Semantic Computation

Part I

語意計算 1

-PeterWolf

什麼是語意分析？

- 語意分析的定義
- 語意 (Semantics): 捕捉文字與結構所表達的意義
- 主要目標：
  - 將語言表達式映射到計算模型中的概念
  - 可能對應到軟體應用程式
  - 可能對應到邏輯中的良構公式 (well-formed formulas)

為什麼需要語意處理？

- 人們用各種不同表達方式描述相同情況
- 語意表示法讓我們能概括特定詞彙
- 更容易將資訊儲存到結構固定的資料庫中

實際例子

- 句子：「馬來西亞的粗棕櫚油產量估計上升了最多百分之六。」
- 這個句子告訴我們：
  - 國家有相關的製造和農業產出
  - 馬來西亞的產出之一是「粗棕櫚油」
  - 某個時間段的估計產量增加了「百分之六或更少」
- 可能需要釐清：
  - 測量類型（例如：體積或價值）
  - 時間段（例如：月、年或十年）
  - 背景知識（例如：這是年度生產量報告）

語意處理的用途

- 作為後續處理的前置步驟：

  1. 問答系統 (Question Answering)

  2. 知識獲取 (Knowledge Acquisition)

     - 將非結構化內容映射到結構化內容

- 三個主要關注議題：

  a) 需要表示什麼資訊？

  b) 目標語意表示是什麼？

  c) 使用什麼處理方法來映射？

需要表示的資訊 (1/2)

- 語意表示的四種主要資訊
- 根據目標任務決定需要表示的內容：

  1. 實體 (Entities) - 被描述的實體
  2. 事件與角色 (Events and Roles)
     - 被提及的事件類型
     - 實體相對於事件所扮演的角色

需要表示的資訊 (2/2)

3. 命題態度 (Propositional Attitude)
   - 陳述 (Statement)
   - 問句 (Question)
   - 請求 (Request)
4. 詞義 (Word Senses) - 句子中每個詞彙出現時的預期意義

這些資訊由什麼決定？

- 名詞片語、動詞片語、整體句子、一般語境

淺層 vs. 深層表示

- **表示法的深度**

- **淺層表示 (Shallow Representations):**
  - **識別主要動詞**
  - **標記對應實體的文本範圍**
  - **找出動詞語意角色的功能參數**
- **深層表示 (Deep Representations):**
  - **包含主要動詞及其語意角色**
  - **實體的深層語意**
  - **量化、類型限制、各種修飾語**
- **表示框架：**
  - **形式邏輯 (Formal Logics)**
  - **框架語言 (Frame Languages)**
  - **圖形式語言 (Graph-based Languages)**

處理方法

- **從文本到表示的映射方法**

- **規則式方法 (Rule-based Approach):**
  - **撰寫機器可讀的規則**
  - **明確指定所有預期映射**
  - **創建執行映射的演算法**
  - **現在沒人這麼做了！**

- **分類器方法 (Classifier Approach):**
  - **將規則表達為人類可讀的標註指南**
  - **使用標註工具建立標註語料庫**
  - **訓練分類器自動標註未標記資料**
  - **現在沒人想這麼做了！(交給 LLM 就好了，是嗎？)**

- **邏輯演算方法 ：**
  - **撰寫「語言」的意義演算規則**
  - **就這樣！沒其它要做的了。**
  - **從 Wolfram 到 Loki.**

  適用範圍：
  - **淺層表示或深層語意分析的子任務**

案例語法簡介

- **Case Grammar 案例語法**
- **核心概念：**
  - **描述不同動詞或動詞類型相關的不同角色**
- **例子：**
  - **行動動詞有施事者 (Agent)**
  - **及物動詞有直接受詞**
  - **角色可能是主題 (Theme) 或受事者 (Patient)**
- **與邏輯表示的差異：**
  - **邏輯：使用固定數量和順序的參數**
  - **案例語法：明確命名必要和修飾角色**

語意角色範例

- 常見的語意（主題）角色

主詞位置的角色：

角色	特性	例句
Agent(施事者)	+有意圖, +有感知	[貓]追老鼠
Cause(起因)	-有意圖, +無生命	[風]使窗戶嘎嘎作響
Experiencer(經驗者)	+有感知	[孩子們]喘了湯
Theme(主題)	-受影響	[球]滾下山

受詞位置的角色

角色	例句
Patient(受事者)	風摧毀了[沙堡]
Recipient(接受者)	老師給了書[給學生]
Instrument(工具)	廚師用[烤箱]烤蛋糕
Theme(主題)	男孩滾動[球]下山
Stimulus(刺激)	孩子們看到[雲]

地點與時間角色

- **語意角色：修飾語**
- **地點相關角色：**

  - Destination(目的地): 我帶了一本書 [到會議]

  - Location(地點): 我在 [洞穴] 尋找寶藏

  - Source(來源): 清潔劑去除 [衣服上的] 污漬
- **時間相關角色：**

  - Time(時間): 他在 [收到成績後] 很高興

  - Duration(持續時間): 課程持續了 [兩小時]

語意框架資源

- 重要的語意框架資源
- FrameNet 專案 (1997):
  - 由 Charles J. Fillmore 創立
  - 定義語意框架的階層結構
  - 包含超過 1200 個不同框架
  - 建立標註語意角色的句子語料庫
  - 然後做個沒完沒了…
- 統一動詞索引 (Unified Verb Index, UVI): - 包含 FrameNet
  - 整合 PropBank 、 OntoNotes 、 VerbNet
  - 由科羅拉多大學的 VerbNet 創建者維護

量化與範圍

- **Quantification and Scoping**
- **量化詞 (Quantifiers):**
  - **描述個體和集合的性質**
- **範圍歧義：**
- **「每隻貓都追一隻老鼠」可能表示：**
  1. **幾隻貓都追同一隻老鼠**
  2. **每隻貓都追自己最喜歡的老鼠**
- **自然語言中的多樣量化：**
  - **精確：「恰好一個」、「至少兩個」**
  - **模糊：「大多數」**
  - **類型限制：「大多數熊」、「大多數飢餓的熊」**

廣義量詞

- **Generalized Quantifiers**
- **組成部分：**

  1. 量詞名稱

  2. 一組被量化的變數

  3. 約束條件（使用這些變數的良構表達式）

- **例子：**

  「大多數小狗喜歡玩具」

  可表示為：

  [Most x, small(x) and dog(x)

  [All y toys(y): like(x,y)]]

- **定義依賴領域：**

  - 「大多數」可定義為「超過 50%」或「超過 60%」

程序語意學 (第五代程式語言： SQL)

- Procedural Semantics
- 概念：
  - 句子的意義是要執行的程序呼叫或查詢
  - 混合了語意學和語用學

- 例子：「找出所有姓 Smith 的客戶」
- SQL 版本：

```
SELECT * FROM Customers
WHERE Last_Name='Smith'
```

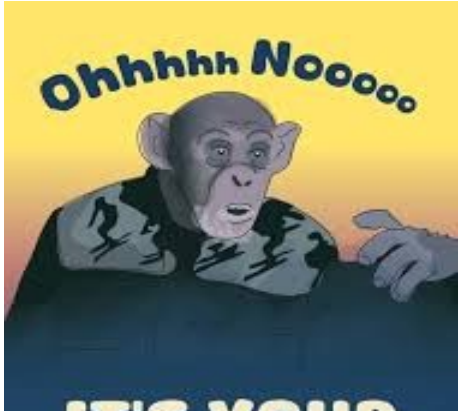
- Python 版本：

```
Customers.retrieveRows(last_name="Smith")
```

程序語意的應用

- 過去的限制：
  - 使用自然語言表達命令的困難度
  - 不值得開發成本
- 現在的機會：
  - 聊天機器人興起
  - 語音存取應用（如智慧音箱）
  - 新的應用場景

- 實作方式：
  - 規則式方法（Oh no...）
  - 標註語料庫並訓練分類器（Oh no no no...）



SQL 查詢範例

- 將自然語言映射到 SQL
- 資料庫表格：城市、國家、人口
- 查詢 1: 「哪些城市位於加拿大？」

  ```
  SELECT City FROM city_table
  WHERE Country = 'Canada'
  ```

- 查詢 2: 「休斯頓在哪個國家？」

  ```
  SELECT Country FROM city_table
  WHERE City = 'Houston'
  ```

- 方法：使用帶語意特徵的上下文無關文法規則（Context-free grammar）

圖形資料庫查詢

- **Cypher 查詢語言**
- **新興格式 : 圖形資料庫**
  - **優化關係三元組檢索**
  - **查詢語言 Cypher 更接近自然語言**
- **例子 :「 Jennifer 工作的公司名稱是什麼？」**

```
MATCH (:Person {name: 'Jennifer'})
    -[:WORKS_FOR]->(company:Company)
RETURN company.name
```

- **斷言 :「 Jennifer 和 Mark 是朋友」**

```
MATCH (jennifer:Person {name: 'Jennifer'})
MATCH (mark:Person {name: 'Mark'})
CREATE (jennifer)-[rel:IS_FRIENDS_WITH]->(mark)
```

領域獨立框架

- **Domain Independent Frameworks**
- **目標：**
  - **捕捉狀態或事件的類型及其結構**
  - **識別與事件相關的語意角色**
  - **處理量詞的解釋**
- **組合性 (Compositionality):**
  - **可以逐步建構**
  - **句法和語意成分類型之間有通用映射**
- **表示形式：**
  - **邏輯中的良構表達式（Well-formed formula, e.g., regex)**
  - **AI 框架語言（類邏輯形式）**
  - **圖形表示**

一階述詞邏輯

- **First Order Predicate Logic for NL**
- **基本符號 :**
  - **項 (Terms): 常數、變數、函數應用**
  - **原子公式 :n 元關係符號加上 n 個項**
  - **邏輯符號 :¬(非)、∨ (或)、∧ (且)**
- **量詞 :**
  - ∃ **存在量詞 (「存在」)**
  - ∀ **全稱量詞 (「對所有」)**
- **限制 :**
  - **無法表達陳述與問句的差異**
  - **需要額外的邏輯運算子 (如 ask 、 tell 、 request)**

Lambda 表達式

- **Lambda Calculus λ 演算**
- **用途：**
  - **表示量詞和帶補語的詞彙**
  - **函數抽象，可應用於參數後化簡**
- **例子：**

  λx: (λy: loves(x,y) (Milwaukee))

  = λx: loves(x, Milwaukee)
- **多變數：**
  - **巢狀：從內到外求值**
  - **單一 lambda 綁定多變數：從左到右**
- **Python 和 NLTK 都支援 lambda 表達式**

動詞與動詞片語

- **動詞的語意表示**
- **不同元數的動詞：**
  - **不及物 :λx: sleep(x)**
  - **及物 :λy: λx: eats(x,y)**
  - **雙及物 :λz: λy: λx: gives(x, y, z)**
- **動詞片語：「 chases a mouse 」 經量詞提升後：**

  λx: ∃y mouse(y) and chases(x,y)
- **完整句子：「 Susan owns a grey cat 」**

  ∃y [grey(y) and cat(y) and owns(Susan, y)]

描述邏輯

- **Description Logic (DL)**
- **目的：**
  - **克服一階邏輯的缺陷**
  - **更好地表示物件**
  - **將不同述詞作為定義的一部分相關聯**
- **三種主要類型：**
  1. **概念 (Concepts): 一元述詞（如「狗」）**
  2. **角色 (Roles): 二元述詞（如「母親」）**
  3. **常數 (Constants): 個體名稱**
- **與自然語言的對應：**
  - **普通名詞 → 概念**
  - **關係名詞 → 角色**
  - **專有名詞 → 常數**

描述邏輯語法

- **DL 的良構公式**
- **概念形成運算子 :**

  - [ALL r d]: 角色 r 的所有 d

  - [EXISTS n r]: 至少 n 個角色 r

  - [FILLS r c]: 角色 r 填入常數 c

  - [AND d1…dk]: 概念的合取
- **句子類型 :**

  - d ≡ e: d 等價於 e

  - d ⊆ e: d 被 e 包含

  - c → d: 常數 c 滿足概念 d
- **例子 :**

  cat ⊆ four_legged

  TardarSauce → cat

DL 複雜概念範例

- 描述邏輯的表達力
- 複雜定義：「有女兒的父親」(Can we say “ 沒有女兒的父親”嗎？)

```
FatherOfDaughters ≡
[AND Male
    [EXISTS 1 :Child]
    [ALL :Child Female]]
```

- 公司定義：「至少有 7 位董事，管理者都是有博士學位的女性，最低薪資 $100/ 小時」

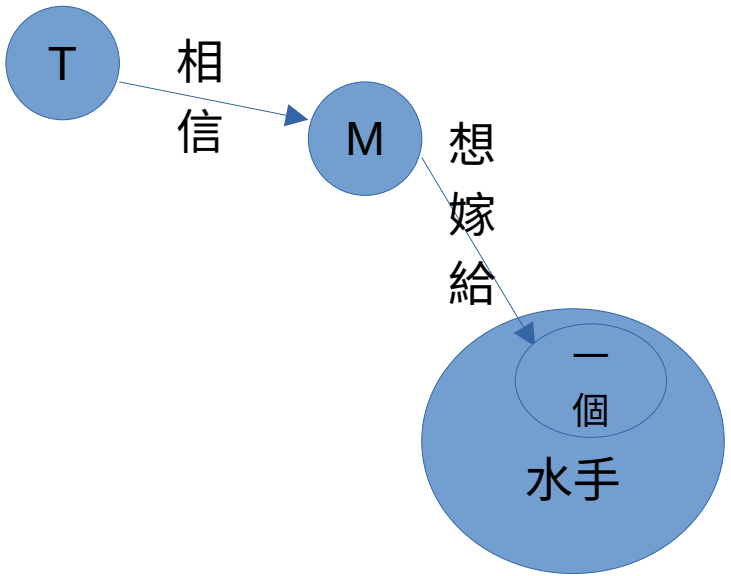
```
[AND Company
    [EXISTS 7 :Director]
    [ALL :Manager
        [AND Woman
            [FILLS :Degree PhD]
            [FILLS :MinSalary '$100/hour']]]]
```

圖形表示法

- Graph-Based Representations
- 優勢：
  - 比邏輯更直接地映射自然語言
  - 結構共享：明確相同實體
  - 量詞可跨越多個子句
  - 更具表達力，保持健全推理
- 劣勢：
  - 很…難…計…算…
- 代表性框架：
  - 概念圖 (Conceptual Graphs): 國際會議起源於 1986 年
  - SNePS: 由 Stuart Shapiro 等人開發
  - AMRL:Amazon Alexa 使用 (2018 年報導)
- 歷史淵源：
  - Charles S. Pierce 的「存在圖」(1880 年代)

圖形表示範例

- Conceptual Graph 實例
- 句子：「Tom 相信 Mary 想嫁給一個水手」
- 圖形表示的優勢：
  - 解決歧義：「一個水手」的存在性
  - 可能在真實世界
  - 可能只在某人的信念語境中
  - 明確區分不同的信念語境
  - 透過語境分割處理主觀性
- AMRL 範例：「找到鯊魚隊比賽附近的餐廳」
  - 包含複雜的空間關係



總結

- **課程重點回顧**
- **語意分析的四大要素：**

  1. 實體的描述

  2. 事件類型及實體角色

  3. 命題態度（陳述 / 問句 / 請求）

  4. 詞義消歧

- **表示法框架：**

  - 程序語意：直接映射到資料庫查詢或 API

  - 領域獨立語意：邏輯、框架語言、圖形

  - 淺層 vs. 深層：根據應用需求選擇

- **實作方法：**

  - 規則式方法

  - 分類器方法（標註語料庫 + 訓練）