Utilisation des connaissances pour l'apprentissage

Plan

- Formulation logique du problème d'apprentissage.
- Apprentissage inductif en logique.
  - Recherche de la meilleure hypothèse courante.
  - Algorithme d'élimination de candidats.
- Utilisation des connaissances pour l'apprentissage
  - Apprentissage à base d'explications
  - Programmation logique inductive

Connaissances et apprentissage

- Comment est-ce que l'agent peut utiliser les connaissances qu'il a sur son environnement pour aider l'apprentissage ?
- Les exemples, les classifications et les hypothèses seront représentés en logique du premier ordre.
- Les connaissances déjà connues vont pouvoir influencer la classification de nouveaux exemples.

Formulation logique

- Reprenons l'exemple du restaurant pour illustrer la nouvelle représentation logique.

Example	Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	*Yes*
X_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	*No*

- L'exemple X1 se représente par l'énoncé logique suivant:

$$Alt(X_1) \wedge \neg Bar(X_1) \wedge \neg Fri(X_1) \wedge Hun(X_1) \wedge Pat(X_1, Some) \wedge Price(X_1, \$\$\$) \wedge \neg Rain(X_1) \wedge Res(X_1) \wedge Type(X_1, French) \wedge Est(X_1, 0-10)$$

- La classification se représente par le prédicat:

$$WillWait(X_1)$$

4

Formulation logique

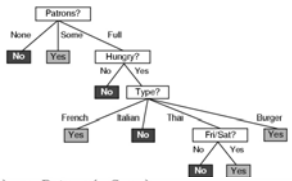
- Chaque hypothèse propose un équivalent pour le prédicat but qui est appelé une définition candidate.
- Si C_i est une définition candidate et Q le prédicat but, alors les hypothèses H_i auront la forme:

$$\forall x \quad Q(x) \quad \Leftrightarrow \quad C_i(x)$$

5

Formulation logique

- Par exemple, l'arbre de décision suivant peut être représenté par l'équivalence suivante:



$$\forall r \; WillWait(r) \Leftrightarrow Patrons(r, Some) \vee Patrons(r, Full) \wedge Hungry(r) \wedge Type(r, French) \vee Patrons(r, Full) \wedge Hungry(r) \wedge Type(r, Thai) \wedge Fri/Sat(r) \vee Patrons(r, Full) \wedge Hungry(r) \wedge Type(r, Burger)\,.$$

6

Espace d'hypothèses

- L'espace d'hypothèse H est l'ensemble de toutes les hypothèses $\{H_1, \ldots, H_n\}$.
- Une hypothèse est incompatible (inconsistent) avec une autre si elles ne s'entendent pas sur la prédiction d'au moins un exemple.
- Une hypothèse peut être incompatible avec un exemple dans deux cas:
  - Exemple faux négatif pour une hypothèse:
    - L'hypothèse dit que l'exemple devrait être négatif, mais il est positif.
  - Exemple faux positif pour une hypothèse:
    - L'hypothèse dit que l'exemple devrait être positif, mais il est négatif.

7

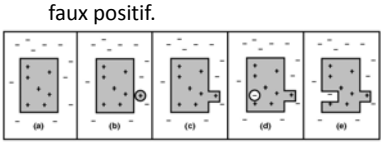
Apprentissage inductif en logique

- On part avec l'espace d'hypothèses.
- Pour chaque exemple, on retire de l'ensemble toutes les hypothèses qui sont incompatibles avec l'exemple.
- Comme l'espace d'hypothèses peut être très grand, il est souvent impossible d'énumérer toutes les hypothèses.
- On va voir deux algorithmes permettant de trouver des hypothèses compatibles sans avoir à toutes les énumérer.

8

Recherche de la meilleure hypothèse courante

- L'idée est de conserver une seule hypothèse et de la modifier pour conserver la compatibilité avec les exemples.
- L'algorithme généralise l'hypothèse s'il rencontre un exemple faux négatif.
- Il spécialise l'hypothèse s'il rencontre un exemple faux positif.



a) Hypothèse compatible
b) Faux négatif
c) Généralisation
d) Faux positif
e) Spécialisation

9

Algorithme

function CURRENT-BEST-LEARNING(*examples*) **returns** a hypothesis

$H \leftarrow$ any hypothesis consistent with the first example in *examples*
for each remaining example in *examples* **do**
  if e is false positive for H **then**
    $H \leftarrow$ **choose** a specialization of H consistent with *examples*
  else if e is false negative for H **then**
    $H \leftarrow$ **choose** a generalization of H consistent with *examples*
  if no consistent specialization/generalization can be found **then fail**
end
return H

10

Exemple

Example	Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	*Yes*
X_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	*No*
X_3	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	*Yes*
X_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	*Yes*

- Le premier exemple est positif. $Alt(X_1)$ est vrai donc on peut commencer avec l'hypothèse:

$$H_1 : \forall x \ WillWait(x) \Leftrightarrow Alternate(x)$$

- Le deuxième exemple est négatif. H_1 prédit positif donc c'est un faux positif. On spécialise en ajoutant une condition.

$$H_2 : \forall x \ WillWait(x) \Leftrightarrow Alternate(x) \wedge Patrons(x, Some)$$

11

Exemple

Example	Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	*Yes*
X_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	*No*
X_3	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	*Yes*
X_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	*Yes*

- Le troisième exemple est positif. H_2 prédit négatif, donc c'est un faux négatif. On généralise en enlevant une condition:

$$H_3 : \forall x \ WillWait(x) \Leftrightarrow Patrons(x, Some)$$

- Le quatrième exemple est positif. H_3 prédit négatif donc c'est un faux négatif. On généralise en ajoutant une disjonction.

$$H_4 : \forall x \ WillWait(x) \Leftrightarrow Patrons(x, Some) \vee (Patrons(x, Full) \wedge Fri/Sat(x))$$

12

Difficultés de l'algorithme

- À chaque étape, il y a plusieurs spécialisation ou généralisation possibles.
- L'algorithme se doit de faire des retours arrières s'il a fait un mauvais choix.
- C'est très coûteux de vérifier la compatibilité de l'hypothèse à chaque tour avec tous les exemples précédents.

13

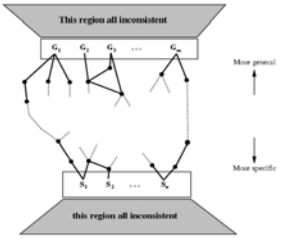
Algorithme d'élimination de candidats

- Cet algorithme, aussi appelé « version space learning », conserve toutes les hypothèses compatibles avec les exemples.
- Cet ensemble d'hypothèses compatibles avec les exemples d'entraînement est dénommé l'espace des versions.
- Cet ensemble est représenté à l'aide de deux ensembles frontières:
  - Ensemble *S*: contient les hypothèses les plus spécifiques.
  - Ensemble *G*: contient les hypothèses les plus générales.

14

Espace des versions

- Toutes les hypothèses compatibles sont plus spécifiques que G et plus générales que *S*.



15

Exemple

- On commence avec

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

- Ex 1: $<Soleil, Chaud, Normal, Fort, Chaud, Stable>$ Cet exemple est positif, donc on doit généraliser S

$$S_1 : \{< Soleil, Chaud, Normal, Fort, Chaud, Stable >\}$$
$$G_1 : \{<?, ?, ?, ?, ?, ? >\}$$

16

Exemple

- Ex 2: $<Soleil, Chaud, Élevé, Fort, Chaud, Stable>$ Cet exemple est positif, donc on doit généraliser S

$$S_2 : \{< Soleil, Chaud, ?, Fort, Chaud, Stable >\}$$
$$G_2 : \{<?, ?, ?, ?, ?, ? >\}$$

- Ex 3: $<Pluie, Froid, Élevé, Fort, Chaud, Change>$ Cet exemple est négatif, donc on doit spécialiser G

$$S_3 : \{< Soleil, Chaud, ?, Fort, Chaud, Stable >\}$$
$$G_3 : \{< Soleil, ?, ?, ?, ?, ? >, <?, Chaud, ?, ?, ?, ? >, <?, ?, ?, ?, ?, Stable >\}$$

17

Exemple

- Ex 4: $<Soleil, Chaud, Élevé, Fort, Froid, Change>$ Cet exemple est positif, donc on doit généraliser S. Dans G, il faut enlever une hypothèse, car elle vient en contradiction avec l'exemple.

$$S_4 : \{< Soleil, Chaud, ?, Fort, ?, ? >\}$$
$$G_4 : \{< Soleil, ?, ?, ?, ?, ? >, <?, Chaud, ?, ?, ?, ? >\}$$

18

Remarques

- L'algorithme d'élimination de candidats est sensible aux erreurs dans les exemples d'entraînement.
- S'il n'y a pas suffisamment d'exemples pour en arriver à générer une seule hypothèse on peut utiliser un concept partiellement appris.
  - Fonction de majorité parmi les hypothèses
  - On peut donner une probabilité, selon les hypothèses que nous avons, qu'une instance donnée soit vraie.
    - Par exemple, si pour une instance donnée, 4 hypothèses disent qu'elle est vraie et 2 hypothèses disent qu'elle est fausse, alors on peut classer l'instance comme étant vraie avec une certitude de 66%.

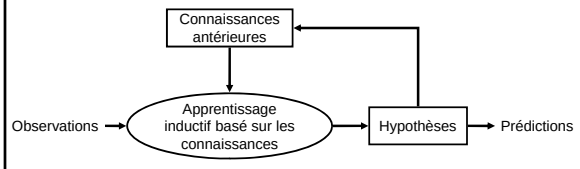
19

Utilisation des connaissances pour l'apprentissage

- Méthodes d'apprentissage inductives
  - Ex: réseaux de neurones ou arbres de décision
  - Elles ont besoins de beaucoup d'exemples d'entraînement pour être performantes.
- Méthodes d'apprentissage analytiques
  - Elles utilisent les connaissances antérieures de l'agent et des techniques de raisonnement déductif pour augmenter l'information que les exemples d'entraînement procurent.
  - Ce qui réduit le nombre d'exemples nécessaires.

20

Processus d'apprentissage



21

Apprentissage analytique

- L'hypothèse retournée doit être
  - consistante avec les exemples d'entraînement,
  - consistante avec les connaissances antérieures.
- Exemples de méthodes
  - Apprentissage à base d'explications.
  - Apprentissage basé sur la pertinence.
  - Programmation logique inductive.

22

Apprentissage à base d'explications

- Le domaine théorique (sous forme de règles) représente ce que l'on connaît déjà du monde.
- Ce domaine est utilisé pour construire une explication de l'exemple d'entraînement.
- L'explication est utilisée pour construire une règle qui est ajoutée à l'ensemble de règles.

23

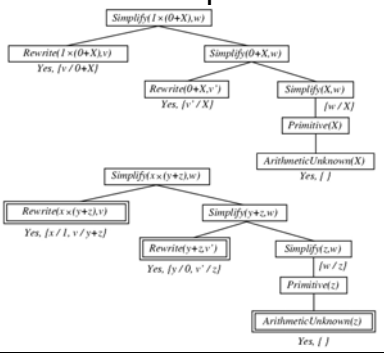
Exemple

- Le problème est de simplifier 1 x (0 + *X*).
- Dans la base de connaissances de l'agent, on a plusieurs règles d'arithmétiques:

$$
\begin{aligned}
&Rewrite(u,v) \wedge Simplify(v,w) \Rightarrow Simplify(u,w)\,.\\
&Primitive(u) \Rightarrow Simplify(u,u)\,.\\
&ArithmeticUnknown(u) \Rightarrow Primitive(u)\,.\\
&Number(u) \Rightarrow Primitive(u)\,.\\
&Rewrite(1 \times u, u)\,.\\
&Rewrite(0 + u, u)\,.\\
&\quad\vdots
\end{aligned}
$$

24

Exemple



25

Exemple

- La règle apprise:

$$Rewrite(1 \times (0+z), 0+z) \wedge Rewrite(0+z, z) \wedge ArithmeticUnknown(z) \Rightarrow Simplify(1 \times (0+z), z) .$$

- Les deux premières préconditions sont vraies peu importe la valeur de *z*, donc on peut les enlever et obtenir la règle plus générale:

$$ArithmeticUnknown(z) \Rightarrow Simplify(1 \times (0+z), z) .$$

26

Résumé du processus

- Étapes de l'apprentissage à base d'explications:
  1. Étant donné un exemple, prouver que le but peut être déduit à partir de l'exemple et des connaissances antérieures.
  2. En parallèle, construire une preuve plus générale avec des variables et en appliquant exactement les mêmes étapes que pour la preuve originale.
  3. Construire une nouvelle règle dont la partie gauche contient les feuilles de l'arbre de preuve et la partie droite contient le but écrit avec des variables.
  4. Enlever toutes les conditions qui sont vraies peut importe les variables dans le but.

27

Programmation logique inductive

- Algorithme de FOIL
  - Commence avec une règle très générale
  - Spécialise la règle en ajoutant des conditions jusqu'à ce que la règle ne couvre plus aucun exemple négatif.
  - Enlève tous les exemples positifs couverts par la règle.
  - Continue à apprendre des règles jusqu'à ce que tous les exemples positifs soient couverts.

28

FOIL

- L'algorithme génère de nouveaux prédicats utilisés pour spécialiser la règle:
  - $Q(v_1,...,v_r)$, où Q est le nom d'un prédicats apparaissant dans la liste des prédicats et où les v_i sont des nouvelles variables ou des variables déjà présentent dans la règle.
    - Au moins une des variables v_i doit être déjà présente dans la règle.
  - $Égale(x_j,x_k)$, où x_j et x_k sont des variables déjà présentent dans la règle.
  - La négation d'une des deux formes précédentes.

29

Exemple

- Considérons l'apprentissage de la règle $PetiteFille(x,y)$.
  - La liste des prédicats contient: $Père$ et $Femme$ pour décrire les exemples.
- La recherche commence avec la règle la plus générale:

$$\Rightarrow PetiteFille(x,y)$$

- Pour spécialiser la règle, on considère les prédicats suivants:
  - $Égale(x,y)$, $Femme(x)$, $Femme(y)$, $Père(x,y)$, $Père(y,x)$, $Père(x,z)$, $Père(z,x)$, $Père(y,z)$, $Père(z,y)$, $PetiteFille(x,y)$, $PetiteFille(z,y)$, $PetiteFille(x,z)$, $PetiteFille(y,x)$, $PetiteFille(y,z)$, $PetiteFille(z,x)$
  - Et la négation de tous ces prédicats.

30

Exemple

- Supposons que c'est le prédicat *Père*(*y*,*z*) qui a été choisi comme étant le prédicat le plus prometteur. On obtient donc:

$$Père(y, z) \Rightarrow PetiteFille(x, y)$$

- La liste des prédicats pour spécialiser cette règle contient tous les prédicats de l'étape précédente, plus les prédicats suivants: $Femme(z)$, $Égale(z,x)$, $Égale(z,y)$, $Père(z,w)$, $Père(w,z)$ et leur négation.
- Ces nouveaux prédicats sont considérés à cette étape parce que la variable z a été ajoutée à l'étape précédente.

31

Exemple

- Si l'algorithme de FOIL choisit le prédicat $Père(z,x)$ à cette étape et $Femme(y)$ à l'étape suivante, alors on obtiendrait la règle suivante qui ne couvre aucun exemple négatif et qui couvre tous les exemples positifs, donc qui l'algorithme arrêterait.

$$Père(y, z) \wedge Père(z, x) \wedge Femme(y) \Rightarrow PetiteFille(x, y)$$

32

Sélection du meilleur prédicat

- Pour choisir le meilleur prédicat parmi les candidats à chaque étape, l'algorithme regarde la performance de la règle sur les exemples d'entraînement.
- Pour cela, il considère toutes les instanciations de variables possibles et se base sur le nombre d'instanciations positives et négatives avant et après l'ajout du prédicat.
- L'algorithme va choisir le prédicat ayant la plus grande valeur pour la fonction Foil_Gain (voir acétate suivant).

33

Sélection du meilleur prédicat

- Plus précisément, considérons la règle R. Soit R' la règle créée par l'ajout du prédicat *L* à la règle *R*. La valeur de $Foil\_Gain(L,R)$ pour l'ajout de L à la règle R est:

$$Foil\_Gain(L,R) \equiv t\left(\log_2 \frac{p_1}{p_1+n_1} - \log_2 \frac{p_0}{p_0+n_0}\right)$$

  - p_0 est le nombre d'instanciations positives de la règle R,
  - n_0 est le nombre d'instanciations négatives de la règle R,
  - p_1 est le nombre d'instanciations positives de la règle R',
  - n_1 est le nombre d'instanciations négatives de la règle R',
  - t est le nombre d'instanciations positives de la règle R qui sont encore couverts après l'ajout de L à R.

34