Tema 5

Aprendizaje inductivo. Adquisición de conceptos

Ciencias de la Computación e Inteligencia Artificial
Universidad de Huelva

noviembre 2021

Índice

1. Introducción
2. Análisis de diferencias
   - Winston
     - Redes semánticas
     - Método de cotejamiento
     - Generalización
     - Especialización
3. Espacio de versiones
   - Find-S
   - List-Then-Eliminate
   - Eliminación de candidatos
4. Bibliografía

Índice

1. Introducción
2. Análisis de diferencias
   - Winston
     - Redes semánticas
     - Método de cotejamiento
     - Generalización
     - Especialización
3. Espacio de versiones
   - Find-S
   - List-Then-Eliminate
   - Eliminación de candidatos
4. Bibliografía

Definición

- El **aprendizaje inductivo** consiste en inducir información de una concepto a partir de un conjunto de cosas en concreto. No requiere información previa del dominio.
- Ejemplo: ese gato tiene 4 patas $\rightarrow$ todos los gatos tienen 4 patas.
- Si veo 1 caso supongo que todos los casos son iguales hasta que encuentre una contradicción que me obligue a remodelar las informaciones.

Introducción

- **Aprendizaje inductivo simbólico**
  - Utiliza una representación simbólica (redes semánticas, reglas, programación lógica, ...)
- **Aprendizaje inductivo subsimbólico**
  - Utiliza una representación subsimbólica (conjuntos difusos,...)
  - Se desarrolla por medio de algoritmos de ajuste paramétrico

Introducción

Formas de aprendizaje inductivo simbólico

- Si los ejemplos reflejan situaciones con múltiples objetos y relaciones
  - **Adquisición de conceptos**
- Si los ejemplos se refieren a conjuntos atributo-valor
  - **Clasificación supervisada**
- Si se pretende adquirir un modelo lógico
  - **Programación Lógica Inductiva**

Índice

1. Introducción
2. Análisis de diferencias
   - Winston
     - Redes semánticas
     - Método de cotejamiento
     - Generalización
     - Especialización
3. Espacio de versiones
   - Find-S
   - List-Then-Eliminate
   - Eliminación de candidatos
4. Bibliografía

Adquisición de conceptos

- Surge en la década de los 70
- Se produce un declive de la aproximación neuronal
- Aparecen los primeros lenguajes de manipulación simbólica
- La investigación se centra en problemas de juegos
- Se intentan obtener mecanismos de aprendizaje simbólico generales con muy poca información de partida
- **Algoritmos propuestos**
  - Winston (1970), Hayes-Roth (1977), Vere (1975), Michalski-Dietterich (1981)
  - Mitchell (1982) propone un marco unificado (espacio de versiones)

Método de Winston

- Desarrollado a lo largo de la década de los 70
- Se considera el punto de partida del **Aprendizaje Basado en Similitudes** (SBL)
- El autor lo denomina **Learning by analyzing differences**
- Introduce el concepto de **quasi-ejemplo** (ejemplo negativo muy próximo a los ejemplos positivos)
- Utiliza como representación las redes semánticas tanto para los ejemplos como para los conceptos
- Se centra en el dominio de aplicación del mundo de bloques

Quasi-Ejemplo

Un Quasi-Ejemplo es un ejemplo **NEGATIVO** que se diferencia en **UNA SOLA PROPIEDAD** con el **MODELO**

Método de Winston

Elementos del algoritmo:

- Un lenguaje de **representación**: **redes semánticas**
- Un mecanismo de **cotejamiento**: Método para análisis de diferencias
- Un proceso de **generalización**: para incorporar ejemplos positivos al modelo
- Un proceso de **especialización**: para rechazar ejemplos negativos (*quasi-ejemplos*) con el modelo

Método de Winston

Elementos del algoritmo:

- Un lenguaje de **representación**: **redes semánticas**
- Un mecanismo de **cotejamiento**: Método para análisis de diferencias
- Un proceso de **generalización**: para incorporar ejemplos positivos al modelo
- Un proceso de **especialización**: para rechazar ejemplos negativos (*quasi-ejemplos*) con el modelo

Método de Winston

Elementos del algoritmo:

- Un lenguaje de **representación**: **redes semánticas**
- Un mecanismo de **cotejamiento**: Método para análisis de diferencias
- Un proceso de **generalización**: para incorporar ejemplos positivos al modelo
- Un proceso de **especialización**: para rechazar ejemplos negativos (*quasi-ejemplos*) con el modelo

Método de Winston

Elementos del algoritmo:

- Un lenguaje de **representación**: **redes semánticas**
- Un mecanismo de **cotejamiento**: Método para análisis de diferencias
- Un proceso de **generalización**: para incorporar ejemplos positivos al modelo
- Un proceso de **especialización**: para rechazar ejemplos negativos (*quasi-ejemplos*) con el modelo

Redes semánticas

Descripción:

- Una red semántica o esquema de representación en Red es una forma de representación del conocimiento lingüístico
- Los conceptos y sus interrelaciones se representan mediante un grafo. En caso de que no existan ciclos, estas redes pueden ser visualizadas como árboles.
- En un grafo o red semántica los elementos semánticos se representan por nodos y las relaciones por aristas.
- Las redes semánticas son usadas, entre otras cosas, para representar mapas conceptuales y mentales.

Redes semánticas

Descripción:

- Una red semántica o esquema de representación en Red es una forma de representación del conocimiento lingüístico
- Los conceptos y sus interrelaciones se representan mediante un grafo. En caso de que no existan ciclos, estas redes pueden ser visualizadas como árboles.
- En un grafo o red semántica los elementos semánticos se representan por nodos y las relaciones por aristas.
- Las redes semánticas son usadas, entre otras cosas, para representar mapas conceptuales y mentales.

Redes semánticas

Descripción:

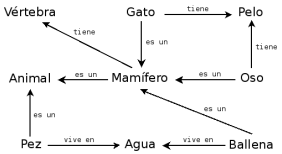
- Una red semántica o esquema de representación en Red es una forma de representación del conocimiento lingüístico
- Los conceptos y sus interrelaciones se representan mediante un grafo. En caso de que no existan ciclos, estas redes pueden ser visualizadas como árboles.
- En un grafo o red semántica los elementos semánticos se representan por nodos y las relaciones por aristas.
- Las redes semánticas son usadas, entre otras cosas, para representar mapas conceptuales y mentales.

Redes semánticas

Descripción:

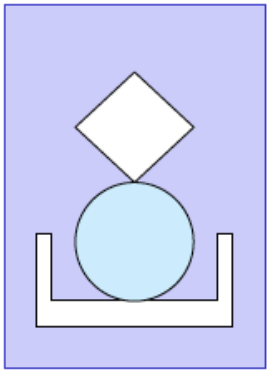
- Una red semántica o esquema de representación en Red es una forma de representación del conocimiento lingüístico
- Los conceptos y sus interrelaciones se representan mediante un grafo. En caso de que no existan ciclos, estas redes pueden ser visualizadas como árboles.
- En un grafo o red semántica los elementos semánticos se representan por nodos y las relaciones por aristas.
- Las redes semánticas son usadas, entre otras cosas, para representar mapas conceptuales y mentales.

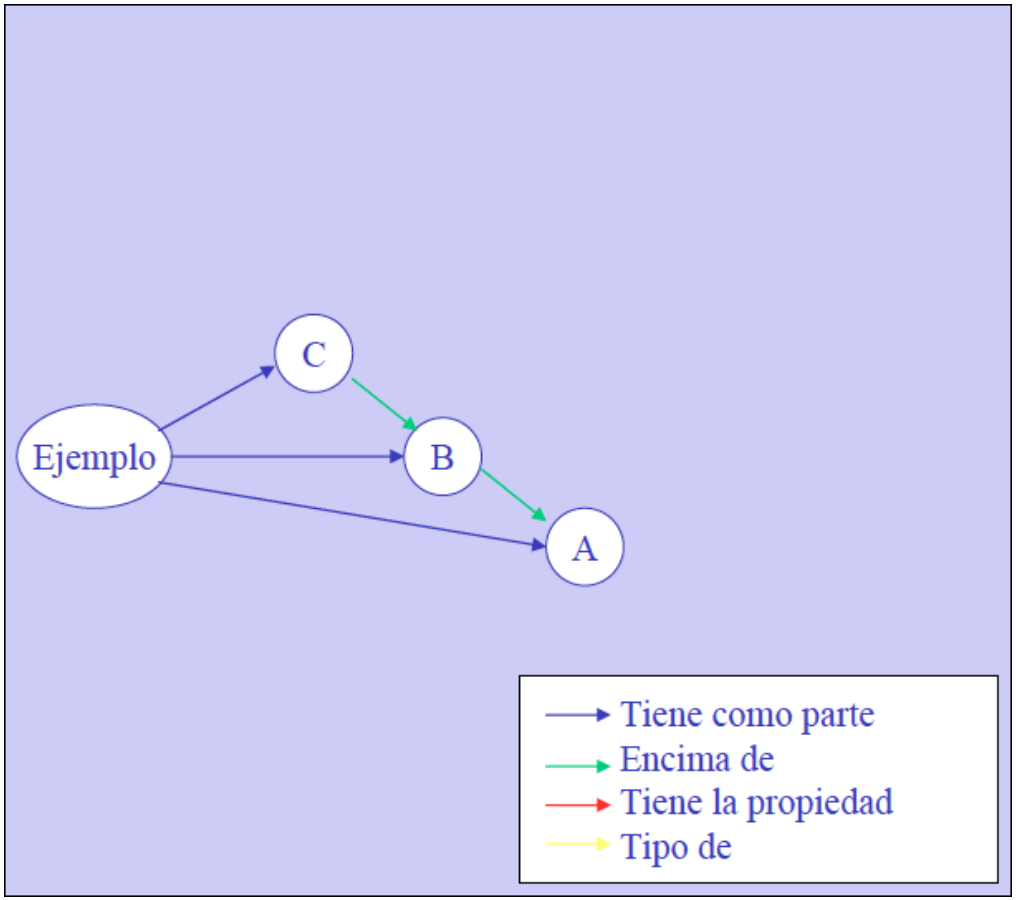
Redes semánticas



Redes semánticas

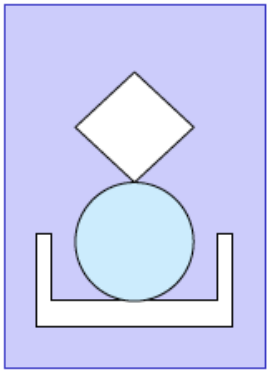
Ejemplo de construcción de una red semántica.

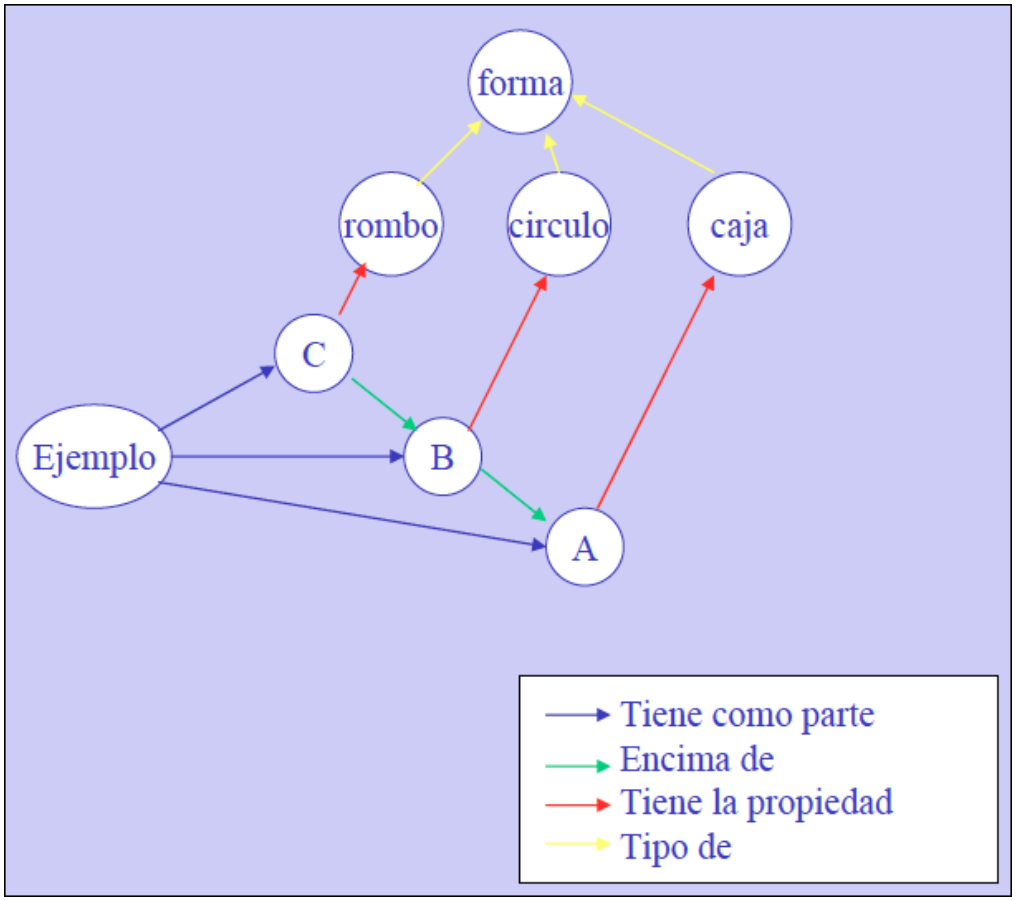




Redes semánticas

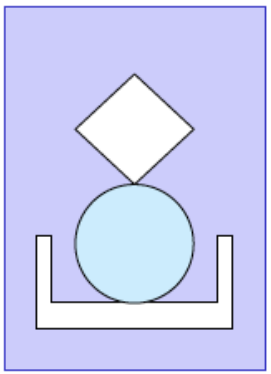
Ejemplo de construcción de una red semántica.

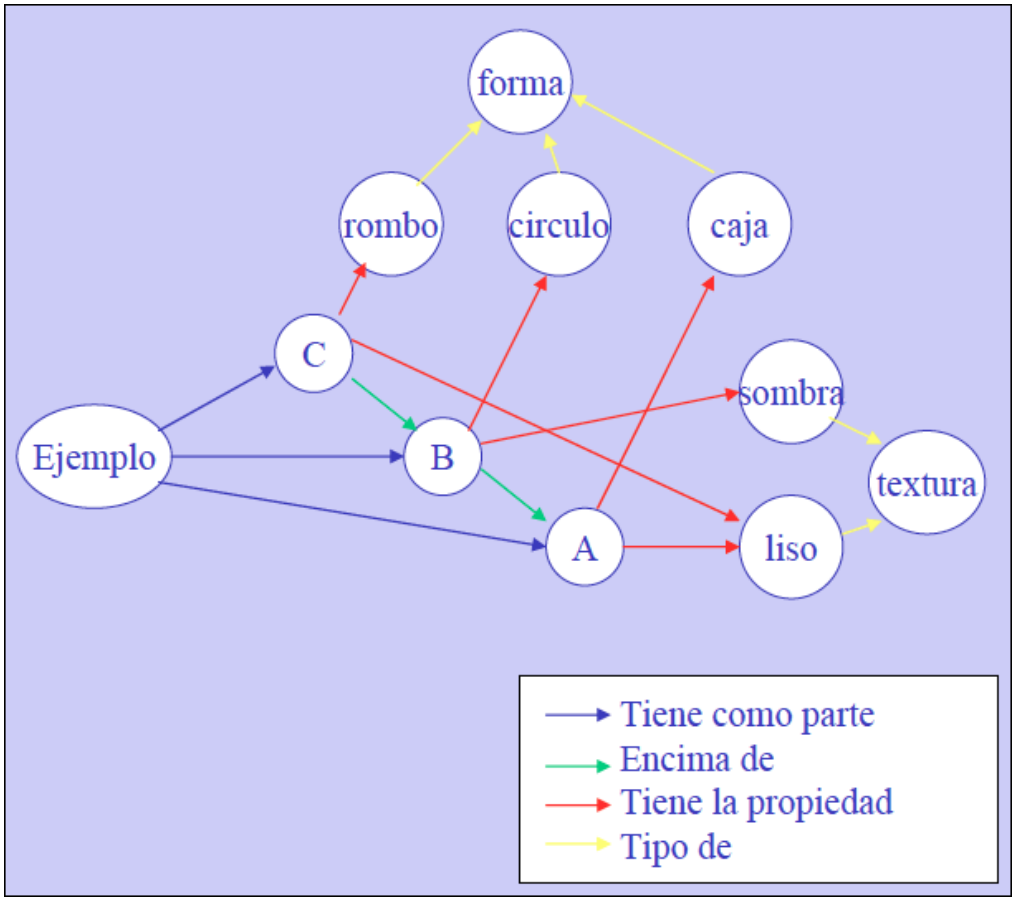




Redes semánticas

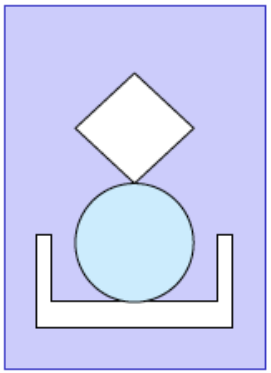
Ejemplo de construcción de una red semántica.

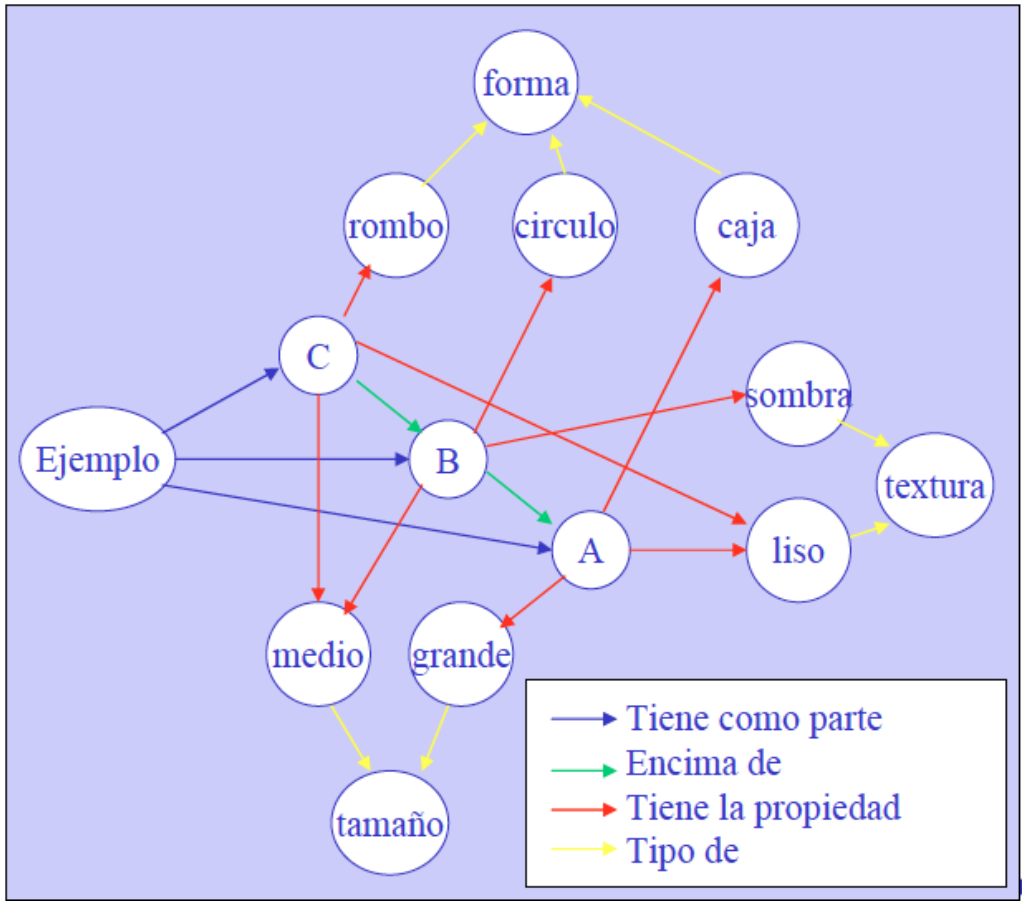




Redes semánticas

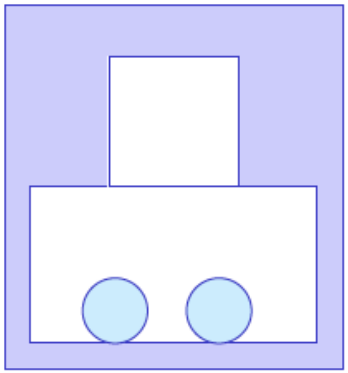
Ejemplo de construcción de una red semántica.





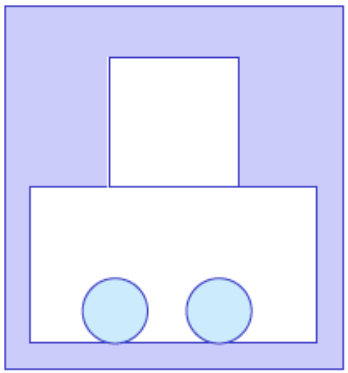
Redes semánticas

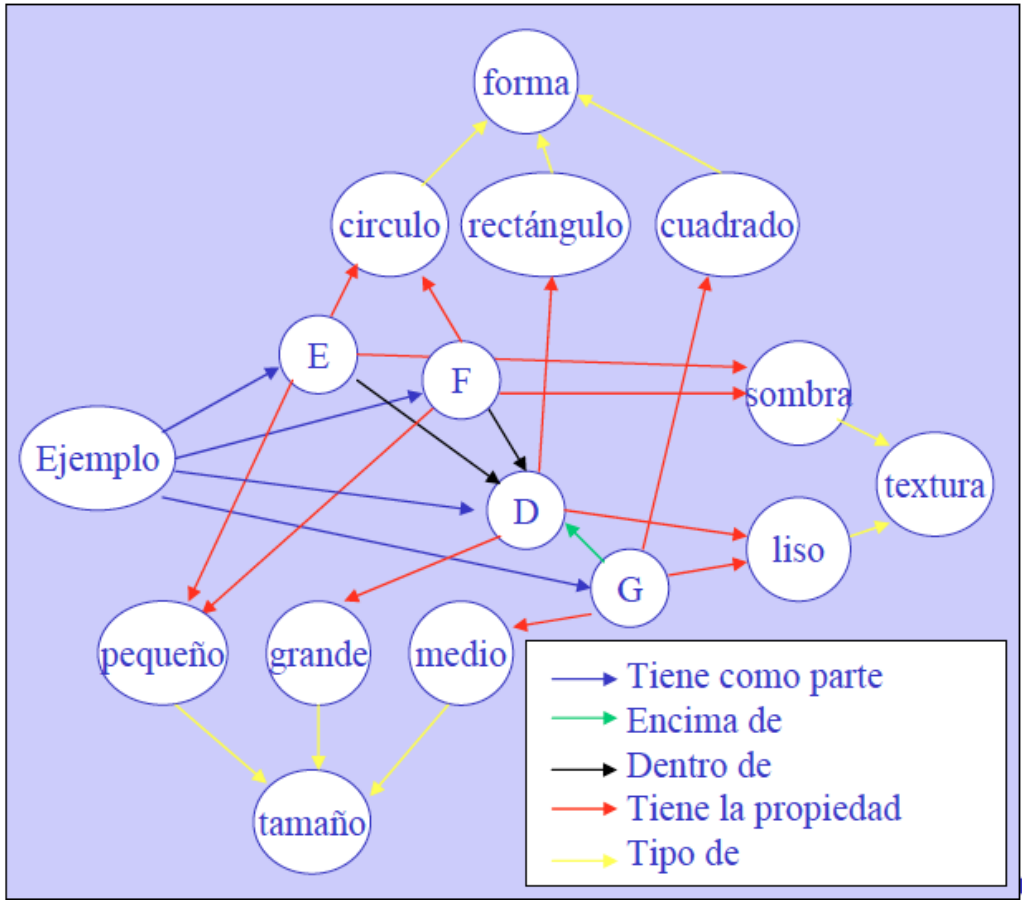
Ejemplo de construcción de una red semántica.



Redes semánticas

Ejemplo 2





Método de cotejamiento

- Permite comparar dos redes semánticas.
  Corresponden a la definición actual del concepto y a la definición del nuevo ejemplo a tratar.
- Según sea el ejemplo positivo o negativo, se dirijirá la búsqueda a generalizar o especializar la definición.
- Sólo se pueden modificar las etiquetas de las relaciones entre nodos, nunca los nodos. (Aunque si agregar).
- Se introducen dos heurísticas, en la búsqueda, sobre las etiquetas: **require-link** y **forbid-link**.

Método de cotejamiento

- Permite comparar dos redes semánticas.
  Corresponden a la definición actual del concepto y a la definición del nuevo ejemplo a tratar.
- Según sea el ejemplo positivo o negativo, se dirijirá la búsqueda a generalizar o especializar la definición.
- Sólo se pueden modificar las etiquetas de las relaciones entre nodos, nunca los nodos. (Aunque si agregar).
- Se introducen dos heurísticas, en la búsqueda, sobre las etiquetas: **require-link** y **forbid-link**.

Método de cotejamiento

- Permite comparar dos redes semánticas.
  Corresponden a la definición actual del concepto y a la definición del nuevo ejemplo a tratar.
- Según sea el ejemplo positivo o negativo, se dirijirá la búsqueda a generalizar o especializar la definición.
- Sólo se pueden modificar las etiquetas de las relaciones entre nodos, nunca los nodos. (Aunque si agregar).
- Se introducen dos heurísticas, en la búsqueda, sobre las etiquetas: **require-link** y **forbid-link**.

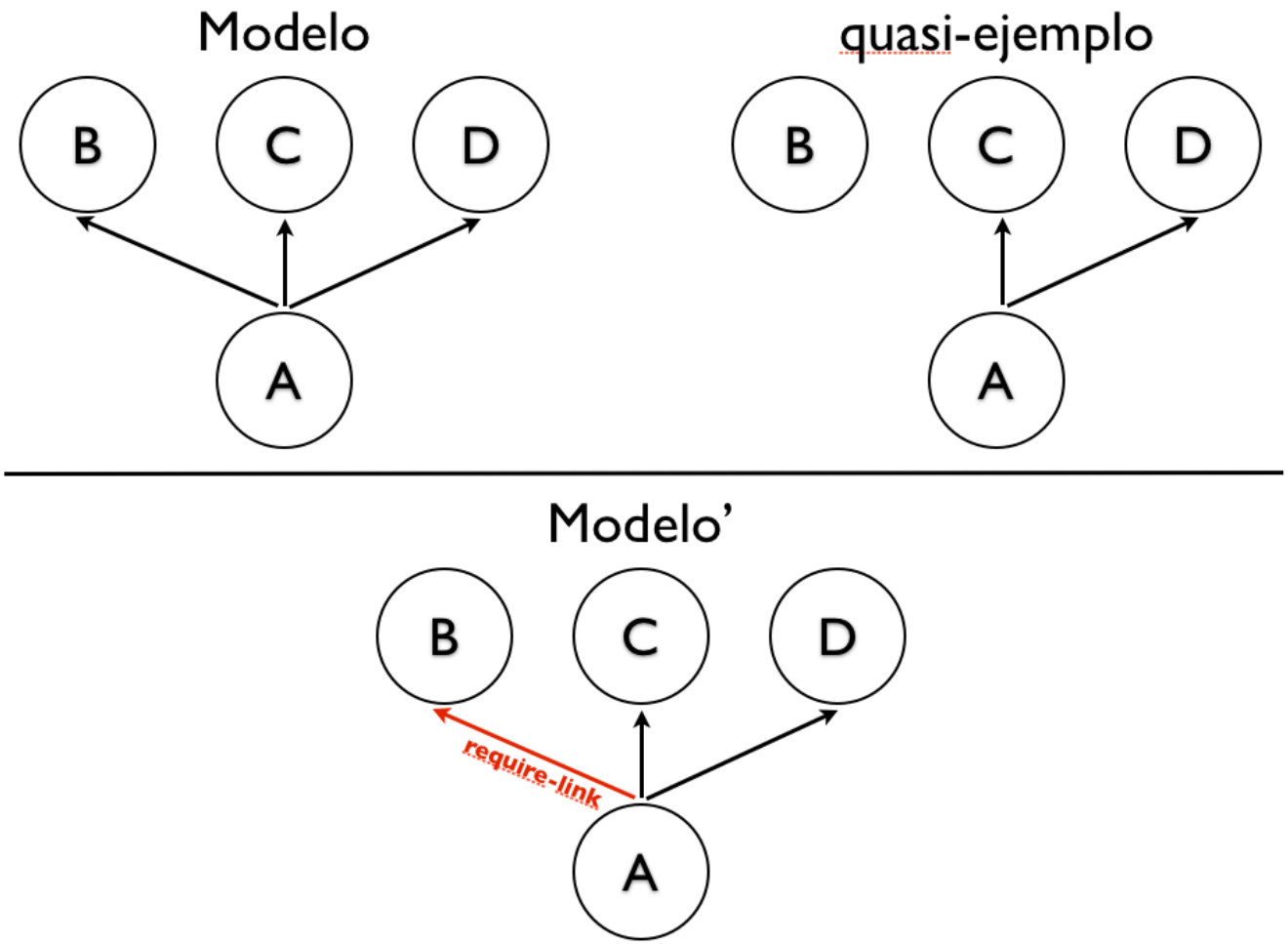
Método de cotejamiento

- Permite comparar dos redes semánticas.
  Corresponden a la definición actual del concepto y a la definición del nuevo ejemplo a tratar.
- Según sea el ejemplo positivo o negativo, se dirijirá la búsqueda a generalizar o especializar la definición.
- Sólo se pueden modificar las etiquetas de las relaciones entre nodos, nunca los nodos. (Aunque si agregar).
- Se introducen dos heurísticas, en la búsqueda, sobre las etiquetas: **require-link** y **forbid-link**.

Método de cotejamiento

- **Require-link**: es una heurística empleada cuando el modelo del concepto que está siendo aprendido (en evolución) tiene una etiqueta *k* en un lugar donde un quasi-ejemplo no. Entonces en la red semántica que representa el concepto esa etiqueta se transforma en **debe** (must).

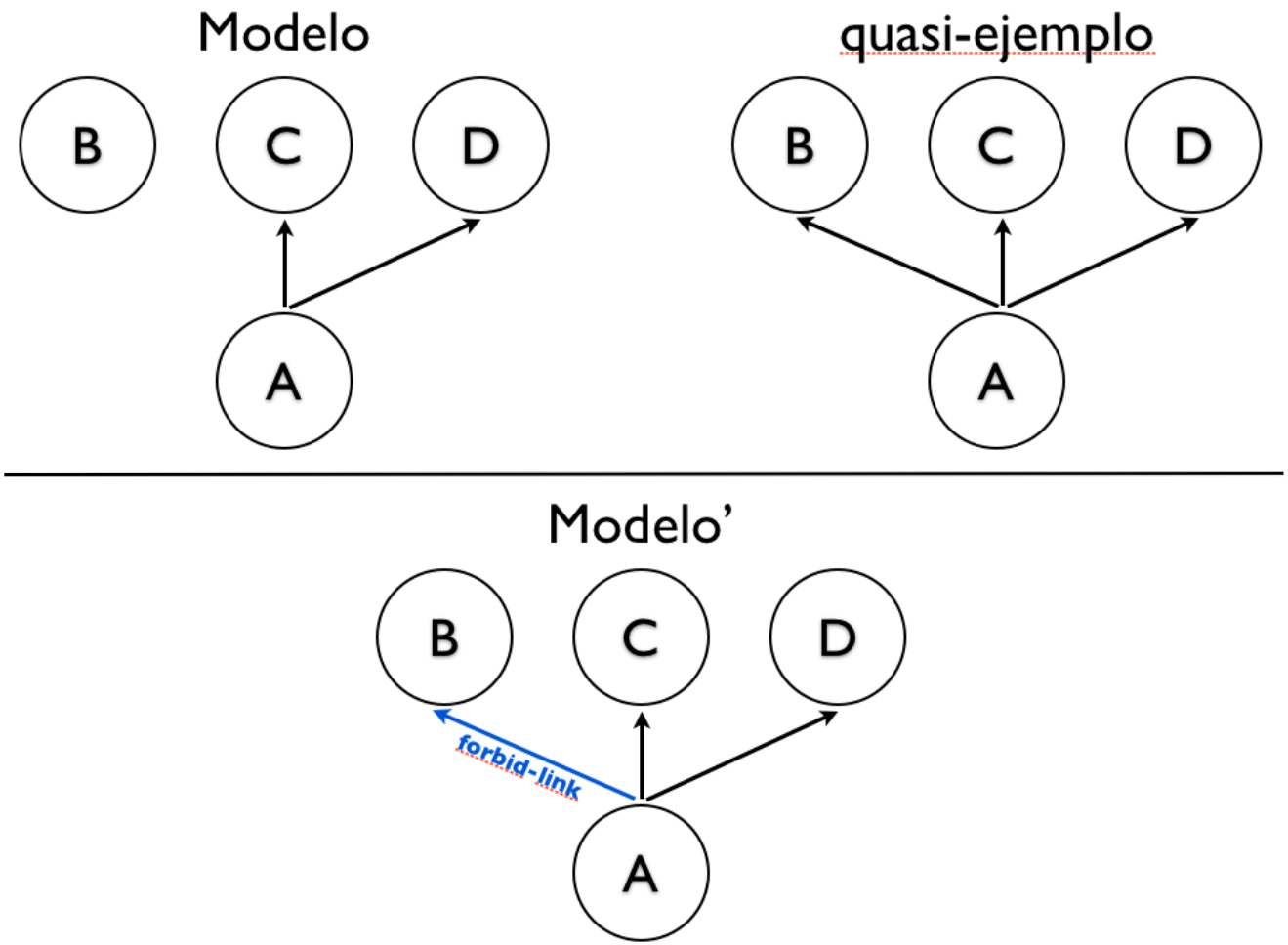
Required-link



Método de cotejamiento

- **Forbidden-link**: esta heurística se aplica cuando un quasi-ejemplo tiene una etiqueta *i* en un lugar donde el modelo no. Entonces una etiqueta **no-debe** (must-not) se coloca en el modelo actual del concepto.

Forbidden-link



Proceso de generalización

Cotejar el ejemplo (positivo) con el modelo actual del concepto.

- Procesar todas las diferencias
  - Si falta una etiqueta, eliminarla del concepto
  - Si hay diferencia en el valor de una propiedad, modificar el rango de la propiedad
  - Si una etiqueta apunta a una clase diferente
    - Si la clase pertenece a una jerarquía, subir en la jerarquía
    - Si la clase no pertenece a una jerarquía, eliminarla

Proceso de especialización

Cotejar el ejemplo (negativo) con el modelo actual del concepto

- Si hay más de una diferencia, ignorar el ejemplo
- Si hay una única diferencia:
  - Si el modelo tiene una etiqueta que el ejemplo no tiene, generar una etiqueta “required-link”
  - Si el ejemplo tiene una etiqueta que el concepto no tiene, generar una etiqueta “forbidden-link”

Algoritmo Winston v1

I. Tomar como modelo inicial la descripción de la primera instancia positiva del concepto.
Llamar a esta descripción la **definición del concepto**.

II. Examinar la descripción de otras instancias positivas conocidas del concepto. **Generalizar** la definición del concepto para incluirlas.

III. Examinar las descripciones de los *quasi-ejemplos* del concepto. **Especializar** la definición del concepto para excluirlos.

IV. Los pasos 2 y 3 se van intercalando a medida que se van tratando ejemplos positivos y negativos del concepto.

Propiedades del algoritmo

- Conservador: si existen dudas sobre lo que hay que aprender, mejor no aprender
- El aprendizaje se realiza en pasos pequeños (ley de Martin, no puedes aprender algo a menos que casi lo sepas de antemano)
- El algoritmo se ha descrito de manera no determinista ya que en cualquier punto puede haber varias posibles generalizaciones o especializaciones aplicables.
- La elección no necesariamente llevará a la hipótesis más sencilla; incluso puede ocurrir que lleguemos a una situación en que ninguna modificación sencilla de la hipótesis la haga consistente con todos los ejemplos. En ese punto será necesario un backtracking (vuelta atrás) al punto de elección

Dificultades del algoritmo de Winston

- hay que comprobar que cada modificación es consistente con todos los ejemplos
- Es muy sensible al orden en el que se presentan los ejemplos
- es difícil encontrar una buena heurística para elegir la mejor modificación y el backtracking es muy costoso (algoritmo del espacio de versiones de Mitchell)
- Ya a finales de los 70 el algoritmo de Winston se aplicó en Meta-Dendral, un sistema para la predicción de resultados de espectometría. Generó conocimiento que fue publicado en una revista de química analítica

Índice

1. Introducción
2. Análisis de diferencias
   - Winston
     - Redes semánticas
     - Método de cotejamiento
     - Generalización
     - Especialización
3. Espacio de versiones
   - Find-S
   - List-Then-Eliminate
   - Eliminación de candidatos
4. Bibliografía

Espacio de versiones

- Propuesto por Mitchell en 1982
- Presenta un marco unificado para la adquisición de conceptos, independiente de la representación a utilizar
- El objetivo es el mismo que en el esquema anterior, es decir, producir una descripción de un concepto a partir de un entrenamiento con ejemplos positivos y negativos.
- No se ve afectado por el orden en que se presentan los ejemplos.
- en lugar de describir un único concepto este esquema mantiene un conjunto de descripciones posibles hasta arribar a la definición del mismo.

Nomenclatura

- **Hipótesis**: representación de un concepto
- **Espacio de hipótesis**: el conjunto de todos los conceptos que pueden ser descritos con la representación escogida
- **Espacio de versiones**: Conjunto de hipótesis coherentes con el conjunto de ejemplos positivos y negativos estudiado, es decir, que reconocen a todos los ejemplos positivos y excluyen a todos los ejemplos negativos.

Nomenclatura

- **Hipótesis**: representación de un concepto
- **Espacio de hipótesis**: el conjunto de todos los conceptos que pueden ser descritos con la representación escogida
- **Espacio de versiones**: Conjunto de hipótesis coherentes con el conjunto de ejemplos positivos y negativos estudiado, es decir, que reconocen a todos los ejemplos positivos y excluyen a todos los ejemplos negativos.

Nomenclatura

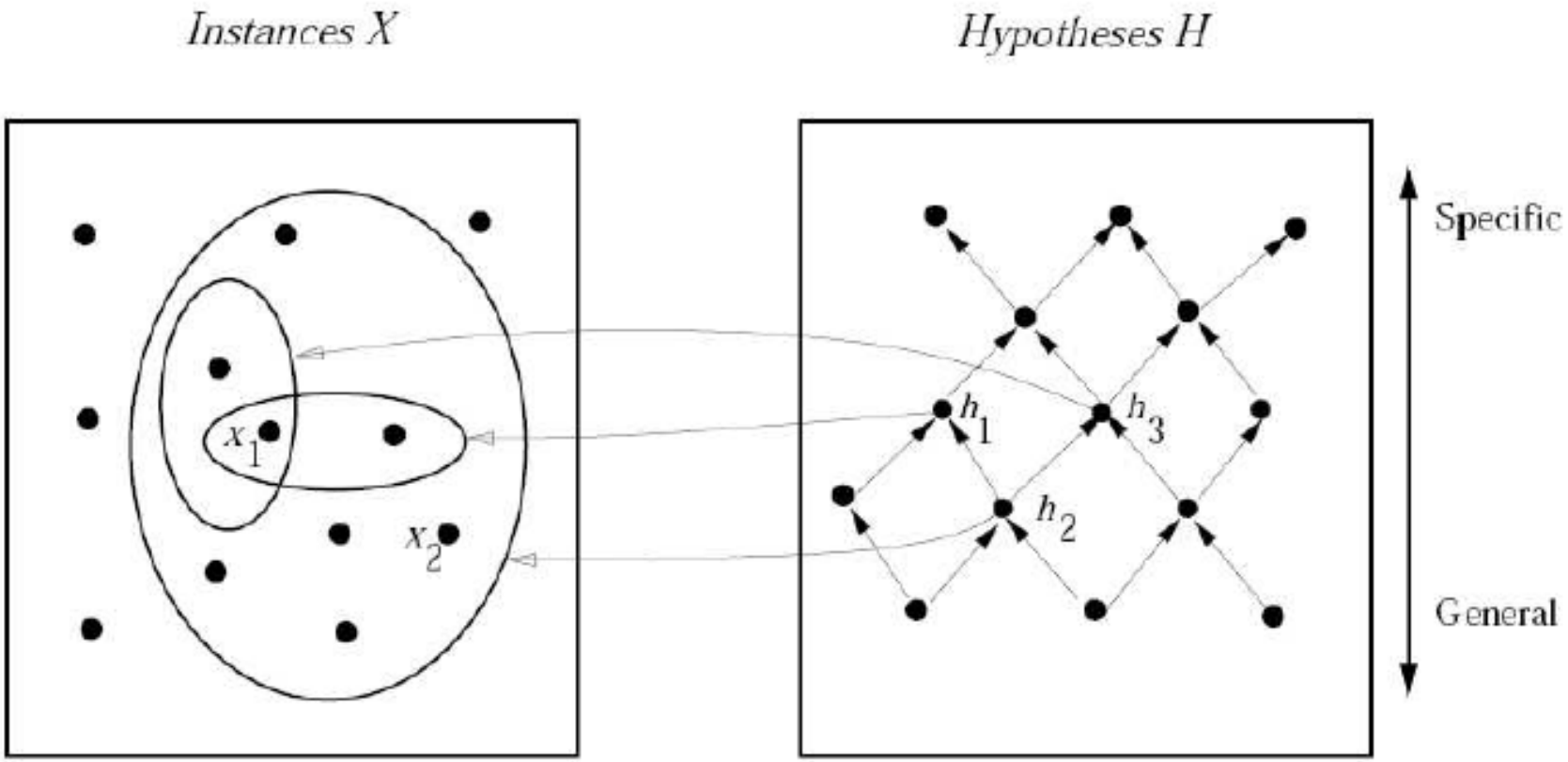
- **Hipótesis**: representación de un concepto
- **Espacio de hipótesis**: el conjunto de todos los conceptos que pueden ser descritos con la representación escogida
- **Espacio de versiones**: Conjunto de hipótesis coherentes con el conjunto de ejemplos positivos y negativos estudiado, es decir, que reconocen a todos los ejemplos positivos y excluyen a todos los ejemplos negativos.

Nomenclatura

- **Hipótesis**: representación de un concepto
- **Espacio de hipótesis**: el conjunto de todos los conceptos que pueden ser descritos con la representación escogida
- **Espacio de versiones**: Conjunto de hipótesis coherentes con el conjunto de ejemplos positivos y negativos estudiado, es decir, que reconocen a todos los ejemplos positivos y excluyen a todos los ejemplos negativos.

Espacio de versiones

- **Orden parcial**: h1 es más general que h2 ($h1 >_g h2$) si el conjunto de instancias cubierto por h2 es un subconjunto del conjunto de instancias cubierto por h1



Espacio de versiones

- **Representación del espacio de versiones**:
  - Enumeración de todas las hipótesis (inviable por su tamaño)
  - Conjunto más general (G) y más específico (S)

Espacio de versiones

- **Representación del espacio de versiones**:
  - Enumeración de todas las hipótesis (inviable por su tamaño)
  - Conjunto más general (G) y más específico (S)

Espacio de versiones

- **Representación del espacio de versiones**:
  - Enumeración de todas las hipótesis (inviable por su tamaño)
  - Conjunto más general (G) y más específico (S)

Ordenación de hipótesis

- $>=_g$ no depende del concepto que se va a aprender
- el operador define *un orden parcial* sobre el conjunto de las hipótesis
- está también la versión estricta: $>_g$
- .. más específica que: $<=_g$

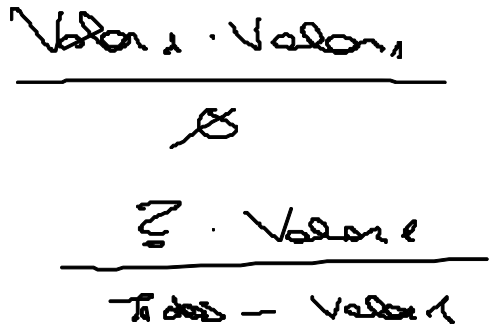
Algoritmo Find-S

- Comienza con la hipótesis más específica: ($\emptyset$,$\emptyset$,$\emptyset$,$\emptyset$,$\emptyset$)
- Generaliza si el ejemplo positivo no está cubierto por la hipótesis

Proceso de generalización

- Hipótesis Actual: (h1, h2, h3, h4, h5)
- Ejemplo nuevo: (e1, e2, e3, e4, e5)
- Hipótesis generalizada: (g1, g2, g3, g4, g5)
  - Si $h_i =?$ Entonces $g_i =?$
  - Si $h_i = e_i$ Entonces $g_i = h_i$
  - Si $h_i \neq e_i$ Entonces $g_i =?$
  - Si $h_i = \emptyset$ Entonces $g_i = e_i$

Proceso de especialización:



- Hipótesis Actual: (h1, h2, h3, h4, h5)
- Ejemplo negativo: (e1, e2, e3, e4, e5)
- Hipótesis especializadas: $[G_1, G_2, G_3, \ldots]$
  - Si $hi =?$ Entonces generar una hipótesis por cada $gi \neq ei$
  - Si $hi \neq ei$ y $hi \neq ?$ entonces la hipótesis no cubre el ejemplo negativo
  - Si $hi = ei$ Entonces $gi = \emptyset$

Algoritmo Find-S

DUAL FIND-S

Concepto de especialización generalizar.

la hipótesis cubre el ejemplo x.

Pseudocódigo: O(n·m)

```
1  Initialize the most specific hypothesis h
2
3  For each positive instance x:
4     For each attribute ai in h:
5        if x[ai] satisfy the constraint of h:   Si el atb satisface la hipótesis
6           Nothing
7        else:                                   Si no la satisface.
8           Replace ai in h with a generalization
9
10 return h
```

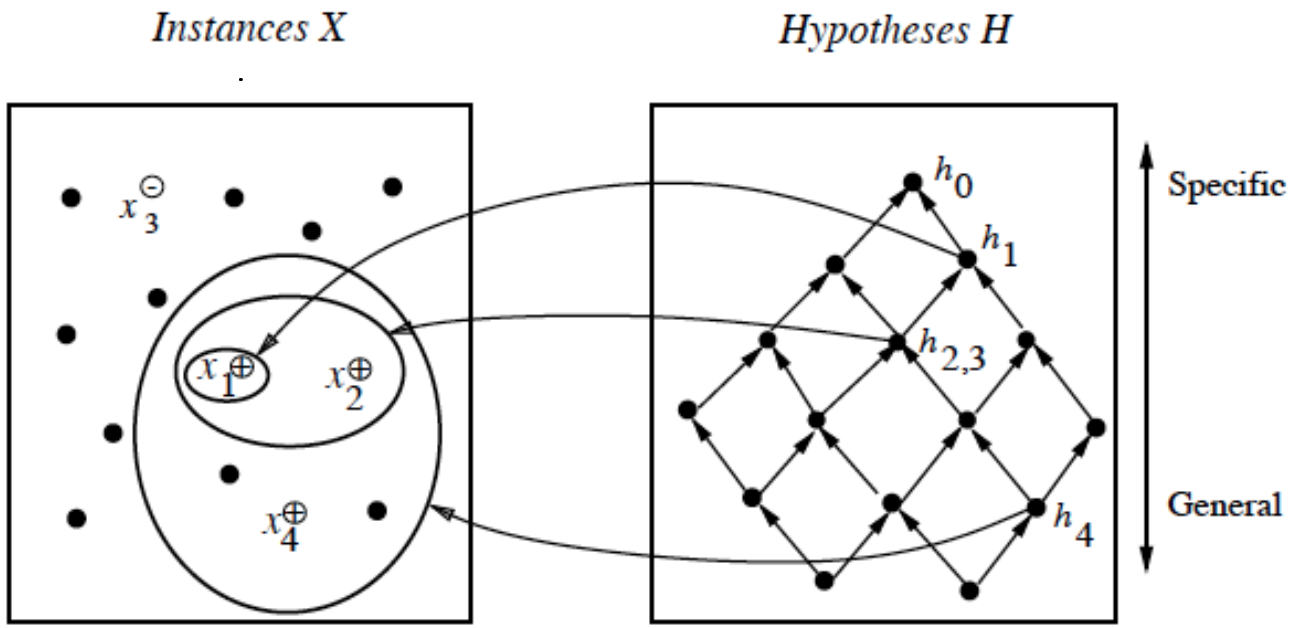
generaliza demasiado y puede cubrir algún ejemplo

Función Especialización → devuelve la lista de posibles hipótesis

Algoritmo Find-S

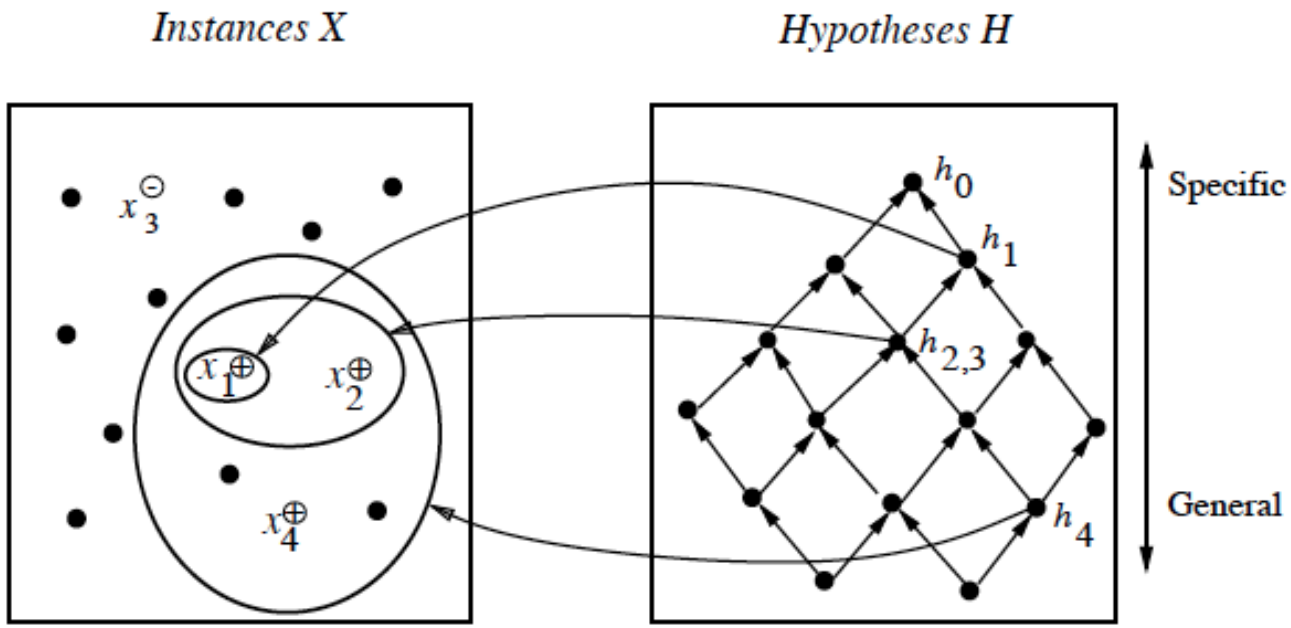
x_1 = <Sunny Warm Normal Strong Warm Same>, +
x_2 = <Sunny Warm High Strong Warm Same>, +
x_3 = <Rainy Cold High Strong Warm Change>, -
x_4 = <Sunny Warm High Strong Cool Change>, +

$h_0 = (\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$
h_1 = (Sunny,Warm,Normal,Strong,Warm,Same)
h_2 = (Sunny,Warm,?,Strong,Warm,Same)
h_3 = (Sunny,Warm,?,Strong,Warm,Same)
h_2 = (Sunny,Warm,?,Strong,?,?)



Algoritmo Find-S

	$h_0 = (\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$
x_1 = <Sunny Warm Normal Strong Warm Same>, +	h_1 = (Sunny,Warm,Normal,Strong,Warm,Same)
x_2 = <Sunny Warm High Strong Warm Same>, +	h_2 = (Sunny,Warm,?,Strong,Warm,Same)
x_3 = <Rainy Cold High Strong Warm Change>, -	h_3 = (Sunny,Warm,?,Strong,Warm,Same)
x_4 = <Sunny Warm High Strong Cool Change>, +	h_2 = (Sunny,Warm,?,Strong,?,?)

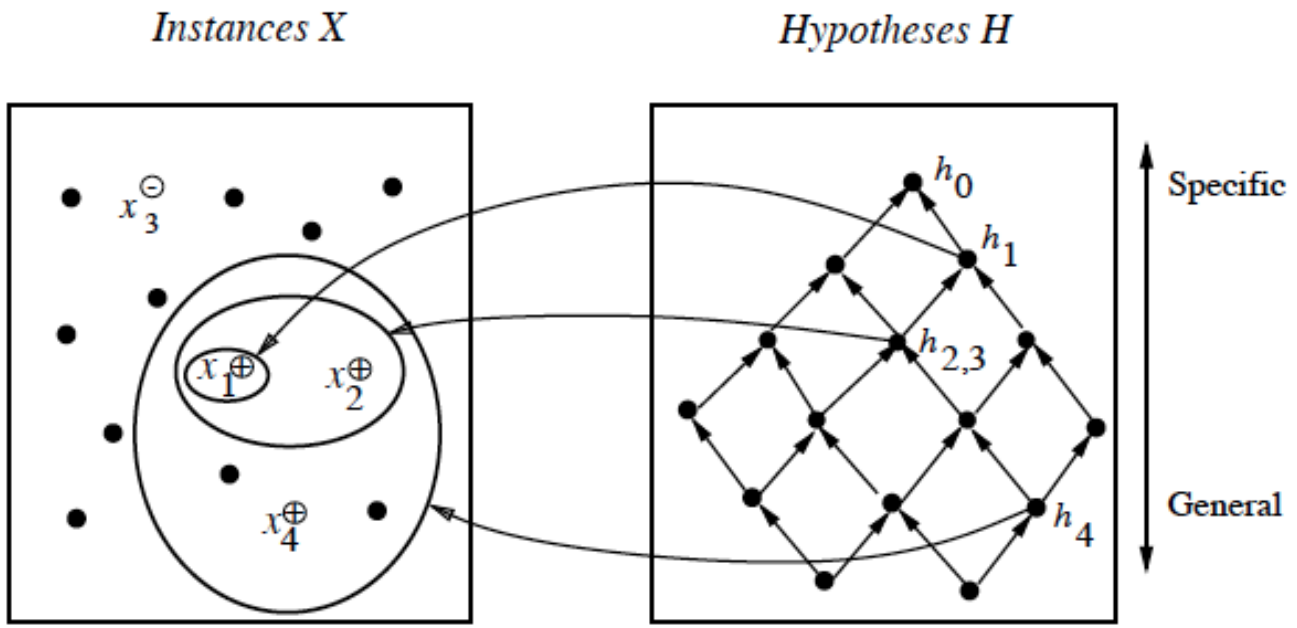


Algoritmo Find-S

x_1 = <Sunny Warm Normal Strong Warm Same>, +
x_2 = <Sunny Warm High Strong Warm Same>, +
x_3 = <Rainy Cold High Strong Warm Change>, -
x_4 = <Sunny Warm High Strong Cool Change>, +

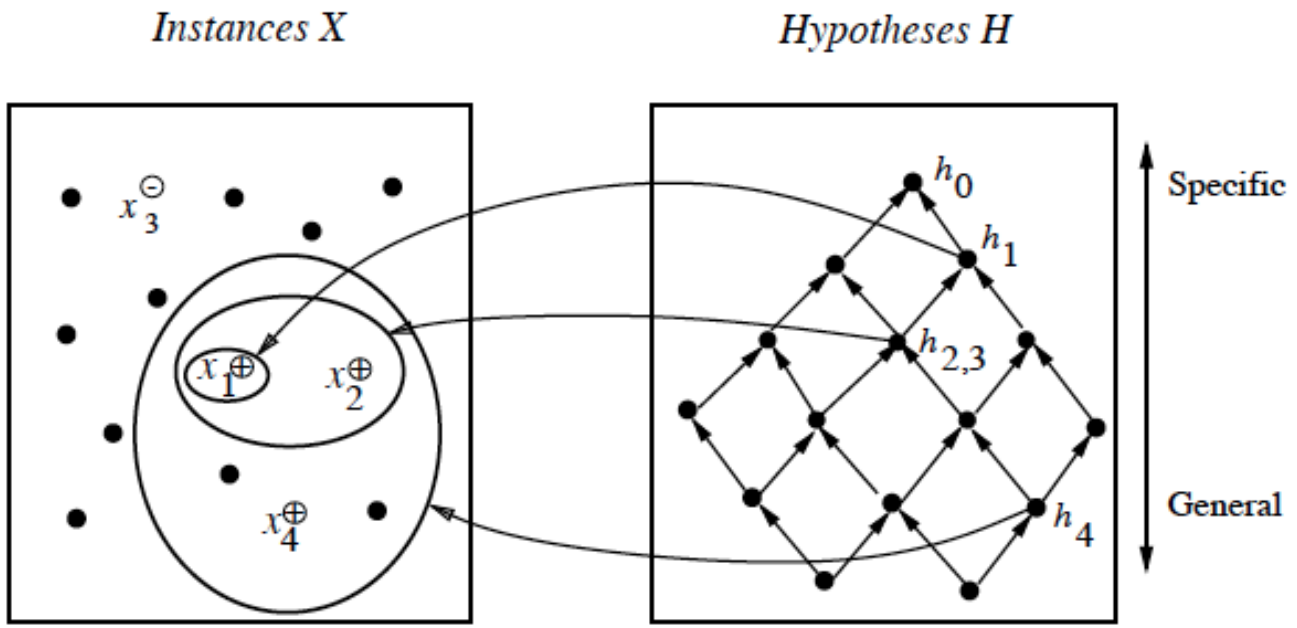
$h_0 = (\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$
h_1 = (Sunny,Warm,Normal,Strong,Warm,Same)
h_2 = (Sunny,Warm,?,Strong,Warm,Same)
h_3 = (Sunny,Warm,?,Strong,Warm,Same)
h_2 = (Sunny,Warm,?,Strong,?,?)

Normal · High
?



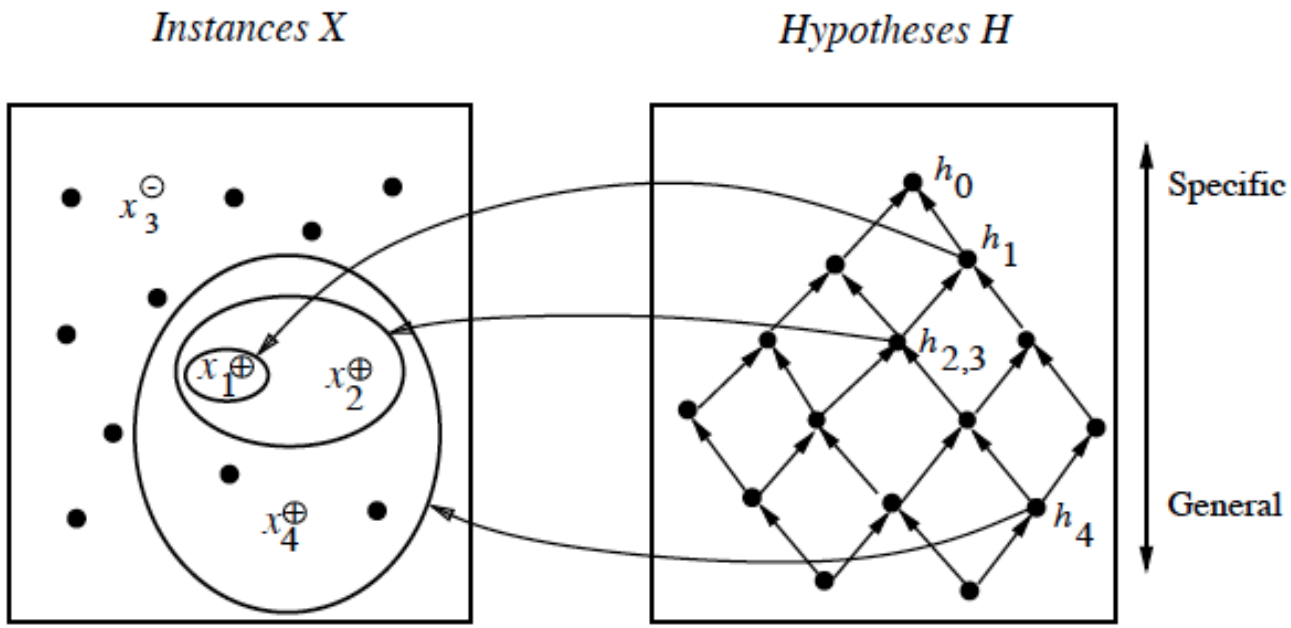
Algoritmo Find-S

x_1 = <Sunny Warm Normal Strong Warm Same>, +	$h_0 = (\emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$
x_2 = <Sunny Warm High Strong Warm Same>, +	h_1 = (Sunny,Warm,Normal,Strong,Warm,Same)
x_3 = <Rainy Cold High Strong Warm Change>, -	h_2 = (Sunny,Warm,?,Strong,Warm,Same)
x_4 = <Sunny Warm High Strong Cool Change>, +	h_3 = (Sunny,Warm,?,Strong,Warm,Same)
	h_2 = (Sunny,Warm,?,Strong,?,?)



Algoritmo Find-S

	$h_0 = (\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$
x_1 = <Sunny Warm Normal Strong Warm Same>, +	h_1 = (Sunny,Warm,Normal,Strong,Warm,Same)
x_2 = <Sunny Warm High Strong Warm Same>, +	h_2 = (Sunny,Warm,?,Strong,Warm,Same)
x_3 = <Rainy Cold High Strong Warm Change>, -	h_3 = (Sunny,Warm,?,Strong,Warm,Same)
x_4 = <Sunny Warm High Strong Cool Change>, +	h_2 = (Sunny,Warm,?,Strong,?,?)



Aspectos negativos

- Siempre se va a generar una hipótesis consistente con los ejemplos (se ignoran los negativos)
- No puede asegurar que se haya aprendido el concepto correcto, porque coge una de las hipótesis posibles.
- No soporta ruido en los ejemplos positivos

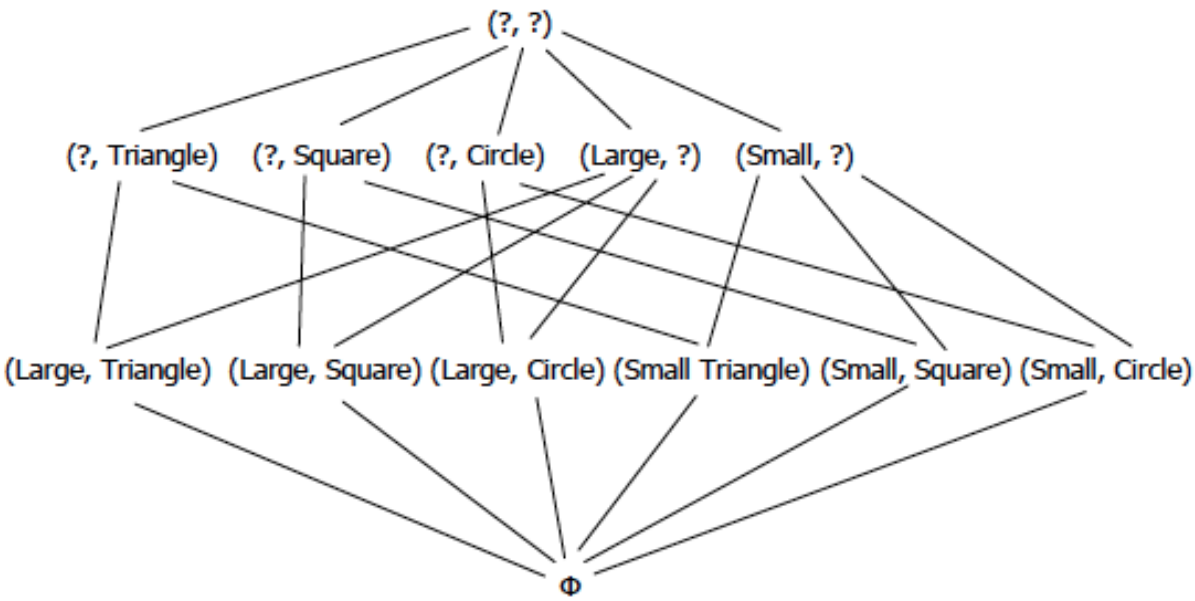
Todas las hipótesis

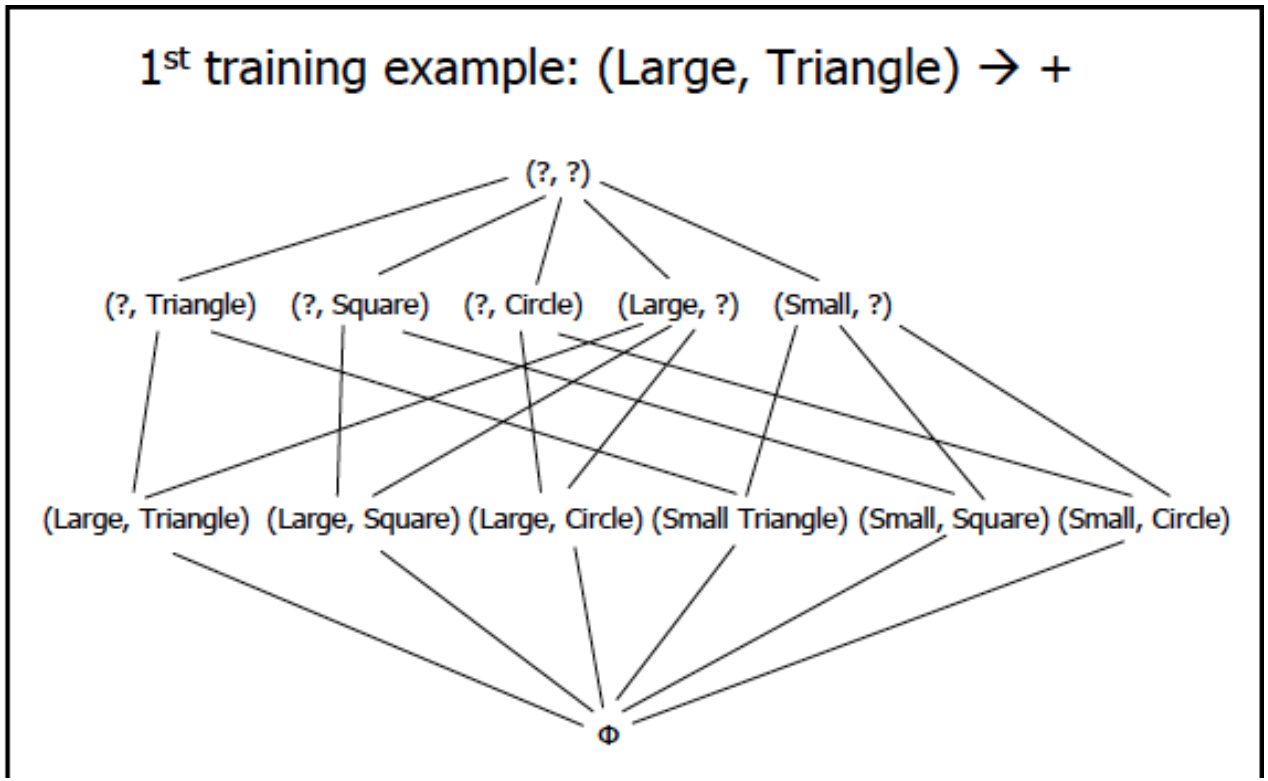
- Construcción del espacio de hipótesis completo.
- Dado un ejemplo positivo, se excluyen las hipótesis que no lo cubren
- Dado un ejemplo negativo, se excluyen todas las hipótesis que si lo cubren

Por ejemplo:

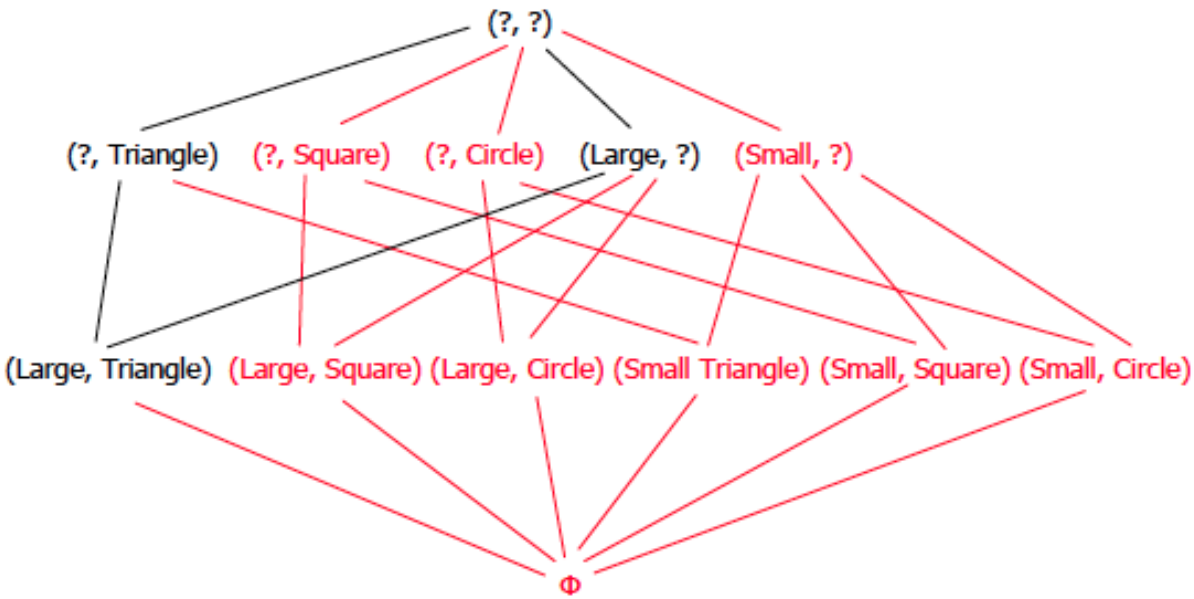
Atributo	Valor
Size	Large, Small
Shape	Triangule , Square, Circle

Example 1


(?, ?)
(?, Triangle)
(?, Square)
(?, Circle)
(Large, ?)
(Small, ?)
(Large, Triangle)
(Large, Square)
(Large, Circle)
(Small Triangle)
(Small, Square)
(Small, Circle)
Φ

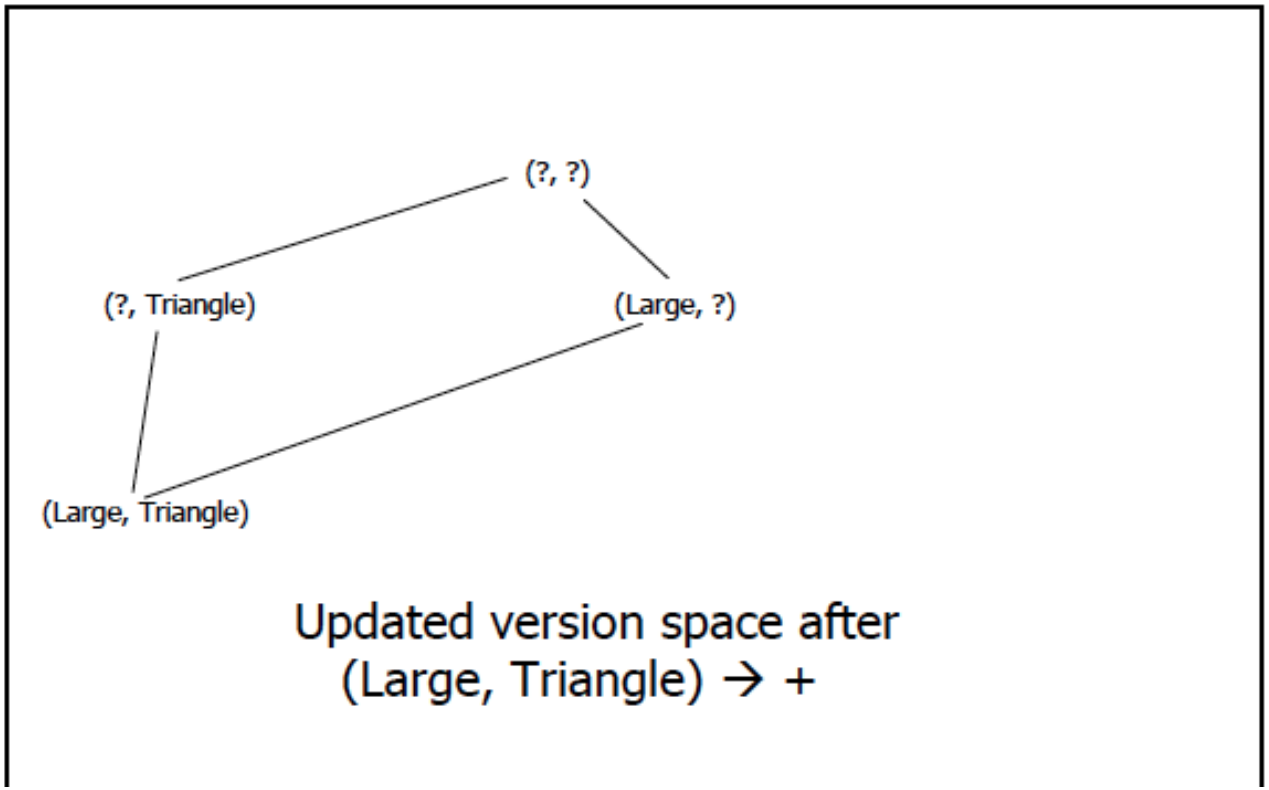

1st training example: (Large, Triangle) → +
(?, ?)
(?, Triangle)
(?, Square)
(?, Circle)
(Large, ?)
(Small, ?)
(Large, Triangle)
(Large, Square)
(Large, Circle)
(Small Triangle)
(Small, Square)
(Small, Circle)
Φ

1st training example: (Large, Triangle) → +

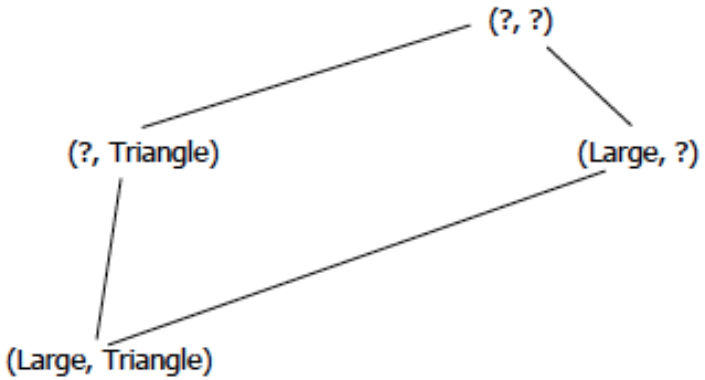


Remove all concepts that do not include (Large, Triangle)

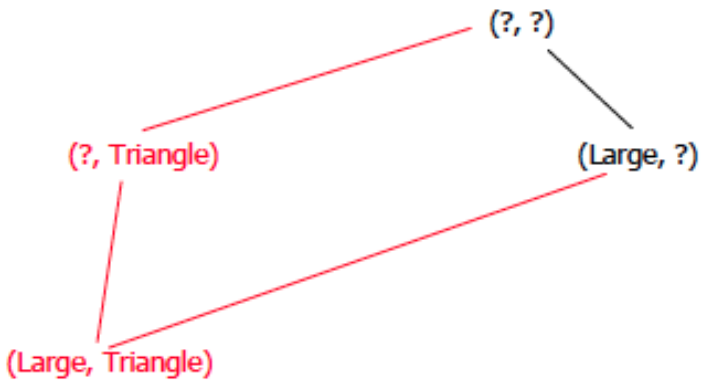
I.e., remove all concept descriptions that (Large, Triangle) does not match


(?, ?)
(?, Triangle)
(Large, ?)
(Large, Triangle)
Updated version space after
(Large, Triangle) → +

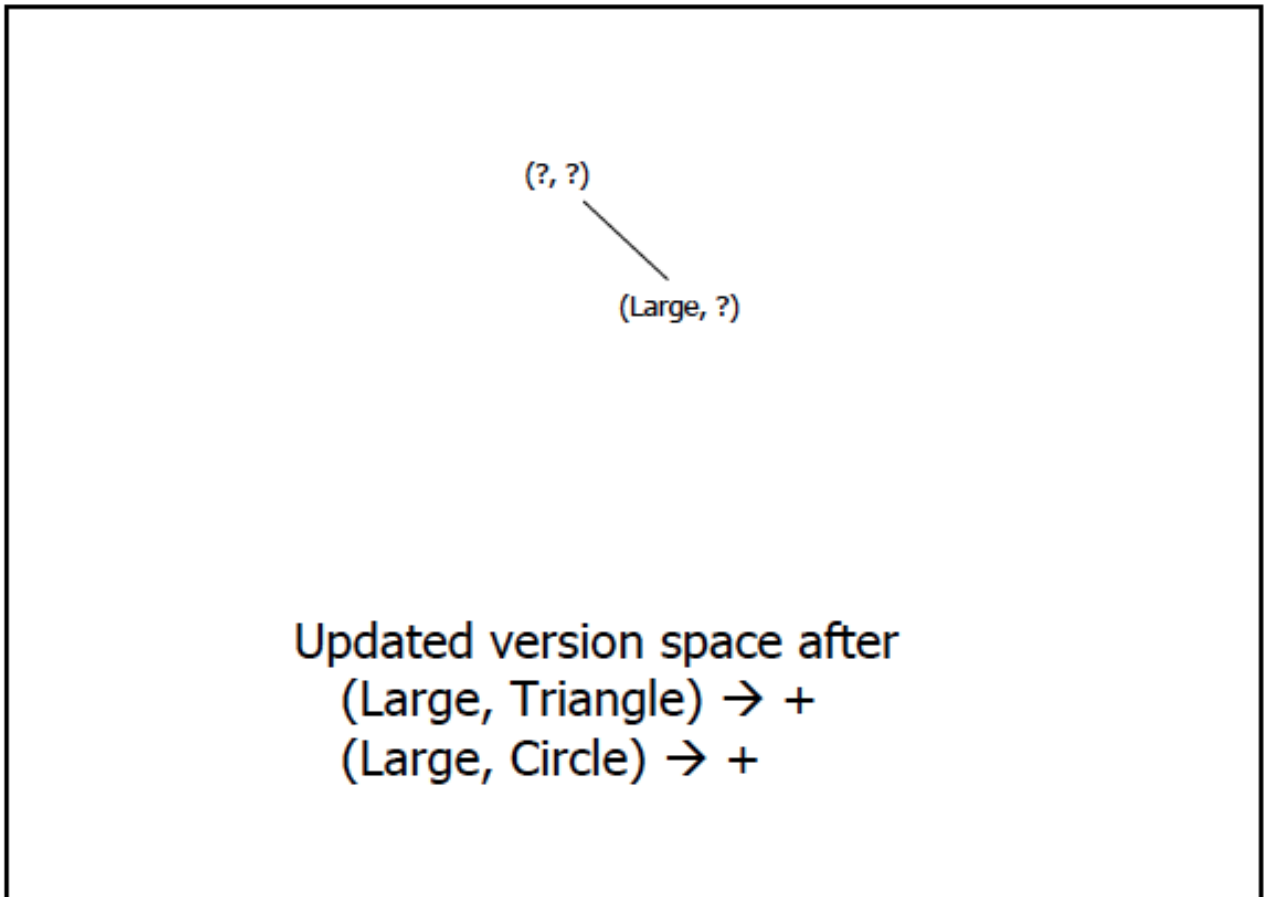
2nd training example: (Large, Circle) → +



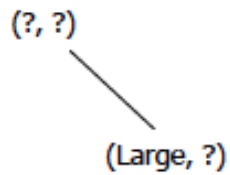
2nd training example: (Large, Circle) → +



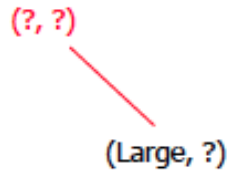
Remove all concept descriptions that (Large, Circle) does not match


(?, ?)
(Large, ?)
Updated version space after
(Large, Triangle) → +
(Large, Circle) → +

3rd training example: (Small, Circle) → -



3rd training example: (Small, Circle) → -



Remove all concept descriptions that (Small, Circle) does match

(Large, ?)

Updated version space after
(Large, Triangle) → +
(Large, Circle) → +
(Small, Circle) → -

Algoritmo

Pseudocódigo

```
for each example x in dataset:
    for each hipotesis h in VS:
        if x not satisfy h:
            remove h

Return remaining VS
```

Eliminación de candidatos

- Las opciones anteriores son demasiado costosas.
- Debemos de tener una opción que sea más tratable.

- El algoritmo de Eliminación de Candidatos mantiene “las cotas” de hipótesis superior e inferior de todas las hipótesis consistentes con los ejemplos.
- Cada tratamiento de ejemplos generaliza y especializa el grafo de las hipótesis, para obtener un espacio final compatible.

Eliminación de candidatos

Algoritmo:

- Entrada: Conjunto de datos.
- Salida:
  - G = Hipótesis genéricas maximales
  - S = Hipótesis específicas maximales
- Las hipótesis se representan en un retículo con orden parcial

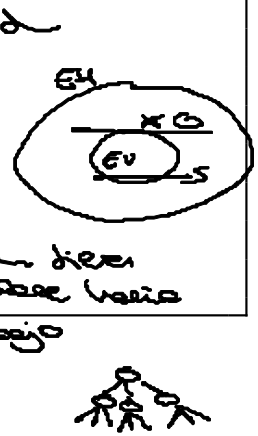
Eliminación de Candidatos

Pseudocódigo:

```
1. Sea G el conjunto de elementos de maxima generalidad de H.
2. Sea S el conjunto de elementos de maxima especificidad de H.

3. Para cada ejemplo d del conjunto de entrenamiento D:
  3.1 Si d es un ejemplo positivo, entonces:
    3.1.1 Eliminar de G cualquier hipotesis inconsistente con d.
    3.1.2 Para cada hipotesis s de S inconsistente con d:
      * Eliminar s de S.
      * Incluir en S todas las generalizaciones minimales h de s, tales que h
        es consistente con d y existe una hipotesis en G mas general que h.
      * Eliminar de S aquellas hipotesis tales que exista en S otra hipotesis
        mas general.

  3.2 Si d es un ejemplo negativo, entonces:
    3.2.1 Eliminar de S cualquier hipotesis inconsistente con d.
    3.2.2 Para cada hipotesis g de G inconsistente con d:
      * Eliminar g de G.
      * Incluir en G todas las especializaciones minimales h de g, tales que h
        es consistente con d y existe una hipotesis en S mas especifica que h.
      * Eliminar de G aquellas hipotesis tales que exista en G otra hipotesis
        mas especifica
```



```
1. Let G be the set of elements of maximum generality of H.
2. Let S be the set of elements of maximum specificity of H.

3. For each example d of training set D:
    3.1 If d is a positive example, then:
        3.1.1 Remove from G any hypothesis inconsistent with d.
        3.1.2 For each hypothesis s of S inconsistent with d:
            * Delete s from S.
            * Include in S all minimal generalizations h of s, such that h
              is consistent with d and there is a hypothesis in G more general than h.
            * Eliminate from S those hypotheses such that another hypothesis exists in S
              more general.

    3.2 If d is a negative example, then:
        3.2.1 Eliminate from S any hypotheses inconsistent with d.
        3.2.2 For each hypothesis g of G inconsistent with d:
            * Remove g from G.
            * Include in G all minimal specializations h of g, such that h
              is consistent with d and there is a hypothesis in S more specific than h.
            * Eliminate from G those hypotheses such that there is another hypothesis
              in G more specific.
```

Ejemplo

Paso 0: $S_0 = \{< \emptyset, \emptyset, \emptyset, \emptyset, \emptyset >\}$, $G_0 = \{<?, ?, ?, ?, ? >\}$

Ejemplo

Paso 1:

- Ejemplo **positivo**: $< Sol; Templ; Normal; Fuerte; Templ; Igual >$
- Nada que eliminar de G_0
- Generalización minimal de S_0:
  $< Sol; Templ; Normal; Fuerte; Templ; Igual >$
- Esta generalización es más específica que la hipótesis de G0
- Luego:
  - $G1 = \{<?; ?; ?; ?; ?; ? >\}$
  - $S1 = \{< Sol; Templ; Normal; Fuerte; Templ; Igual >\}$

Ejemplo

Paso 2:

- Ejemplo **positivo**: $< Sol; Templ; Alta; Fuerte; Templ; Igual >$
- Nada que eliminar de G1
- Generalización minimal de
  $S1 :< Sol; Templ; ?; Fuerte; Templ; Igual >$
- Esta generalización es más específica que la hipótesis de G1
- Luego:
  - $G2 = \{<?; ?; ?; ?; ?; ? >\}$
  - $S2 = \{< Sol; Templ; ?; Fuerte; Templ; Igual >\}$

Ejemplo

Paso 3:

- Ejemplo **negativo**:
  $< Lluvia; Fria; Alta; Fuerte; Templada; Cambio >$
- Nada que eliminar de S2.
- Especializaciones minimales de G2 que son mas generales que la hipotesis de S2:
  $< Sol; ?; ?; ?; ?; ? >$, $<?; Templ; ?; ?; ?; ? >$ y $<?; ?; ?; ?; ?; Igual >$.
- Luego:
  - $S3 = \{< Sol; Templ; ?; Fuerte; Templ; Igual >\}$
  - $G3 = \{< Sol; ?; ?; ?; ?; ? >, <?; Templ; ?; ?; ?; ? >$ y $<?; ?; ?; ?; ?; Igual >\}$

Ejemplo

Paso 4:

- Ejemplo **positivo**: $< \mathit{Sol}; \mathit{Templ}; \mathit{Alta}; \mathit{Fuerte}; \mathit{Fria}; \mathit{Cambio} >$
- Eliminamos de G3 la hipótesis: $<?; ?; ?; ?; ?; \mathit{Igual} >$
- Generalización minimal de S3: $< \mathit{Sol}; \mathit{Templ}; ?; \mathit{Fuerte}; ?; ? >$.
- Luego:
  - $S4 = \{< \mathit{Sol}; \mathit{Templ}; ?; \mathit{Fuerte}; ?; ? >\}$
  - $G4 = \{< \mathit{Sol}; ?; ?; ?; ?; ? >, <?; \mathit{Templ}; ?; ?; ?; ? >\}$

Propiedades

Sean S y G obtenidos por eliminación de candidatos

- Si S y G son no vacíos, resultan ser respectivamente la cota específica y cota general del espacio de versiones (respecto del conjunto de entrenamiento)
- Si S = G = $\{h\}$, entonces h es la única hipótesis de H consistente con todos los ejemplos
- Si S = G = $\emptyset$;, no existe $h \in H$ consistente con los ejemplos

Propiedades

Convergencia hacia el concepto objetivo, siempre que:

- Conjunto de entrenamiento suficientemente grande
- Ejemplos sin errores (ausencia de ruido)
- El concepto objetivo esta en H

Índice

1. Introducción
2. Análisis de diferencias
   - Winston
     - Redes semánticas
     - Método de cotejamiento
     - Generalización
     - Especialización
3. Espacio de versiones
   - Find-S
   - List-Then-Eliminate
   - Eliminación de candidatos
4. Bibliografía

Bibliografía

- Mitchell (1997): “Machine Learning”. McGraw-Hill.
- A. Moreno Ribas y otros (1994). “Aprendizaje Automático.” Ediciones UPC (Universidad Politécnica de Cataluña)
- Método Winston