Data and Schema Matching

"A survey of approaches to automatic schema matching,"
by Rahm and Bernstein, *VLDB Journal* 10(4)
"Reconciling Schemas of Disparate Data Sources:
A Machine Learning Approach," by Doan et al, SIGMOD 01

Zachary G. Ives
University of Pennsylvania

April 16, 2003

Some slides on LSD courtesy of
Prof. AnHai Doan, UIUC

Administrivia: Schedule Correction

Wednesday: wrap-up and discussion of research in data sharing

Monday 4/21: your 5-minute project presentations

- We get to hear about the cool projects you've been working on!
- Slides are allowed (but not required)
  - What did you do?
  - What were the hard problems?
  - How are you solving them?
  - How are you evaluating your work?

Take-home final exam will be distributed Monday

Will likely be 3-4 essay questions; open-book, open-notes

Next Friday, 4/24, 11AM (instead of 4/23 lecture): Talk on schema matching by Prof. AnHai Doan, UIUC

Deadlines: final exam and project due before 6PM, 5/2

Data Sharing

We've been discussing sharing *semantically rich* data across the web:

- Data integration and data warehousing
- Semantic web and peer data management
- Same techniques apply to problems like e-commerce

- In all of these, there are huge challenges addressed by:
  - Data cleaning (very briefly)
  - Schema matching (in more detail)

Data Cleaning

Actually, refers to several possible types of problems in warehousing/integration/DBs:

- Data is “dirty”, i.e., has typos
- Data is ambiguous/imprecise
- Correspondences between objects in different representations is unknown

Two options:

- Offline, find items we think are the same and merge them together
- Or, online or offline, perform “approximate joins” and similar operations

Dirty or Imprecise Data

This is often something like WHIRL

- What are some key attributes of this approach?

Can also use data mining and probabilistic machine-learning approaches here

- Many AI folks are working on this problem
- Often requires multiple passes over the data
  - Look for “close matches” or “closest matches”

Finding Correspondences

Many different methods

- Most overlap with “imprecise data” category

Challenges:

- Very expensive to compute such things
- How do we define mappings in our query language?
  - Generally use “concordance relations”

Better if we can compute correspondences and mappings at the schema level

(There may be a concordance relation/function)

Schema Matching

A problem that has been the focus of work since the 1970s, in the AI, DB, and knowledge representation communities

- Today, people are realizing that this is a core problem to most of the things they want to do:
  - E-commerce exchanges
  - Data integration/warehousing
  - Semantic web
- Goal: make it (mostly) generic and reusable in different application domains

  Generally use probabilistic, machine-learning-based techniques

What's the Schema Matching Problem?

Given two schemas, S1 and S2:

- Create a mapping between the two:
  - Mapping might be directional or symmetric
  - Mapping might be in the form of a query, or it might be a set of expressions between items in each schema
- Many people simply look at finding correspondences between elements as the first step

  Correspondences are often informally justified

A Matching Example

S1 elements:

- Cust
  - C#
  - CName
  - FirstName
  - LastName

S2 elements:

- Customer
  - CustID
  - Company
  - Contact
  - Phone

A Matching Example

S1 elements:		S2 elements:
Cust		Customer
C#	=	CustID
CName	=	Company
FirstName	=	Contact
LastName	+	
		Phone

What Goes into a Match Decision?

- Data values
  - May find common patterns or phrases in data values
- Element names
- Constraint information
- Structural information
- Domain knowledge: Synonyms, related terms, etc.
- Cardinality relationship between elements

What are implications for instance-level vs. schema-level?

What Makes Matching Complicated 1/2

How do we deal with partial and composite matches?

Contact
- email
- street
- city
- state
- zip
- hphone
- wphone
- fax

BillAddress
- street
- city
- stateOrProvince
- zipOrRegion
- country

phone
fax

ShipAddress
- street
- city
- stateOrProvince
- zipOrRegion
- country

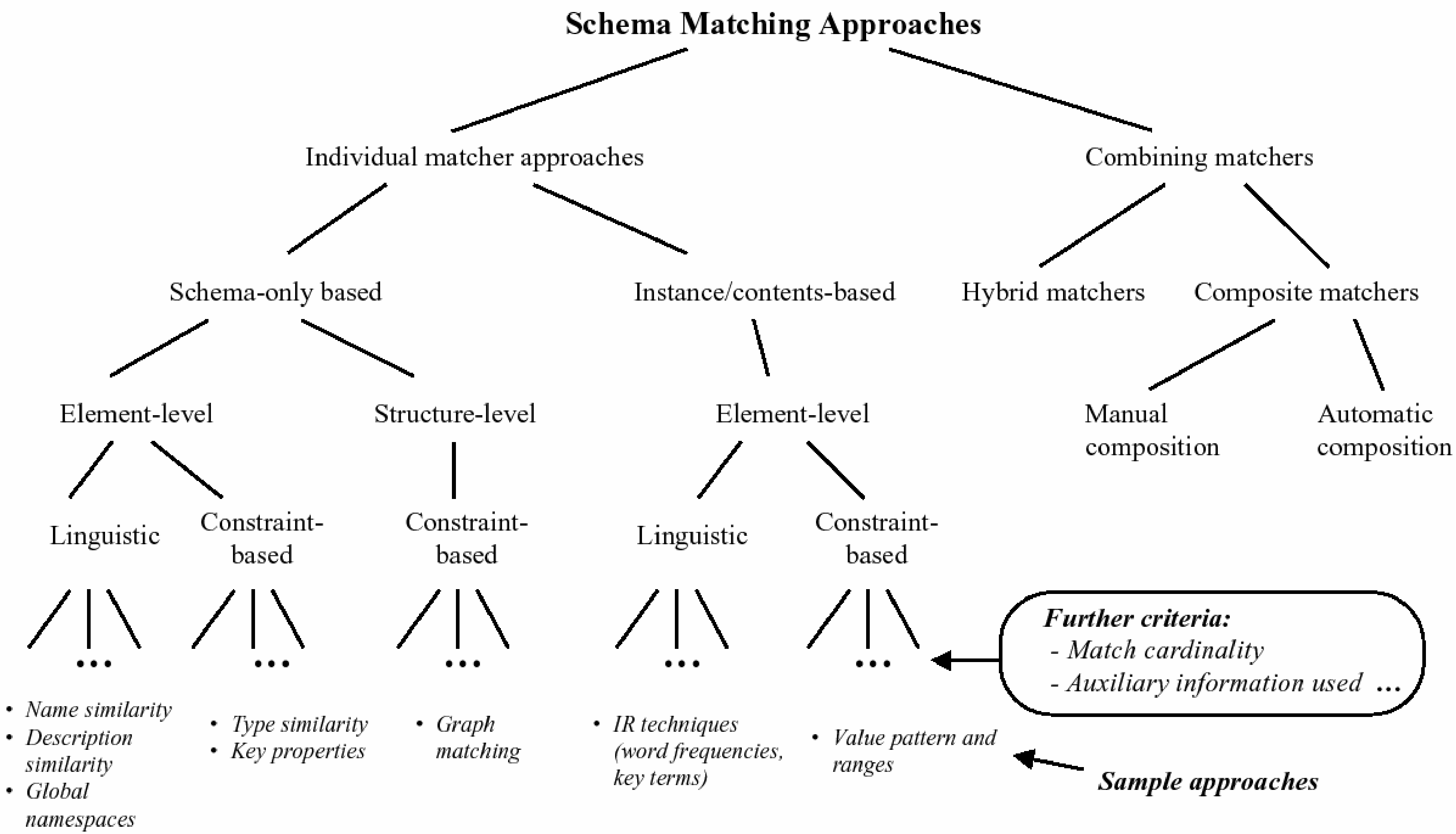
What Makes Matching Complicated 2/2

May have different levels of representation:

MealsRequested
  breakfast
  lunch
  dinner

MealList: set of {
  time, order
}

Approaches People Have Used



An Example Matcher:
LSD (Doan, Domingos, Halevy)

A "composite matcher" for mapping data sources to a mediated schema

- Train with mappings from a few sources; let it run on the rest
- Tries to combine information from many different approaches – "multi-strategy learning"
- Favors the approaches that give the best results
  - Uses a machine learning approach called "stacking"

Example Matching Problem

Mediated schema

price	agent-name	agent-phone	office-phone	description
listed-price	contact-name	contact-phone	office	comments

Schema of realestate.com

"office"
occurs in name
=> office-phone

realestate.com

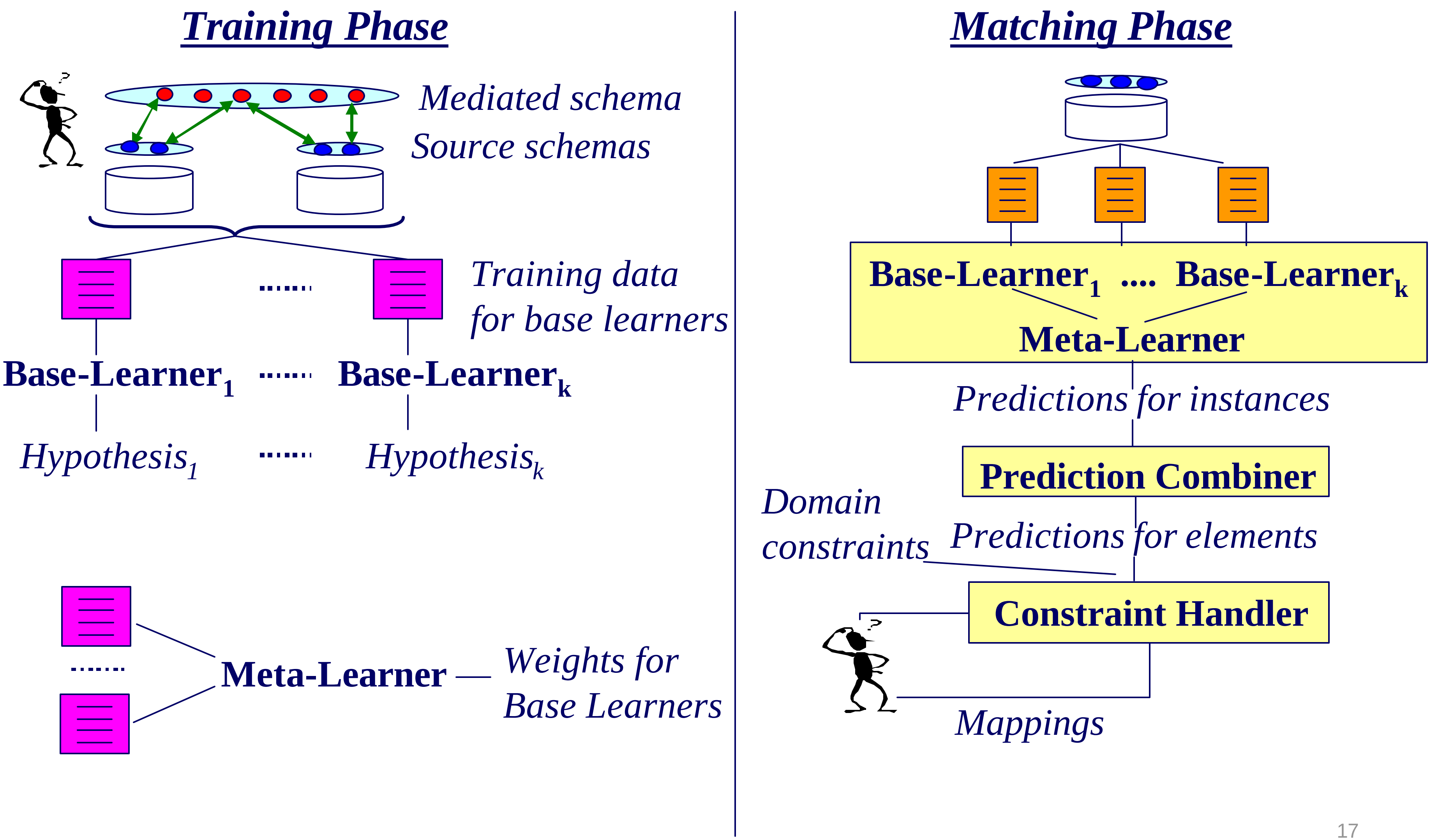
listed-price	contact-name	contact-phone	office	comments
\$250K	James Smith	(305) 729 0831	(305) 616 1822	Fantastic house
\$320K	Mike Doan	(617) 253 1429	(617) 112 2315	Great location
....	....	....	....	....

"fantastic" & "great"
occur frequently in
data instances
=> description

homes.com

sold-at	contact-agent	extra-info
\$350K	(206) 634 9435	Beautiful yard
\$230K	(617) 335 4243	Close to Seattle

The LSD Architecture



Training the Base Learners

Mediated schema

address	price	agent-name	agent-phone	office-phone	description

realestate.com

location	price	contact-name	contact-phone	office	comments
Miami, FL	$250K	James Smith	(305) 729 0831	(305) 616 1822	Fantastic house
Boston, MA	$320K	Mike Doan	(617) 253 1429	(617) 112 2315	Great location
…….	…….	…….	…….	…….	…….

Name Learner

(“location”, address)
(“price”, price)
(“contact name”, agent-name)
(“contact phone”, agent-phone)
(“office”, office-phone)
(“comments”, description)

Naive Bayes Learner

(“Miami, FL”, address)
(“$250K”, price)
(“James Smith”, agent-name)
(“(305) 729 0831”, agent-phone)
(“(305) 616 1822”, office-phone)
(“Fantastic house”, description)
(“Boston,MA”, address)
…….

Stacking

Training

- uses training data to learn weights
- one for each (base-learner,mediated-schema element) pair
- weight (Name-Learner,address) = 0.2
- weight (Naive-Bayes,address) = 0.8

Matching: combine predictions of base learners

- computes weighted average of base-learner confidence scores

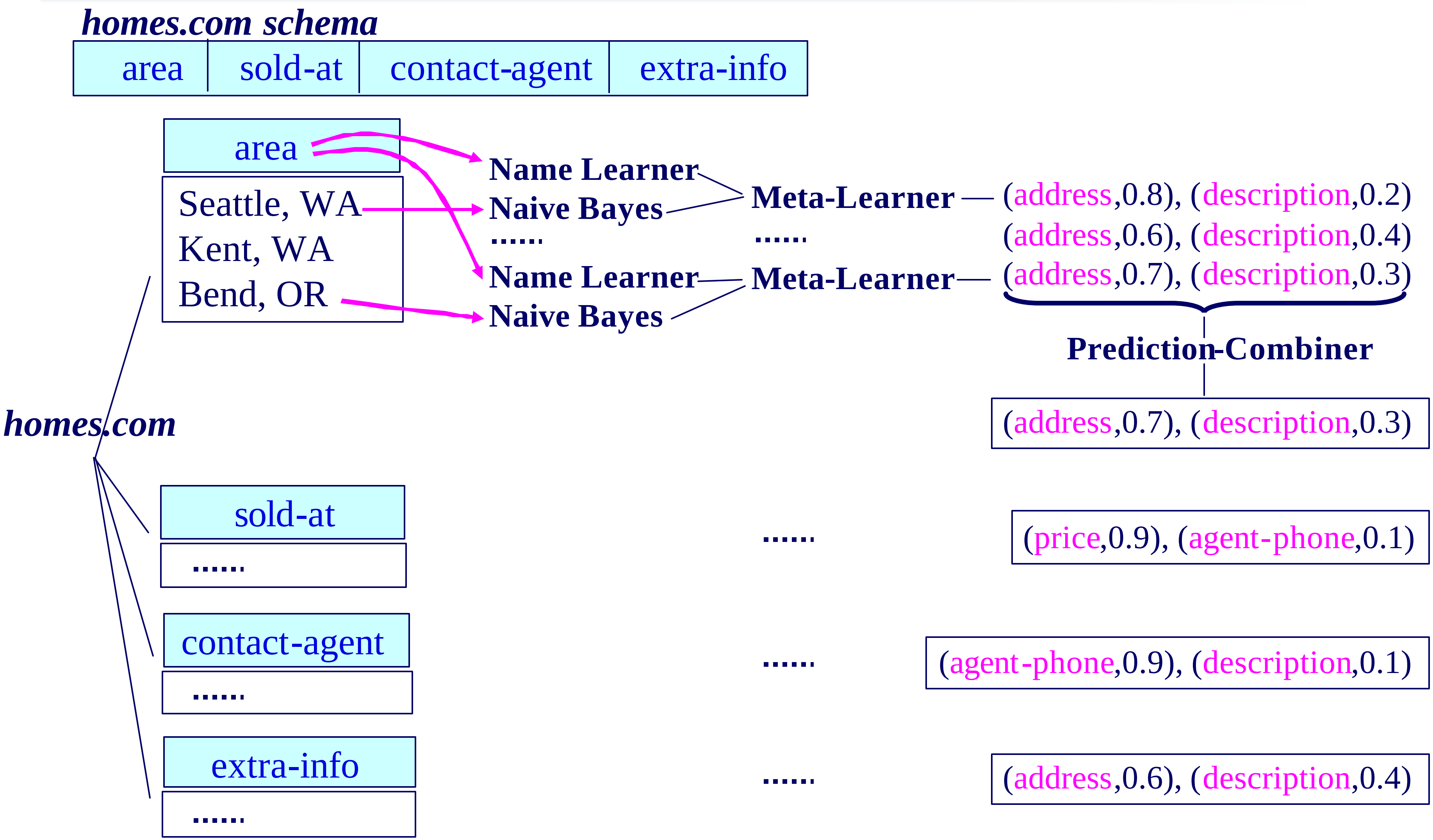
area

Seattle, WA
Kent, WA
Bend, OR

Name Learner — (address,0.4)
Naive Bayes — (address,0.9)

Meta-Learner — (address, 0.4*0.2 + 0.9*0.8 = 0.8)

Combining Info from the Learners



Does It Work?

LSD's accuracy: 71 - 92%

This is pretty good but far from perfect:

- Sometimes, even a human may not do better:
  - Some matches need an expert to determine
  - Some things are inherently ambiguous
- But sometimes a human can do better!
- This helps simplify the process of finding matches, but it's not a panacea

Current hot topic: how to use previous mappings to "bootstrap" new mapping creation