QuickFOIL: Scalable Inductive Logic Programming [R23-5]

- Q. Zeng, J.M. Patel, D. Page (U. Wisconsin-Madison)
- 帰納論理プログラミング(Inductive Logic Programming)
  再び:80年代～90年代に学習分野で盛ん
- 問題例

訓練サンプル	
U(Daniel, Jacob)	U(John, Jason)
U(Jason, Andrew)	U(Noah, John)
U(Noah, Andrew)	U(Jason, Justin)
U(Daniel, William)	U(Noah, Justin)

青は正例，赤は負例
U: Uncle, B: Brother, S: Sister

背景知識		
B(Andrew, Jacob)	P(Daniel, Andrew)	S(Daniel, June)
B(Jason, Noah)	P(Jason, Jacob)	S(Daniel, Jennifer)
B(Jacob, Andrew)	P(Noah, Jacob)	S(Daniel, Rachel)
B(Noah, Jacob)	P(Noah, Justin)	S(Daniel, Jason)
B(Owen, William)	P(Jimmy, Jason)	S(John, William)
		S(Noah, Gwen)
		S(Jason, Sara)

目的:仮説(例:Uncle(X, Y) :- Brother(Z, Y), Parent(X, Z))を導き出す
・仮説＋背景知識で正例が導出でき，負例が導出できないようなもの

アプローチ（1）

- トップダウンのアプローチ
  - FOIL（1990にQuinlanが提案）でも採用
  - 一般的な仮説からスタートし，探索しながら詳細化
    - 発見された仮説により，正例のみがカバーできるなら，そこで終了
    - そうでなければさらに仮説を拡張
  - 大規模データに向く
  - 比較：ボトムアップ法は小規模でインクリメンタルな場合に向く
- 例：U(X, Y) :– から開始
  - まず述語Bを対象に，変数すべての組合せ B(X, Y), B(Y, X), B(X, Z), B(Z, X), B(Y, Z), B(Z, Y), B(X, X), B(Y, Y) を列挙
  - それぞれについて，たとえば U(X, Y) :– B(Z, Y) の適切さ（有望であるか）を評価

アプローチ（2）

- 例（続き）：U(X, Y) :− B(Z, Y) の評価
  - データセットを元に正解・不正解を判定
  - 負例が入っているので，この仮説はさらに詳細化すべき
  - どの仮説を優先的に詳細化（展開）するか
    - FOILでは情報利得に基づくスコアリング関数を利用：直感的には精度の利得が大きく，カバー率が高い仮説を採用

X	Y	Z
Daniel	Yacob	Andrew
Jason	Andrew	Jacob
Noah	Andrew	Jacob
Daniel	William	Owen
John	Jason	Noah

- 本研究の貢献①：スコアリング関数の工夫
  - 一般に（特に大規模データでは）正例が多く偏りが生じる：偏りを考慮した指標（相関係数の一種）を導入
  - 従来の指標では多くの正例をカバーすることを重視し，正・負例の識別は重視していなかった：これに対する指標も加える

担当：石川（名大）

アプローチ（3）

- 本研究の貢献②：重複した候補の除去
  - 生成される仮説は，見た目は違っても意味が同じものがある
  - 重複の除去の問題を，論理積問合せ（conjunctive query）の等価性の問題に帰着させ，処理
- 本研究の貢献③：RDBを用いた実装
  - 仮説探索の処理を，RDBの演算子（結合，半結合（semijoin），逆結合（antijoin）を用いて表現
  - 例：U(X, Y) :- B(Z, Y) の U(X, Y) :- B(Z, Y), P(X, Z) への展開は，前者に対するタプル集合と背景知識におけるリレーションPの結合で表現
  - インメモリのRDBMS上で実装：ハッシュ結合における工夫
- 実験による評価
  - WebKBやHIVデータベースからの仮説抽出