Análisis Comparativo de Tres Sistemas de la Programación Lógica Inductiva

Hilda Escobedo Hernández

Instituto de Investigaciones Eléctricas
escobedo@iie.org.mx

Resumen

En este artículo se presenta un análisis comparativo entre tres sistemas representativos de la Programación Lógica Inductiva(PLI): GOLEM, FOIL y PROGOL aplicados a la Minería de Datos. A la fecha se han reportado aplicaciones exitosas de la PLI, sin embargo, en la literatura especializada no aparece un análisis comparativo formal desde el punto de vista del usuario. En este sentido, en el presente trabajo se comparan dichos sistemas bajo tres dimensiones: la facilidad de uso del sistema, la estrategia de búsqueda y la facilidad de expresar un conocimiento previo. Para ejemplificar este estudio se utilizó una base de datos real. Los resultados experimentales mostraron, por una parte, que sistemas cumplen mejor en cada una de las dimensiones, y por otra, que no hay una dominación de un sistema sobre los otros. Finalmente, consideramos que la caracterización de los sistemas puede ser usada para la selección de un sistema, dada una aplicación especifica.

1. Introducción.

La mayoría de los sistemas de aprendizaje que se han aplicado a Minería de Datos están basados en la lógica proposicional y han demostrado su validez al aplicarlos sobre conjuntos de datos reales[16], sin embargo, presentan tres limitaciones importantes que restringen su campo de aplicación [4] a) usan limitado formalismo para la representación del conocimiento, b) tienen dificultad en incorporar en su proceso de aprendizaje un conocimiento previo, y c) tienen restricciones en su vocabulario. La primera restricción dificulta la representación de ciertos dominios que no pueden ser expresados de manera efectiva a nivel proposicional, la segunda es importante porque lo ideal en cualquier sistema de aprendizaje es que el conocimiento descubierto pueda ser almacenado y utilizado para descubrimientos posteriores, la tercera restricción se debe a que sería muy deseable que los sistemas realicen invención de nuevos predicados que ayudarán a definir mejor el conjunto de entrenamiento. Los sistemas de la PLI intentan vencer este tipo de limitaciones [11], utilizando un lenguaje de representación más potente: la lógica de predicados de

primer orden, y hacen uso de un conocimiento previo en su proceso de aprendizaje, además algunos sistemas tienen incorporada la facilidad para realizar invención nuevos predicados.

El Objetivo central de la PLI es construir un programa lógico que junto con conocimiento que se tenga del dominio, obtenga como consecuencia lógica conjunto de entrenamiento. Generalmente, el programa lógico que se construye forma de varias cláusulas hipótesis, por lo que de ahora en adelante le llamaremos teoría que denotaremos como Σ. Para encontrar a Σ, es necesario recorrer un espacio de búsqueda[12]. La dificultad de encontrar una Σ correcta esta determinado por tamaño del espacio de búsqueda y las propiedades de sus relaciones de generalidad que son consideradas en los sistemas de la PLI.

En este trabajo analizaremos las técnicas de generalización y las heurísticas utilizan los sistemas: GOLEM, FOIL y PROGOL para evitar tratar con espacios búsqueda complejos, lo cual nos permitirá mostrar las ventajas y limitaciones entre estos sistemas para encontrar a una Σ correcta. Según sea el método con el que recorre el espacio de posibles cláusulas, podemos distinguir varias técnicas: generalización relativa menos general bajo subsunción que es la que utiliza el sistema GOLEM [6], la búsqueda de grafos refinados que utiliza FOIL y la deducción inversa (en inglés, *inverse entailment*) que utiliza PROGOL.

En la sección 2, se presenta una descripción de cada sistema donde se explica como se obtiene a Σ y se explora la estrategia de búsqueda que ofrece mejores resultados entre estos sistemas. En la sección 3, se presenta una aplicación a Minería de Datos utilizando los tres sistemas bajo estudio en una Base de Datos real, la representa una muestra estadística de pacientes que han sufrido un infarto, en aplicación se pretende encontrar reglas que nos permitan identificar de acuerdo ciertas características de los pacientes cuales tienen probabilidades de sobrevivir.

2 Descripción de los Sistemas.

En la terminología de Base de Datos, los sistemas de PLI construyen definicione para una relación de la base de datos, fijada como relación objetivo o relación por el usuario. y para ello se utiliza el resto de predicados de la Base de Datos En esta sección se presentan las principales características de los sistemas estudio comparando los mecanismos de búsqueda que utilizan para encontrar teoría (Σ), que mejor explique al conjunto de entrenamiento, concluyendo con el ofrece mejores resultados.

2.1 GOLEM.

El sistema GOLEM utiliza una búsqueda *bottom-up* (de lo específico a lo general), sigue el método de generalización llamado Generalización Relativa Menos General bajo Subsunción (por sus siglas en ingles, RLGGs) [1]. Para evitar el problema de encontrar una RLGGs infinita el conocimiento previo es definido extensionalmente, es decir; por medio de un conjunto de hechos (cláusulas unitarias) [4]. La RLGGs construida por GOLEM, fue forzada a tener un número tratable de literales, requiriendo para esto que la posible cláusula hipótesis contenga solamente cláusulas definidas que sean *ij*-determinadas. Estas cláusulas tienen una restricción en la conexión de variables acotadas por los parámetros *i* y *j*. GOLEM acepta como entrada un conjunto de ejemplos positivos, ejemplos negativos y el conocimiento previo, todos estos están representados como hechos.

Para la inducción de la Σ [3], como primer paso elige aleatoriamente un par de ejemplos positivos y computa la RLGGs, usando el conocimiento previo. En el segundo paso, GOLEM vuelve a seleccionar aleatoriamente un ejemplo positivo y construye la RLGGs junto con la cláusula resultante del primer paso, eliminando los ejemplos positivos que satisfaga la hipótesis. El proceso del segundo paso continua mientras incrementalmente sean satisfechos los ejemplos positivos, en caso contrario, se repite el proceso desde el primer paso hasta que se cubran todos los ejemplos positivos o el criterio de detenerlo sea satisfecho. La cláusula de la RLGGs calculada es frecuentemente demasiado especifica y por lo tanto se generaliza eliminando literales del cuerpo de la cláusula hasta que se pueda ejecutar sin satisfacer ningún ejemplo negativo[12]. En GOLEM se introdujo el concepto de cláusulas *ij*-determinadas, que después se usaron en FOIL y PROGOL.

El sistema GOLEM usa un algoritmo de reducción *bottom-up*. Pero dicho algoritmo no garantiza una optimización en la Σ final, produciendo cláusulas dentro de esta que cubre ejemplos negativos. Una ventaja de este sistema es su facilidad de uso, solo se le deberá indicar el conjunto de ejemplos positivos, ejemplos negativos y el conocimiento previo en archivos por separado. La desventaja de utilizar este sistema es que solo puede usarse en dominios pequeños acotado por el número de cláusulas que pueden ser almacenadas (aproximadamente 10000 [3]). También resulta un proceso tedioso el de generar todos los hechos que representan al conocimiento previo, o quizá el usuario puede no conocerlo. Este sistema no tiene la facilidad de inventar nuevos predicados para una mejor definición de la relación objetivo, además este sistema es susceptible al ruido.

2.2 FOIL.

El sistema Foil (versión 1992), es considerado una extensión natural de algunas ideas de los sistemas proposicionales, para ser utilizado con la lógica de primer orden. FOIL es un sistema del tipo de búsqueda *top-down* (de lo general a lo especifico) usa la técnica de grafos refinados [7,12] (la principal técnica de especialización en la

PLI), para la cual emplea varias heurísticas entre ella la más usada es la llamada ganancia de información, la cual apoya una especialización bajo la capacidad de discriminar entre los ejemplos positivos y negativos. Similar al GOLEM requiere que el conocimiento previo sea definido extensionalmente. La forma en que son definidos los ejemplos positivos, ejemplos negativos y el conocimiento previo es por medio de tuplas. El proceso inductivo lo empieza con una cláusula muy general, la cual es gradualmente especializada agregando literales al cuerpo. La cabeza de la cláusula es siempre la literal de la relación objetivo con variables únicas para cada posición de sus argumentos. Por ejemplo: padre(X,Y):- ?, en está cláusula se representa relación objetivo "padre" la tarea de FOIL es buscar las literales del cuerpo de cláusula, eligiendo el nombre del predicado y las variables a usar. Si se introduce una nueva variable en una nueva literal, las tuplas se tienen que extender para incluir los valores de esa variable, esto se hace de manera automática por FOIL. El añadir una literal en el cuerpo de una cláusula sirve para dos propósitos [10], el primero, incrementar la proporción de ejemplos positivos cubiertos y el segundo servir para introducir nuevas variables necesarias para construir la Σ final. Las literales del primer tipo, se refieren a literales que son evaluadas usando la heurística información (la que obtenga mayor ganancia de información es la que se agrega), y segunda clase de literales son llamadas literales *ij*-determinadas que fueron utilizadas motivados por el éxito de GOLEM.

Los ejemplos negativos para FOIL, pueden ser proporcionados por el usuario indicarse que sean construidos automáticamente, de acuerdo a la suposición mundo cerrado, donde toda la información que no este presente explícitamente en ejemplos positivos son negativos. FOIL utiliza como lenguaje para expresar teorías un dialecto de Prolog e incluye algunas características adicionales [10] como son: heurística para la poda del espacio de búsqueda de literales, métodos para incluir igualdad y negación de fallo.

Una ventaja de usar este sistema es que es de fácil uso, pero el usuario deberá definir la siguiente información: todas las posibles constantes del conjunto entrenamiento, indicar cuales predicados representan a la relación objetivo y los representan al conocimiento previo, además deberá acotar los argumentos de predicados. Este sistema al igual que GOLEM, sólo puede utilizarse en dominios pequeños limitado por la capacidad de almacenamiento de las tuplas.

2.3 PROGOL.

El sistema Progol, es un sistema de tipo de búsqueda *top-down* [5]. Usa aproximación al problema general de la PLI llamado deducción inversa (en ingles, *inverse entailment*), cuenta con un interprete de Prolog estándar argumentado capacidades inductivas. El conocimiento previo puede ser definido de dos formas: extensionalmente e intensionalmente, es decir, por medio de un conjunto de hechos reglas respectivamente, permitiendo por medio de las reglas expresar un meta-

conocimiento del dominio, está es una ventaja muy importante comparada con los sistemas que aquí analizamos. Para usar este sistema, el usuario debe utilizar un lenguaje tipificado de expresiones de primer orden (cláusulas definidas) las cuales serán usadas como el "espacio de las hipótesis" [2]. Las expresiones son definidas usando "declaraciones de modo", en las cuales el usuario especifica que predicados serán usados en la cabeza y cuáles en el cuerpo de las cláusulas hipótesis.

Para la inducción de la Σ [5], el proceso consiste en seleccionar un ejemplo positivo e+ para ser generalizado, construyendo la cláusula más especifica llamada la *cláusula bottom* representada por "$\perp$", que debe implicar lógicamente al ejemplo seleccionado según las restricciones del lenguaje, tal que $B \wedge \perp \vdash e+$, donde B representa el conocimiento previo. Sobre esta cláusula PROGOL realiza una búsqueda *top-down* acotada por: "cláusula-vacía" $\leq H \leq \perp$ (donde H es la hipótesis a búscar). Para esta búsqueda, PROGOL lista todas las cláusulas que subsumen a la *cláusula bottom*, a partir de esta lista se realiza la búsqueda de la cláusula más general. En su estrategia de búsqueda utiliza el algoritmo A*, el cual es guiado por una medida de predicción/comprensión, la cláusula que ofrezca mejor puntuación es la que garantiza la máxima comprensión de los datos y es agregada a la base de conocimiento, todos los ejemplos que cubra esta cláusula son eliminados del conjunto de entrenamiento. El proceso continua desde el primer paso hasta que todos los ejemplos positivos sean satisfechos.

Las declaraciones de modos no solamente definen el lenguaje de las hipótesis sino también define la separación entre ejemplos (positivos y negativos) y el conocimiento previo. PROGOL, ofrece una serie de parámetros para controlar el proceso de generalización [14]. Este sistema tiene la capacidad de introducir nuevos predicados cuando sea necesario, es decir, puede proponer un predicado útil que no se encuentre ni en los ejemplos ni en el conocimiento previo y que describa mejor la relación objetivo.

En PROGOL, la técnica que utiliza para construir la cláusula bottom "$\perp$" para reducir el espacio de búsqueda ya existía en GOLEM, sin embargo con GOLEM se tiene que definir el conocimiento del dominio de manera extensionalmente mientras que en PROGOL puede ser definido intensionalmente. Aunque FOIL y PROGOL usan la misma dirección de búsqueda, PROGOL realiza una búsqueda sistemática acotada por: "cláusula-vacía" $\leq H \leq \perp$.

3 Resultados Experimentales.

En esta sección se presentan los resultados obtenidos al aplicar los sistemas: GOLEM, FOIL y PROGOL, utilizando una Base de Datos real, que contiene el

historial clínico de una muestra estadística de 120 pacientes que han sufrido un infarto en seis diferente lugares del corazón[1].

Para esta aplicación se pretende encontrar reglas que nos permitan identificar acuerdo a ciertas características de los pacientes cuales tienen probabilidades sobrevivir o no. Los resultados obtenidos se presentan en una tabla que permite comparar los resultados obtenidos con base al porcentaje de los ejemplos que lograron clasificarse dentro de las reglas encontradas y cuales de estos están clasificados correctamente.

3.1 Descripción de los Datos.

Objetivo: Encontrar las características de los pacientes que tienen riesgo de muerte pueden sobrevivir a causa de un infarto con base a las variables que se consideran, cuales son: edad, sexo, diabetes y localización del infarto.

Los atributos relevantes elegidos de la base de datos fueron:

1.- Nombre: Edad
Tipo : Nominal (existen tres grupos de edades: uno, dos, tres)
Descripción: Indica la edad del paciente. Grupo uno (1 a 24 años), grupo dos (25 55 años) grupo tres (56 en adelante)

2.- Nombre: Sexo
Tipo :Nominal (mas y fem)
Descripción: indica el sexo del paciente: masculino y femenino.

3.- Nombre: Diabetes
Tipo :Nominal (valores posibles: si, no)
Descripción: El valor si indica que el paciente tiene diabetes y no lo contrario.

4.- Nombre: locinf
Tipo :Nominal (existen seis lugares posibles: antsep, diaf, antlat, posinf, antext y sub)
Descripción: indica seis posibles lugares del corazón donde se puede localizar infarto.

Para esta aplicación se utilizaron los tres sistemas, los cuales se aplicaron con en el mismo objetivo planteado. Para la representación del conocimiento se utilizaron predicados que representaron a los ejemplos positivos, ejemplos negativos conocimiento previo. Para representar a los ejemplos, se utilizó el predicado llamado "evento", el cual representa las características que tiene el paciente que ha sufrido

[1] Esta muestra representativa se obtuvo en el historial clínico de pacientes del Hospital Civil de la ciudad de Xalapa, Veracruz.

infarto y el lugar del corazón donde este se produjo. Por ejemplo: evento(tres, mas, no, antext), significa que el paciente esta en el grupo de edad tres, es de sexo masculino, no tiene diabetes y sufrió el infarto en la parte del corazón llamada "antext".

Ejemplos positivos:

```
evento(tres, mas, no, antext).
evento(tres,fem,no,antsep).
evento(dos,fem,si,diaf).
evento(uno,mas,no,antlat).
evento(dos,mas,no,si,sub).
....
```

Para los ejemplos negativos se consideraron a todos los ejemplos que no son positivos y se utilizaron en cada tipo de sistema bajo estudio. Para representar el conocimiento previo o conocimiento del dominio se analizó la información contenida en la base de datos, donde se observó que se podían formar tres conjuntos de pacientes, representados por tres tipos de predicados. El primero representaría a los pacientes que no tienen riesgo de muerte a causa de un infarto, el segundo a pacientes que si lo tienen y el tercer conjunto representaría a los pacientes que con las mismas características algunas veces lograban sobrevivir y otras veces no. Para representar este conocimiento se utilizaron tres predicados a los que llamamos: clase_no, clase_si, y clase_nosesabe, los cuales nos representaban a cada conjunto respectivamente. Es importante señalar que en nuestra Base de Datos se tenían más características de los pacientes que pertenecían a la clase_no y clase_nosabe, por lo que los resultados obtenidos lograron mejor clasificación para estas. A continuación damos algunos ejemplos del conocimiento previo.

Conocimiento previo:

```
clase_no(tres,mas,si,diaf).
clase_no(dos, fem,no,sub).
clase_no(uno,fem,no, diaf).
....
clase_nosesabe(dos,fem,si,antext).
clase_nosesabe(dos,fem,no,antsep).
...
clase_si(dos,fem,si,antlat).
...
```

3.2 Resultados Obtenidos

A continuación presentamos las teorías Σ que se obtuvieron al utilizar los sistemas: GOLEM, FOIL y PROGOL.

GOLEM

```
evento(tres,mas,A,antlat).
evento(A,B,si,antsep).
evento(uno,mas,no,antext).
evento(dos,B,C,diaf).
evento(A,mas,no,diaf).
evento(tres,B,no,D):- clase_no(tres,B,no,D).
evento(uno,B,no,D):- clase_no(uno,B,no,D).
evento(A,B,no,antsep):- clase_nosesabe(A,B,no,antsep).
evento(dos,fem,si,D):- clase_si(dos,fem,si,D).
```

FOIL

```
evento(A,B,C):- C= antsep, A= uno.
evento(A,B,C):- C <> postinf, C<> sub, C<> antlat, C<> antext, B=mas.
evento(A,B,C):- C= antext, A<> uno, B=fem.
evento(A,B,C):- A<>dos, B=mas, C=antlat.
evento(A,B,C):- C= antsep, clase_nosesabe(A,B,D,C).
evento(A,B,C):- C=antlat, clase_no(A,B,D,C).
evento(A,B,C):- clase_no(A,B,D,C), B=mas.
evento(A,B,C):- clase_nosesabe(A,B,si,C).
evento(A,B,C):- clase_nosesabte(A,B,si,C)
```

PROGOL

```
evento(uno,mas,no,antext).
evento(dos,fem,sii,diaf).
evento(dos,fem,no,antext).
evento(tres,mas,si,antlat).
evento(tres,fem,si,antext).
evento(uno,mas,A,B):- clase_no(uno,mas,A,B).
evento(uno,fem,A,B):- clase_no(uno,fem,A,B).
evento(dos,mas,A,B):- clase_no(dos,mas,A,B).
evento(dos,fem,A,B):- clase_nosesabe(dos,fem,A,B).
evento(tres,mas,A,B):- clase_no(tres,mas,A,B).
evento(tres,fem,A,B):- clase_nosesabe(tres,fem,A,B).
```

En los resultados obtenidos se presenta reglas y excepciones. Las reglas nos permitirán la clasificación de los ejemplos y las excepciones representan a los ejemplos para los cuales no se logro una generalización. En la Tabla 1, se comparan los resultados obtenidos con base al porcentaje de los ejemplos que lograron clasificarse y de estos se representa el porcentaje de ejemplos clasificados correctamente.

Tabla 1. Comparación de resultados con base a ejemplos cubiertos y cubiertos correctamente.

Sistemas de la PLI	% Ejemplos Cubiertos	% Ejemplos Cubiertos Correctamente
GOLEM	52	73
FOIL	85	82
PROGOL	98	84

En la Tabla 1, los resultados demuestran que los sistemas FOIL y PROGOL ofrecen mejores resultados de clasificación (por el número de ejemplos clasificados), sin embargo, GOLEM obtiene resultados sin necesidad de que el usuario especifique un control en su proceso de generalización, aunque la teoría obtenida es más especifica. La teoría del sistema FOIL presenta reglas más generales que logran una mejor clasificación comparada con GOLEM, sin embargo, se requirió hacer varias pruebas acotando sus diferentes argumentos hasta lograr obtener resultados. En la aplicación sólo se obtuvieron resultados acotando la variable: localización de infarto. El sistema PROGOL encontró una teoría que permite clasificar mejor los ejemplos con base a las variables: edad y sexo, para obtener esta teoría se guío la búsqueda especificando en las "declaraciones de modo" utilizadas por este sistema, las variables de entrada fueron: edad y sexo, y las variables de salida: diabetes y localización del infarto, además se utilizaron varios parámetros para controlar el proceso de generalización, indicando: el número máximo de literales en la cláusula, profundidad de sus variables, utilización de ejemplos negativos y manejo de ruido, entre otros.

4 Conclusiones.

Con base en el análisis y la experimentación realizada se observó que el sistema PROGOL obtiene teorías con mejores resultados en términos del porcentaje de ejemplos clasificados y el porcentaje de ejemplos clasificados correctamente, esto se debe a que realiza una búsqueda sistemática acotada. Las facilidades de este sistema son por un lado, que el usuario puede proporcionarle por medio de reglas un meta conocimiento del dominio, favoreciendo con esto, que puede ser aplicado en dominios grandes (sin limitaciones) y por otro lado, que este sistema permite realizar invención de nuevos predicados para una mejor definición de la relación objetivo. Sin embargo, tiene como desventaja la dificultad de su uso, en el sentido de que el usuario debe conocer el tipo de patrones que pueden ser descubiertos en la base de datos para guiar la búsqueda y controlar el proceso de generalización por medio de varios parámetros que deberá proporcionar.

Por otra parte, los sistemas GOLEM y FOIL tienen más dificultad de encontrar teoría debido a su estrategia de búsqueda. En particular, las reglas encontradas FOIL son más generales por lo que logran una mejor clasificación. La ventaja utilizar estos sistemas es su facilidad de uso. La desventaja es que solo puede usarse en dominios pequeños, que se encuentran limitados a la capacidad almacenamientos de sus datos. Otra desventaja de estos sistemas es que requieren postprocesamiento en la teoría final por parte del usuario, ya que puede contener cláusulas que satisfacen ejemplos negativos o que no cubran ningún ejemplo.

Agradecimientos.

Agradezco al Dr. Luciano García Garrido de la Universidad de Matemáticas Habana, Cuba. Al Dr. Joaquin Peréz Ortega y Dr. Guillermo Rodriguez Ortiz, Instituto de Investigaciones Eléctricas por sus valiosos comentarios en el desarrollo este trabajo.

Referencias.

[1] Muggleton,S., Feng, C.: Efficient induction of logic program, the Turing Institute(1992).
[2] Muggleton,S., Firth, J.: Cprogol4.4 a tutorial introduction, Department of computer science, University of York. 2 (1994)
[3] Morales, E.:Learning chess patterns in Inductive Logic Programming, London, Academ Press, tesis ph (1992)
[4] Muggleton,S., De Raedt, L.:Inductive Logic Programming: Theory and Methods. Journal of Logic Programming (1994) 19/20:629-679.
[5] Muggleton, S.: Inverse Entailment and PROGOL, Oxford University computation Laboratory,United kingdom (1995).
[6] Nienhuys-Cheng,S.H., De Wolf, R.: Least Generalization and Greatest Specializations of sets of clauses pp.341-363 (1996).
[7] Quinlan,J.R.,Cameron-Jones,R.M: Induction of Logic Programs: FOIL and related System (1992).
[8] Piatetsky-Shapiro,G.,William, F.J.: Knowledge Discovery in Databases (1991) pp. 1-27.
[9] Jorg-Uwe, K., Wrobel, S: Controlling the complexity of Learning in Logic through syntactic and task-oriented models (1992).
[10] Cameron_Jones, R.,M., Quinlan R.,J.: Efficient top-down Induction of Logic Programs, University of Sydney,Australia. (1996).
[11] Brantko,I.,King,R.: Applications of inductive Logic Programming, Bulletin Sigart, volumen 5, number 1, (1994).
[12] Nienhuys-Cheng,S.H., De Wolf: Foundations of Inductive Logic Programming, LNAI-tutorial, 1228. Springer-Verlag, Berlin (1997).
[13] Muggleton, S.: Inductive Logic Programming, derivations, successes and shortcomings. Bulletin Sigar, volumen 5, number 1, (1994).
[14] Sam,R.: An Introduction to Progol, (1997).
[15] Blockeel, H., De Raedt, L.: Relational Knowdge Discovery in Databases, Department Computer Science. (1996).
[16] Fayyad,M.,U,Piatetsky-Shapiro,G.,Smyth,P.,Uthurusamu,R.: Advances in Knowledge Discovery and Data Mining, AAAI Press/The Mit Press (1997).