Automatically Labeling the Inputs and Outputs of Web Services

Kristina Lerman
Anon Plangprasopchok
Craig Knoblock
USC Information Sciences Institute

CALO Intelligent Office Assistant

Find hotels

Select hotel by price, features and reviews

Check weather forecast

Find flights

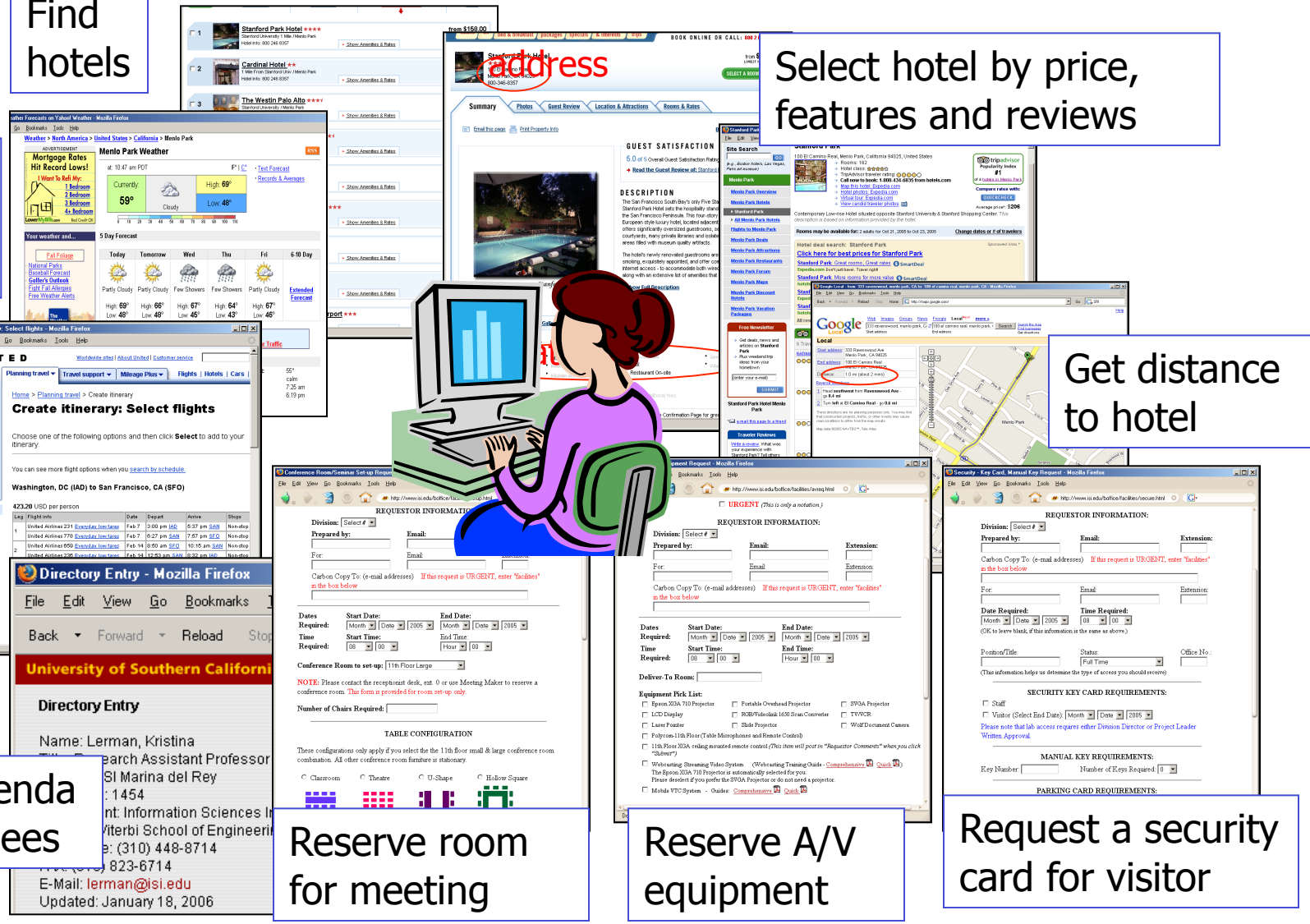
Get distance to hotel

Email agenda to attendees

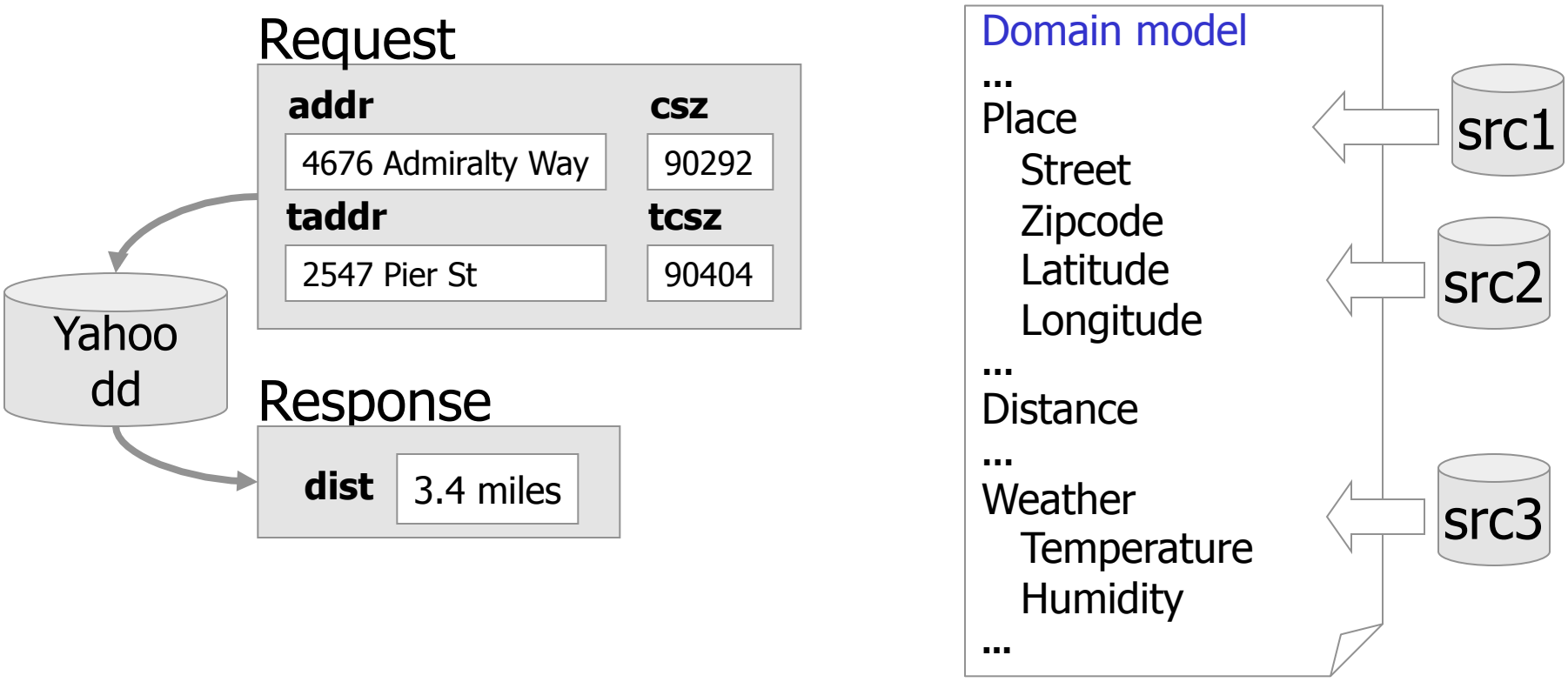
Reserve room for meeting

Reserve A/V equipment

Request a security card for visitor



Example: Using Yahoo Distance Source



yahoo_dd(addr,csz,taddr,tcsz,dist) →
distanceInMiles(Street, Zipcode, Street, Zipcode, Distance)

Information Integration

Information integration systems provide seamless access to heterogeneous information sources

- Today…
  - User must manually model an information source by specifying
    - Semantics of the input and output parameters
    - Functionality (operations) of the source
- Tomorrow …
  - Automatically model new sources as they are discovered
  - Alternative solution: standards (Semantic Web, …)
    - Slow to be adopted
    - Info providers may not agree on a common schema

Modeling Information Sources

- Research problem: Given a new source, automatically model it
  - Learn semantics of the input and output parameters (semantic labeling)
  - Learn operations it applies to the data (inducing functionality) (Carman & Knoblock, 2005)
- Focus on semantic labeling problem
  - Applied to Web services
    - Metadata readily available
    - Easy to extract data
  - Can be extended to RSS and Atom feeds, etc.

Web Services

Web services attempt to provide programmatic access to structured data

- Web service description (WSDL) file defines
  - Input and output parameters
  - Operations syntax

```
-<s:complexType name="ZipCodeCoordinates">
  <s:element name="LatDegrees" type="s:float"/>
  <s:element name="LonDegrees" type="s:float"/>
-<wsdl:message name="GetZipCodeCoordinatesSoapIn">
 <wsdl:part name="zip" type="s:string"/>
-<wsdl:message name="GetZipCodeCoordinatesSoapOut">
 <wsdl:part name="GetZipCodeCoordinatesResult" type="tns:ZipCodeCoordinates"/>
```

Service description is *syntactic* – client needs a priori understanding of the *semantics* to invoke the service

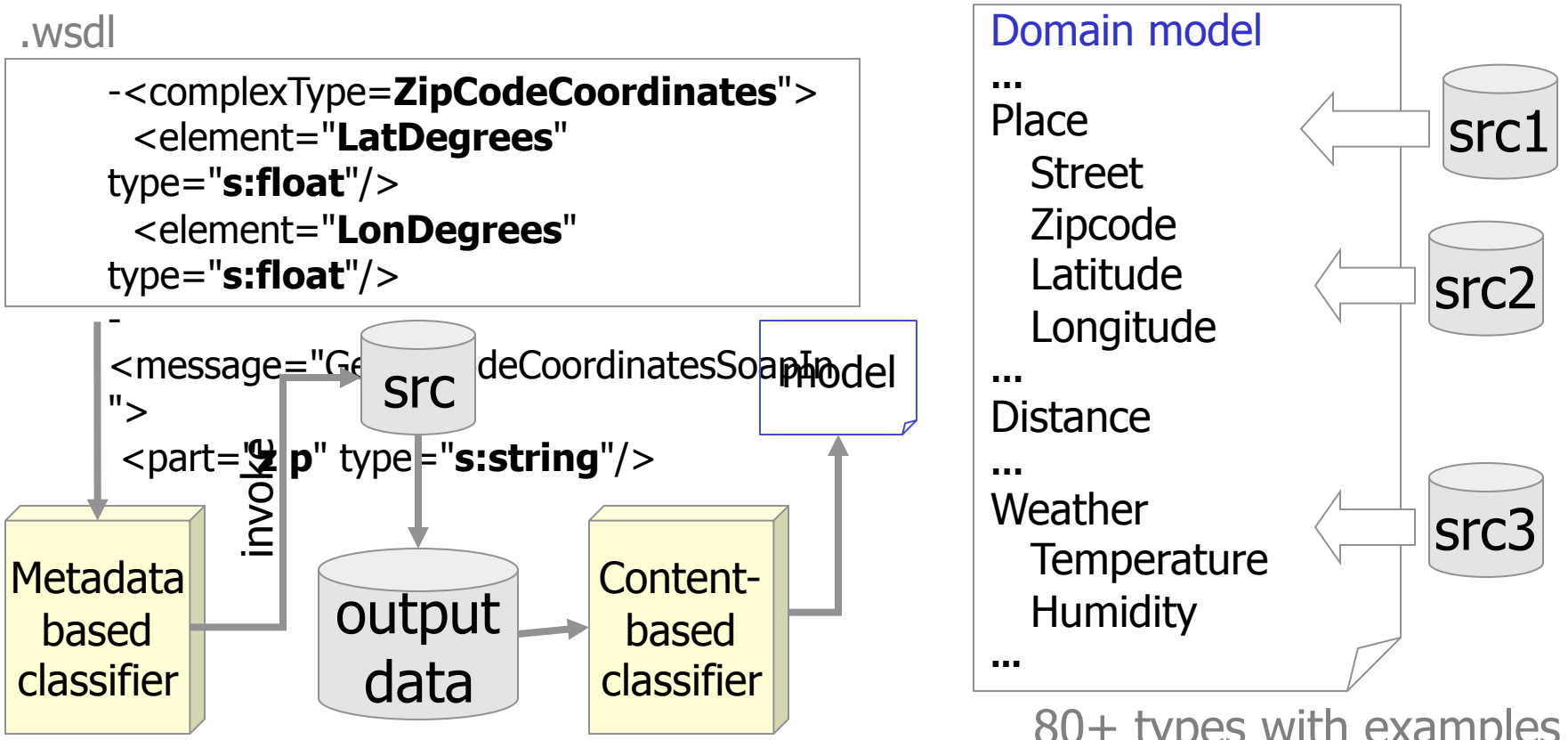
Our Approach to Semantic Labeling

We leverage existing knowledge to learn semantics of data used by Web services

- Background knowledge captured in a lightweight domain model
  - 80+ semantic types: Temperature, Zipcode, Flightnumber …
  - Populated with examples of each type (from known sources)
  - Expandable
- Semantic labeling: mapping inputs/outputs to types in the domain model
  - Map input types based on metadata in WSDL file
  - Test by invoking Web service with examples of these types
  - Map output types based on content of data returned

Our Approach to Semantic Labeling

Leverage existing knowledge to learn semantics of data used by Web services



Contributions

- Metadata-based classification
  - Logistic Regression classifier to label data used by Web services using metadata in the WSDL file
  - Automatically verify classification results by invoking the service
- Content-based classification
  - Label output data based on their content
- Automatically label live services
  - Weather and Geospatial domains
  - Combine metadata and content-based classification

Metadata-based Classification

- Observation 1

  Similar data types tend to be named with similar words, and/or belong to operations that have similar name
  - Treat as (ungrammatical) text classification problem
  - Approach taken by previous works
- Observation 2

  The classifier must be a soft classifier
  - Instance can belong to more than one class
  - Rank classification results

Independence Assumption

- Naïve Bayes classifier
  - Used to classify parameters used by Web services (Hess & Kushmerick, 2004)
    - Each input/output parameter represented by a term vector **t**
  - Based on independence assumption
    - Terms are independent from each others given the class label D (semantic type)

      $P(D|\boldsymbol{t}) \leftarrow \Pi_i \, P(t_i|D)$
    - Independence assumption unrealistic for Web services
      - e.g., "TempFahrenheit": "Temp" and "Fahrenheit" often co-occur in the Temperature semantic type
- Logistic regression avoids the independence assumption
  - Estimates probabilities from the data

    $P(D|\boldsymbol{t}) = \text{logreg}(\boldsymbol{wt})$

Metadata-based Classification Evaluation

- Data collection
  - Data extracted from 313 WSDL files from Web service portals (bindingpoint and webservicex)
- Data processing
  - Names were extracted from operation, message, datatype and facet (predefined option)
  - Names tokenized into individual terms
- 10,000+ data types extracted
  - Each one assigned to one of 80 classes in geospatial and weather domains (e.g. latitude, city, humidity).
  - Other classes treated as "Unknown" class

Evaluation Results

- Both Naïve bayes and Logistic regression were tested using 10-fold cross validation

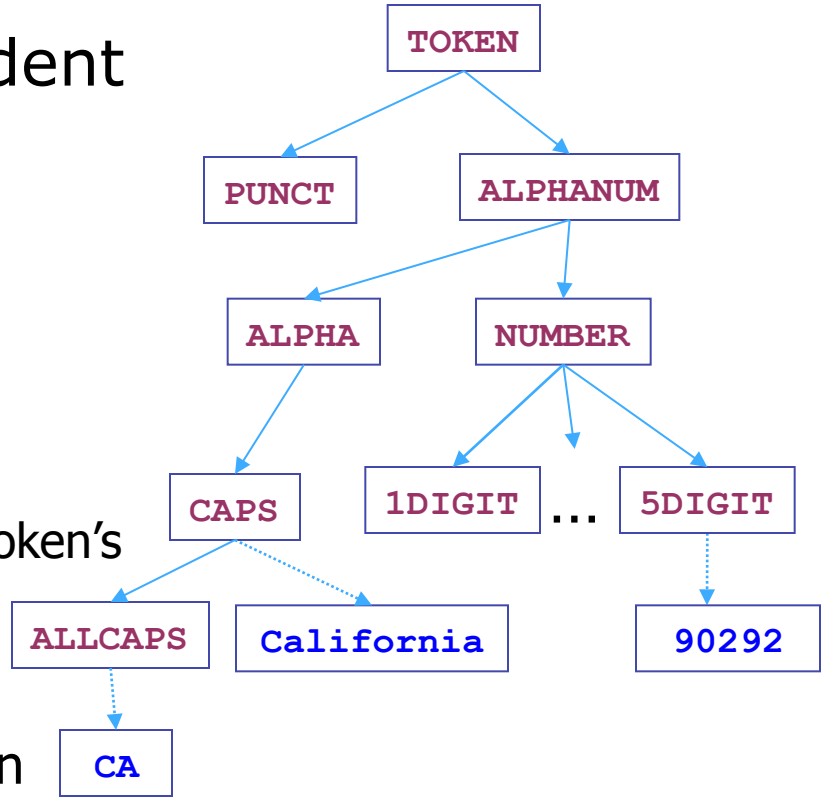
Classifier	Top1	Top2	Top3	Top4
Naïve Bayes	0.65	0.84	0.88	0.90
Logistic Regression	0.93	0.98	0.99	0.99

Content-based Classification

- ✓ Idea: Learn a model of the content of data and use it to recognize new examples

Developed a domain-independent language to represent the structure of data

- Token-level
  - Specific tokens
  - General token types
    - based on syntactic categories of token's characters
- Hierarchy of types
  - allows for multi-level generalization



Patterns for Describing Data

- Pattern is a sequence of tokens and general types
  - Phone numbers

Examples	Patterns
310 448-8714	
310 448-8775	[(310) 448 – 4DIGIT]
212 555-1212	[(3DIGIT) 3DIGIT – 4DIGIT]

- Algorithm to learn patterns from examples
- Patterns for all semantic types in the domain model

Patterns for Semantic Labeling

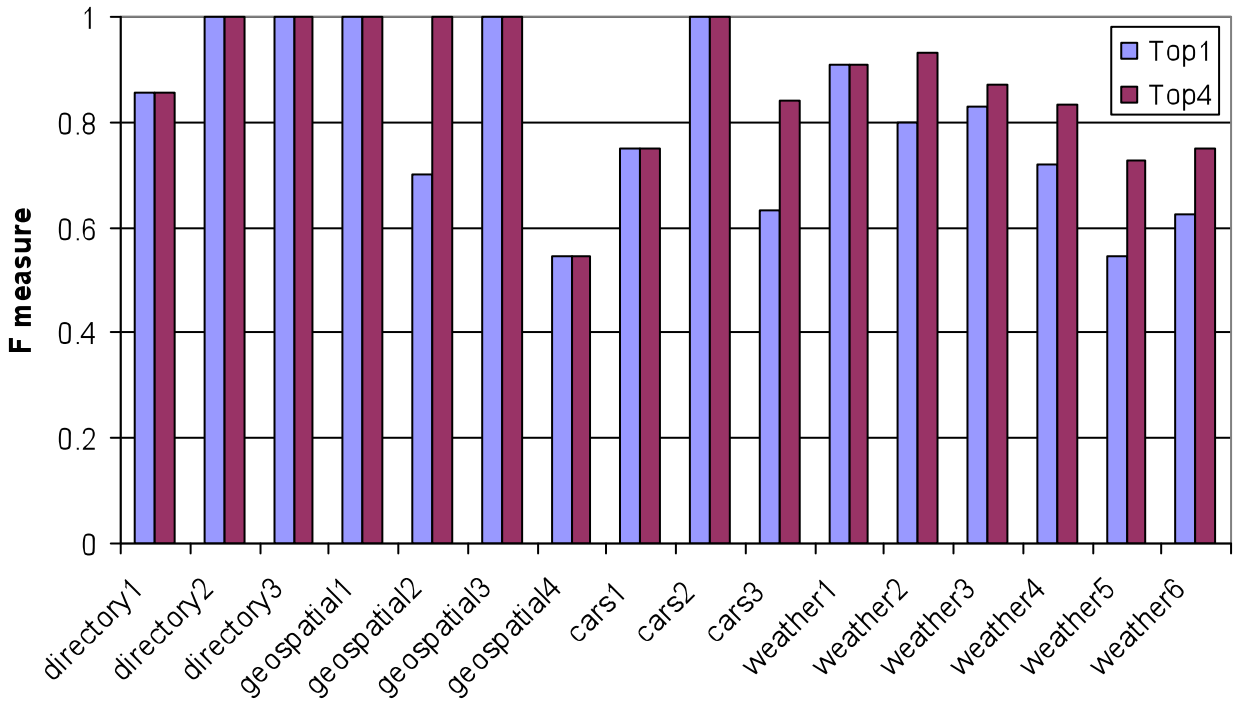
- Use learned patterns to map new data to types in the domain model
  - Score how well patterns associated with a semantic type describe a set of examples
    - Heuristics include:
      - Number of matching patterns
      - How specific the matching patterns are
      - How many tokens of the example are left unmatched
  - Output four top-scoring types

Semantic Labeling Evaluation

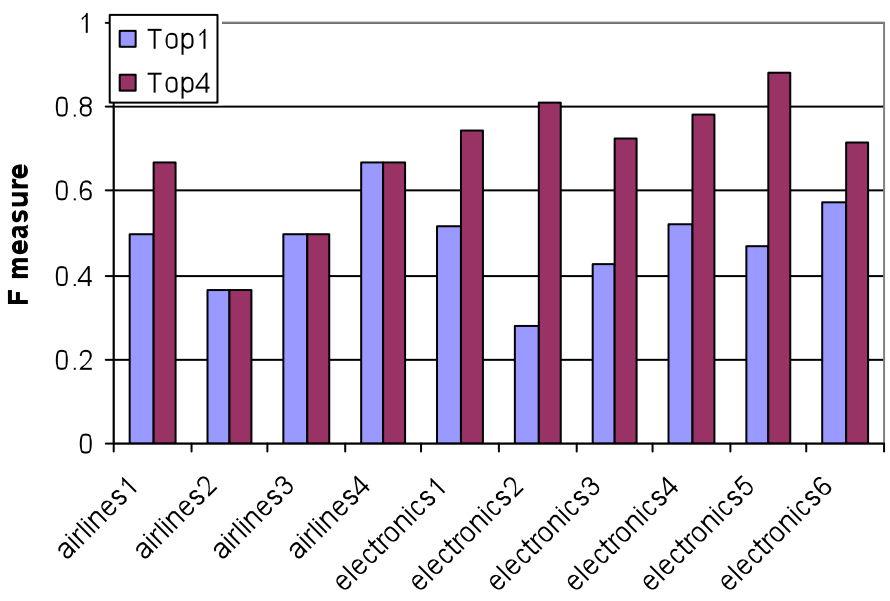
Information domains and semantic types

- Weather Services
  - Temperature, SkyConditions, WindSpeed, WindDir, Visibility
- Directory Services
  - Name, Phone, Address
- Electronics equipment purchasing
  - ModelName, Manufacturer, DisplaySize, ImageBrightness, …
- UsedCars
  - Model, Make, Year, BodyStyle, Engine, …
- Geospatial Services
  - Address, City, State, Zipcode, Latitude, Longitude
- Airline Flights
  - Airline, flight number, flight status, gate, date, time

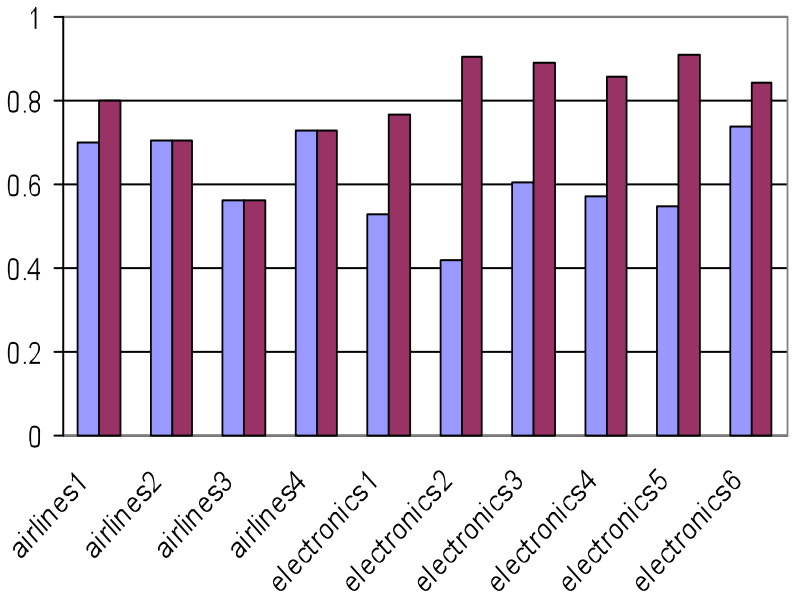
Evaluations Results



Evaluations Results 2



Using all semantic types in classification



Restricting semantic types to domain of the source

Empirical Validation

- Automatically model the inputs and outputs used by Geospatial and Weather Web Services
  - Given the WSDL file of a new service
  - 8 services (13 operations)
- Results

classifier	total	correct	accuracy
input parameters			
metadata-based	47	43	0.91
output parameters			
metadata-based	213	145	0.68
content-based	213	107	0.50
combined	213	171	0.80

Conclusion

- Two algorithms for semantic labeling of data used by Web services
  - Metadata-based classification
    - Semantically label input and output parameters
  - Content-based classification
    - Semantically label output parameters
- Active testing
  - Invoke the service to verify classification results
  - Automatically verify classification results

Related Research

- Metadata-based classification of data types used by Web services and HTML forms (Hess & Kushmerick, 2003)
  - Naïve Bayes classifier
  - No invocation of services
- Woogle: Metadata-based clustering of data and operations used by Web services (Dong et al, 2004)
  - Groups similar types together: Zipcode, City, State
  - Cannot invoke services with this information
- Schema matching
  - Map instances of data from one database to another
    - Use metadata (schema names) and content features (word frequencies) (Li & Clifton 2000; Doan, Domingos & Halevy 2001)
    - No invocation – data is available

Future Directions

- Represent complex data types
  - Date
    - June 22, 2006
    - 06/22/06
    - Jun 22
  - But, we can correctly recognize Month, Day, Year
- Automate invocation and data collection
- Combine with ongoing work on modeling functionality of Web services

  Svc(Zipcode, TempF, TempF, TempF) →
  CurrentWeather(Zipcode, TempF, HiTemp, LoTemp)