Rheinisch Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik VI
Prof. Dr.-Ing. Hermann Ney

Speech Recognition and Language Processing in SS 2003

Natural Language Database Queries using Classification and Regression Trees

Samuel S. Okae

July 31, 2003

Main literature

- R. Kuhn, R. De Mori: “the application of semantic classification trees to natural language understanding," *IEEE transactions on pattern analysis and machine intelligence*, Vol. 17, No. 7, pp. 449-460, 1995.

- R.O. Duda, P.E. Hart, D.G. Stork: *pattern classification*, John Wiley and sons, New York, NY, 2001, pp. 395-411.

Overview

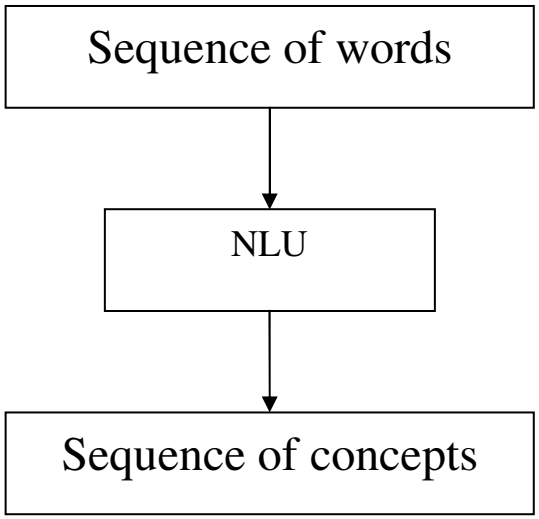
- Basics of natural language understanding (NLU)
- Basics of decision trees
- Application of decision trees to natural language processing
- Semantic classification trees
- ATIS evaluation
- Results
- Conclusion

Introduction

- objective of NLU

  to map a natural language input onto a semantic representation

e.g.



Applications of NLU

- text-based applications
  - machine translation
  - text summarization
  - information extraction

- dialogue-based applications
  - spoken dialogue systems

Spoken dialogue systems

- aim:

  to provide an interface between a user and a computer-based application

- example:

  - air travel information systems

  - telephone directory assistance

- NLU used for interpretation of inputs

Related work

- rule-based approaches (CFG, PCFG)
- Hidden Markov Models (HMM)
- machine translation approach
- decision trees

Decision trees

- definition:

  a decision tree is a classifier in the form of a tree structure where each node is either:

  leaf node - indicates the value of the target attribute (e.g. word sequence in NLP)

  decision node - specifies some test to be carried out on attribute-values

- lines or branches represent outcome of a test at decision node

Decision trees

aim:

test attributes in order to predict an output

used for :

- equipment or medical diagnosis
- risk analysis
- calendar scheduling
- used largely for prediction purposes

Building Decision Trees

1. how many splits will be used?
2. which property should be tested?
3. when should a node be declared a leaf node?
4. if a tree becomes “too large” how can it be made smaller?
5. If a leaf node is impure how should the category label be assigned?
6. how should missing data be handled?

Decision tree construction algorithm

1. *create root node*
2. *select the best attribute for the current node*
3. *generate child nodes, one for each possible value of the selected attribute*
4. *test property at the current node and create branches:*
   - *if no question declare as leaf node*
   - *false decisions on the left branch*
   - *true decisions on the right branch*
5. *if each child node is a leaf node, stop*
6. *else set child to current node and go to 2*

Applications of DTs to NLP

- probabilistic language modeling
- trained to guess the past participle of verbs
- applied to the task of partitioning newspaper articles into classes

Semantic Classification Trees

- specialized decision trees originally developed for semantic classification of word sequences
- learn semantic rules from training data to translate natural language inputs into database queries
- do not cover only words but also cover gaps of words

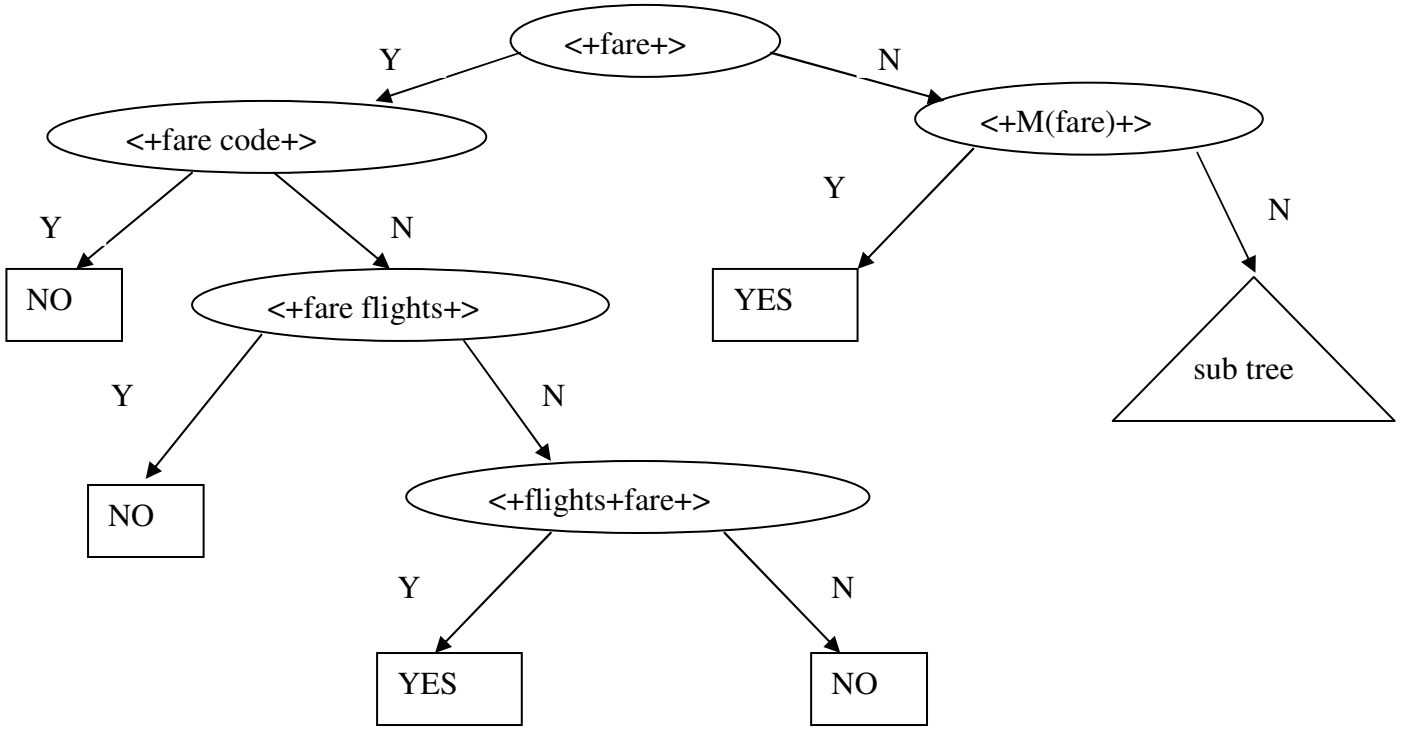
Single-Symbol SCTs

- associate each node with "Known Structure" (KS)
- KS contains symbols, words and gaps of words
- symbols < and > represent start and end of a sentence
- + indicates a gap of at least one word
- M(w) indicates multiple occurence of word w

ATIS example

Input:show me fare flights to Aachen

- single-symbol SCT



Building SCTs

three key elements:

1. a set of possible Yes or No questions that can be applied to data items

2. a strategy for selecting the best question at any node depending on the training data (Gini impurity)

3. a good method for pruning the trees in order to prevent over fitting

Building SCTs

generation:

- data item = symbol (word sequence)
- each node associated with regular expression (KS)
- root node : < + >
- questions generated by operations on gaps

Building SCTs

given + in KS and lexical item wi

four expressions are obtainable:

1. wi
2. wi+
3. +wi
4. +wi+

- at the root ks is <+>
  <wi > , <+ wi > ,< wi +>,<+ wi +> are generated

- each becomes a potential question

Building SCTs

Gini criterion:

- use Gini impurity to choose question
- change in impurity at node T

  $\Delta I(T) = i(T) - p_Y * i(YES) - p_N * i(NO)$

  - i(T) is the Gini impurity at node T
  - YES and NO are the child nodes of T
  - p_Y is probability of yielding a YES child node
  - p_N is probability of yielding a NO child node

Building SCTs

- The gini impurity any node T is given as

$$i(T) = -\sum_{i \neq j}(p(w_i)\, p(w_j))$$

  where $p(w_i)$ is the fraction of patterns at node T that are in category w_i

- best question for node T
  - question with greatest drop in impurity

Set-Membership SCTs

- similar to single-symbol SCTs
- ask for the presence or absence of any member of a set of words

SCT growing algorithm

1. *create root node and set known structure (ks) to <+>*
2. *select best question for node using Gini impurity*
3. *test node to see if there are further splits*
4. *if not set to leaf node, else generate YES and NO child nodes*
5. *set YES node to current and go to 2*
6. *set NO node to current and go to 2*

- *Steps 5 and 6 become independent paths to follow.*

Computational complexity

- D: total number of sentences
- L: upper limit of the number of words in a sentence
- V: size of the vocabulary

- for at most L words in a sentence, there are at most 4V meaninful questions

- rough upper bound:
  - 4*L*V

Computational complexity

for $D<4*L*V$

Single symbol: $O(D^3*L^2*V)$

Set membership: $O(D^3*L^2*V^2)$

- for $D>4*L*V$

single symbol: $O(D^2*L^3*V^2)$

set membership: $O(D^2*L^3*V^3)$

Classification of substrings

- why?

  - to classify multiple occurence of substrings

- submit same sentence as many times as number of substrings to be classified

- mark current substring with symbol (*)

- consider next substring

Example:

question: show me flights from Aachen, no sorry, from Bonn to Düsseldorf stopping over in Köln?

changed to: show me flights from CIT, no sorry, from CIT to CIT stopping over in CIT?

classify first CIT: submit show me flights from *CIT, no sorry, from CIT to CIT stopping over in CIT to SCT

second CIT: submit show me flights from CIT, no sorry, from *CIT to CIT stopping over in CIT to SCT

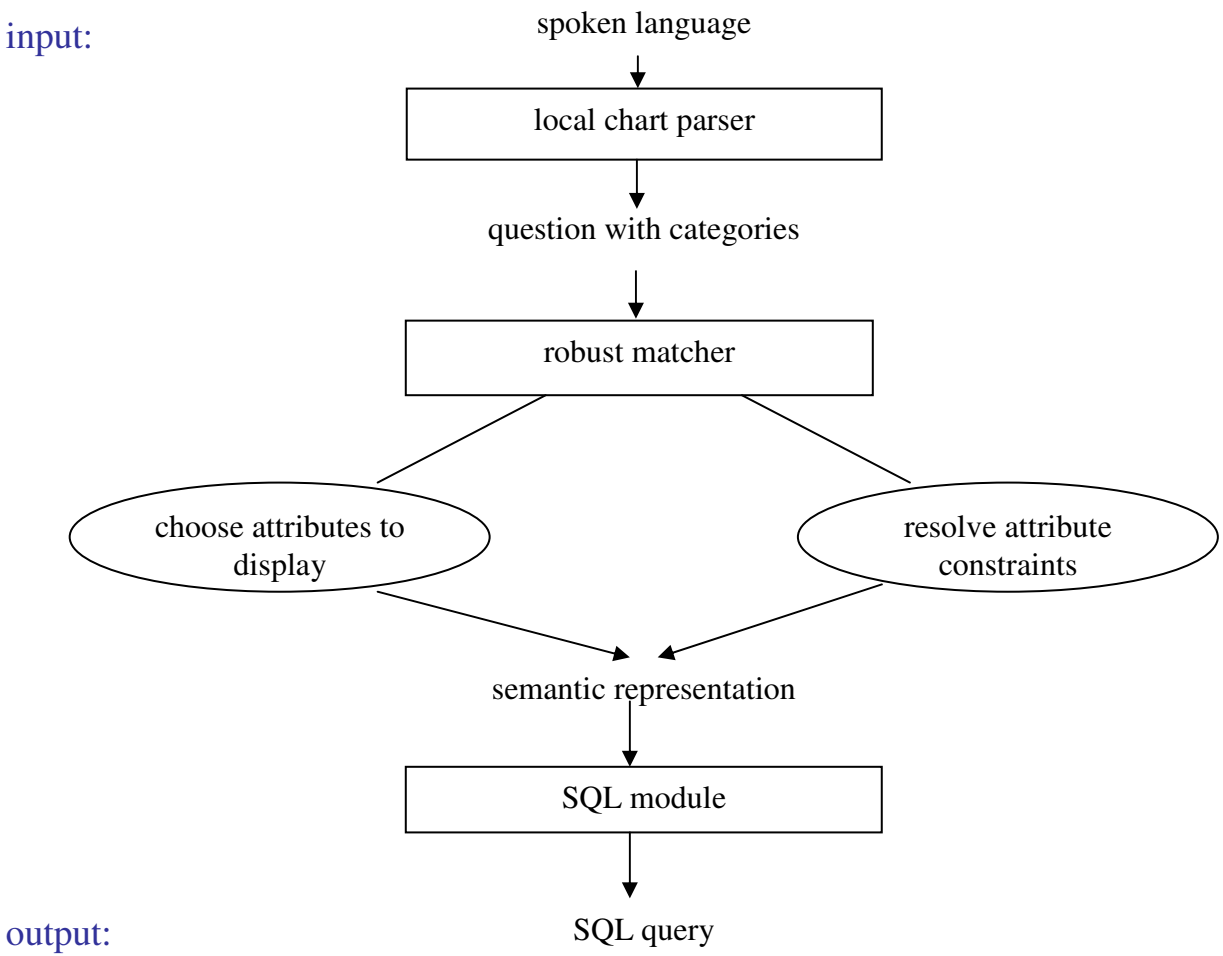
ATIS evaluation for spoken dialogue systems

- Air Traffic Information System
- DARPA-sponsored project for speech recognition and understanding
- translate spoken questions about air travel into a SQL database query
- one participating system was CHANEL which used SCT

CHANEL

- by the Centre de recherche informatique de Montreal (CRIM) and partially by Institute for Robotics and Intelligent Systems, Canada
- CRIM Hybrid Analyzer for natural language
- uses an SCT-based linguistic analyzer for NLU
- generates database queries for natural language based inputs

CHANEL Architecture



Local chart parser

- looks for words or phrases which carry important information
- replace with three-letter symbols carrying the associated meaning
- passed on to the robust matcher
- recovered to produce sql query

Robust matcher

has two parts:

1. generates a list of attributes to be displayed

2. resolves the constraints by using
   role assignments SCTs to assign roles

A Comparison to other approaches

- Class ‘A‘ sentences: do not depend on previous questions
- Class ‘D‘ sentences : depend on previous questions
- ATIS 1: SPREC (speech recognition performance)
- ATIS 2: NL (natural language understanding of written text)
- ATIS 3: SLS (spoken language understanding)

November 1992 NL ‘A’ Benchmarks

test data: 3248 class ‘A‘ sentences from ATIS 2

systems	method used	NL ‘A’ un weighted error [%]	NL ‘A’ weighted error [%]
ATT	HMM	19.5	34.7
BBN	CFG	11.1	15.7
CHANNEL	SCT	30.2	40.5

December 1993 NL ‘A’ Benchmarks

test data: 5501 class ‘A‘ sentences from ATIS 2

systems	method used	NL ‘A’ un weighted error [%]	NL ‘A’ weighted error [%]
ATT	HMM	7.4	14.7
BBN 1	CFG	9.6	19.0
BBN 2	CFG	16.1	31.9
CHANNEL(o)	SCT	21.7	42.2
CHANNEL(d)	SCT	12.3	23.9

Summary

- natural language database queries using classification and regression trees
- NLU component was needed
- semantic classification trees were used for NLU component
- ATIS as test bed

THANK YOU