos por el alumno en su fichero fuente. Dependiendo del tipo de lenguaje de programación los comentarios se indican de una forma o de otra.
- Análisis. Esta subfase consiste en analizar diferentes tipos de ítems (instrucciones, operaciones, funciones, etc.) del fichero fuente entregado por el alumno de tal forma que la cantidad de tipos ítems encontrados darán lugar a las componentes del vector de características de esa práctica. Tanto la elección de ítems a analizar como la cantidad dependen del lenguaje de programación utilizado. Los tipos o clases de ítems a analizar deben ser representativos del lenguaje de programación y de la práctica en cuestión (condicionales, bucles, asignaciones, operaciones, funciones específicas del lenguaje, funciones específicas de la práctica, etc.).

Así, por ejemplo, si el lenguaje de programación fuera C los ítems que se podrían tener en cuenta serían:

Tipo	Ítem
Condicionales	if, switch
Bucles	for, while
Asignaciones	=
Operaciones aritméticas	+,-,/
Operaciones lógica	&&, ||
Funciones del lenguaje	return, exit
Funciones específicas del problema	malloc

Tabla 1. Ejemplos de ítems.

Esta construcción del vector de características es automática de tal forma que el sistema analiza práctica por práctica y construye el vector de características asociado. Así, por ejemplo, teniendo en cuenta los ítems de la tabla 1, el vector de características para una determinada práctica podría ser:

(9,5,10,18,2,3,2,t05a18e01)

del cual se deduce que en el fichero fuente se han encontrado 9 instrucciones condicionales, 5 instrucciones de bucle, 10 asignaciones, 18 operaciones aritméticas, 2 operaciones lógicas, 3 funciones del lenguaje, 2 funciones específicas del problema y nombre del fichero fuente t05a18e01.

2.2 Clasificación

Las redes neuronales artificiales son un intento de modelar las capacidades de procesamiento de la información de los sistemas nerviosos. Los sistemas nerviosos animales se componen de miles o millones de células interconectadas. Cada una de ellas se excita o se inhibe ante estímulos que le llegan de diferentes puntos de entrada y/o de células vecinas. Esta alta conexión de células individuales, cada una con una respuesta determinada, es la base de la emergencia de conciencia y comportamiento complejo. La mayor parte de los estudios se centran en observar cómo estos componentes individuales cooperan para formar sistemas masivamente paralelos de procesamiento de la información.

Existen diferentes modelos de redes neuronales artificiales dependiendo de la estructura en la están conectadas las neuronas, el algoritmo para su aprendizaje, etc. [4, 5]. Entre ellas destacan el perceptrón, las memorias asociativas y las redes auto-organizativas.

Los mapas auto-organizativos crecientes presentan grandes ventajas frente a otras redes clasificadoras e incluso frente a los mapas auto-organizativos originales como son la no definición a priori de la estructura de la red neuronal y la adaptatividad para aprender nuevos patrones una vez que el aprendizaje ya había finalizado [2, 3, 4].

Basándose en la capacidad de los mapas auto-organizativos crecientes de que una neurona responde de forma similar ante patrones de entrada parecidos y de que neuronas cercanas responderán a patrones que difieran poco, vamos a construir un mapa de las prácticas entregadas. De este modo, podremos obtener aquellas prácticas que son iguales (si responde la misma neurona) y aquellas que son muy parecidas (responden neuronas vecinas).

Antes de adiestrar el mapa es necesario normalizar las componentes de los vectores de características para que sus componentes afecten de forma relativa y no absolutamente al cálculo de distancias entre patrones [1].

3 Implementación y pruebas

El sistema automático propuesto lo hemos aplicado a las asignaturas de Fundamentos de los Sistemas Operativos y en Diseño de los Sistemas Operativos. A continuación vamos a describir tanto los la implementación del sistema como los experimentos realizados y los resultados obtenidos de la asignatura Fundamentos de los Sistemas Operativos, en concreto, de la práctica 4.

Una vez que los alumnos han entregado su práctica (fichero fuente) ya sea por correo, por ftp, o bien, mediante disco, tenemos que codificar, en el caso de que no se le hubiera pedido al alumno, el nombre del fichero fuente con el fin de poder identificar unívocamente cada práctica. La codificación que hemos utilizado hace referencia al turno y al grupo (2 alumnos por grupo). Dentro de cada turno hemos numerado los grupos. La codificación resultante es:

txxayyezz

donde *xx* indica el turno, *yy* indica el grupo y *zz* indica el ejercicio de la práctica entregada ya que en una de las prácticas hubo 5 ejercicios.

Una vez que las prácticas están codificadas ejecutamos el programa de extracción de características. Este programa leerá el fichero *comentarios.txt* que contiene los caracteres de comentarios y realiza un filtrado eliminando los comentarios del fichero fuente. A continuación leerá el fichero *items.txt* que contiene los ítems a analizar y generará un fichero llamado *vectores.txt* con los vectores de características. Como el lenguaje de programación de las prácticas que han realizado los alumnos es C, los ficheros que se utilizan en esta fase presentan la siguiente forma:

Fichero	Contenido
comentarios.txt	/* */ //
items.txt	if switch for while = + - / && || return exit malloc
vectores.txt	9 5 10 18 2 3 2 t00a00e00 7 3 11 20 2 3 3 t00a00e01 ..... 7 3 15 20 2 3 3 t17a20e05

Tabla 2. Ficheros fase de extracción.

A continuación ejecutamos el programa clasificador que leerá el fichero *vectores.txt*. El programa generará el fichero *resultado.txt* que contiene el parecido entre las prácticas. Este fichero presenta la siguiente forma:

Fichero	Contenido
resultado.txt	Neurona 0: Patrones: t00a00e01 t10a08e01 t09a12e01 Neurona 5 a 0.04 Neurona 9 a 0.59 Neurona 1: Patrones: t14a15e01 t11a18e01 Neurona 17 a 0.09 Neurona 89 a 0.19 Neurona 109 a 0.29 ..... Neurona 270: Patrones: t01a01e01 Neurona 55 a 0.94 Neurona 80 a 0.78 Neurona 234 a 0.89

Tabla 3. Fichero resultado.

Según el ejemplo de la tabla 3, la neurona 0 representa a tres patrones y está a distancia 0.04 de la neurona 5 y a distancia 0.09 de la neurona 9. Este dato indica que hay tres prácticas iguales (t00a00e01, t10a08e01 y t09a12e01) y que son bastante parecidas a las prácticas asociadas a la neurona 5 y menos parecidas a las prácticas asociadas a la neurona 9. También podemos ver que la neurona 270 representa a una sola práctica y esta a bastante distancia de las neuronas a las que está conectadas.

Después de aplicar el sistema automático a la práctica 4 de la asignatura de Fundamentos de los Sistemas Operativos obtuvimos que los siguientes resultados

- De los 258 grupos el sistema detectó 68 posibles copias entre grupos.
- Después de verificar de forma exhaustiva si en realidad los 68 grupos eran copias, observamos que 10 de ellos no lo eran.
- De los 68 grupos los profesores habían detectado visualmente sólo 16 de ellos.

4 Conclusiones

En este artículo hemos descrito un sistema automático basado en mapas auto-organizativos crecientes que permite clasificar las prácticas entregadas por los alumnos e inferir qué prácticas son iguales y cuales no. De esta forma podemos reducir el tiempo de corrección de las prácticas y además detectar a los alumnos que se han copiado.

Este sistema automático de ayuda presenta varias ventajas:

- permite ayudar al profesorado en la corrección de prácticas sobre todo si en la asignatura hay una gran cantidad de alumnos ya que reduce el tiempo empleado en la evaluación práctica.
- permite detectar posibles copias en las prácticas entregadas por los alumnos.
- siempre y cuando se le indique al alumno que se está utilizando un sistema de detección de copias, permite que el alumno ya no se copie y por tanto intente realizar la practica correspondiente.

Teniendo en cuenta que los futuros planes de estudio están orientados a reducir el número de asignaturas por curso y a su vez que sean anuales, el contenido práctico de las asignaturas aumentará considerablemente por lo que

será conveniente disponer de un sistema como este que permita reducir el tiempo de corrección y aumentar el control en la evaluación.

El siguiente paso será adaptar este sistema para que lo podamos utilizar como ayuda en la evaluación de trabajos teóricos entregados mediante soporte informático.

Referencias

[1] B. Fritzke. *Growing cell structures - a self-organizing network for unsupervised and supervised learning.* Neural Networks Vol. 7, No. 9. 1994.

[2] B. Fritzke. *Growing self-organizing networks - why?.* ESANN. 1996.

[3] T. Kohonen. *Self-Organization and Associative Memory.* Springer-Verlag. 1988.

[4] T. Kohonen. *Self-Organizing Maps.* Springer-Verlag. 1994.

[5] Dan W. Patterson. Artificial Neural Networks: Theory and Applicaciton. Prentice-Hall. 1996.