Transparent Machine Learning for Information Extraction

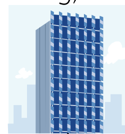
Laura Chiticariu
Yunyao Li
Fred Reiss
IBM Research - Almaden

EMNLP 2015
CONFERENCE ON EMPIRICAL METHODS IN NATURAL LANGUAGE PROCESSING
LISBON

Motivation

Case Study 1: Social Media

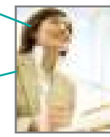
Product catalog, Customer Master Data, …



Social Media

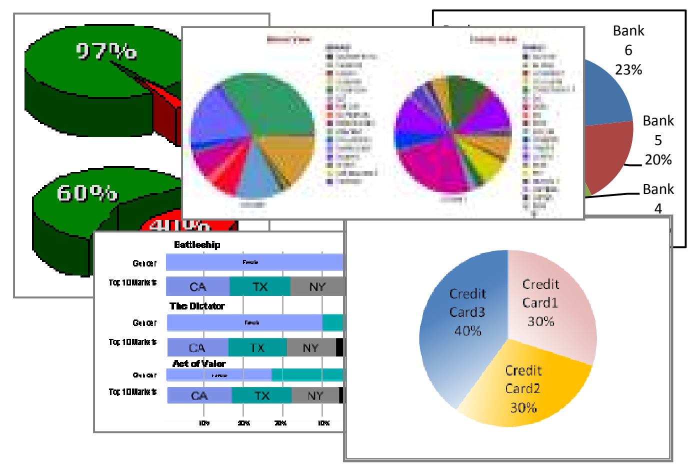
Information Extraction

- Personal Attributes
- Relationships
- Life Events
- Products Interests



360º Profile

Statistical Analysis, Report Gen.



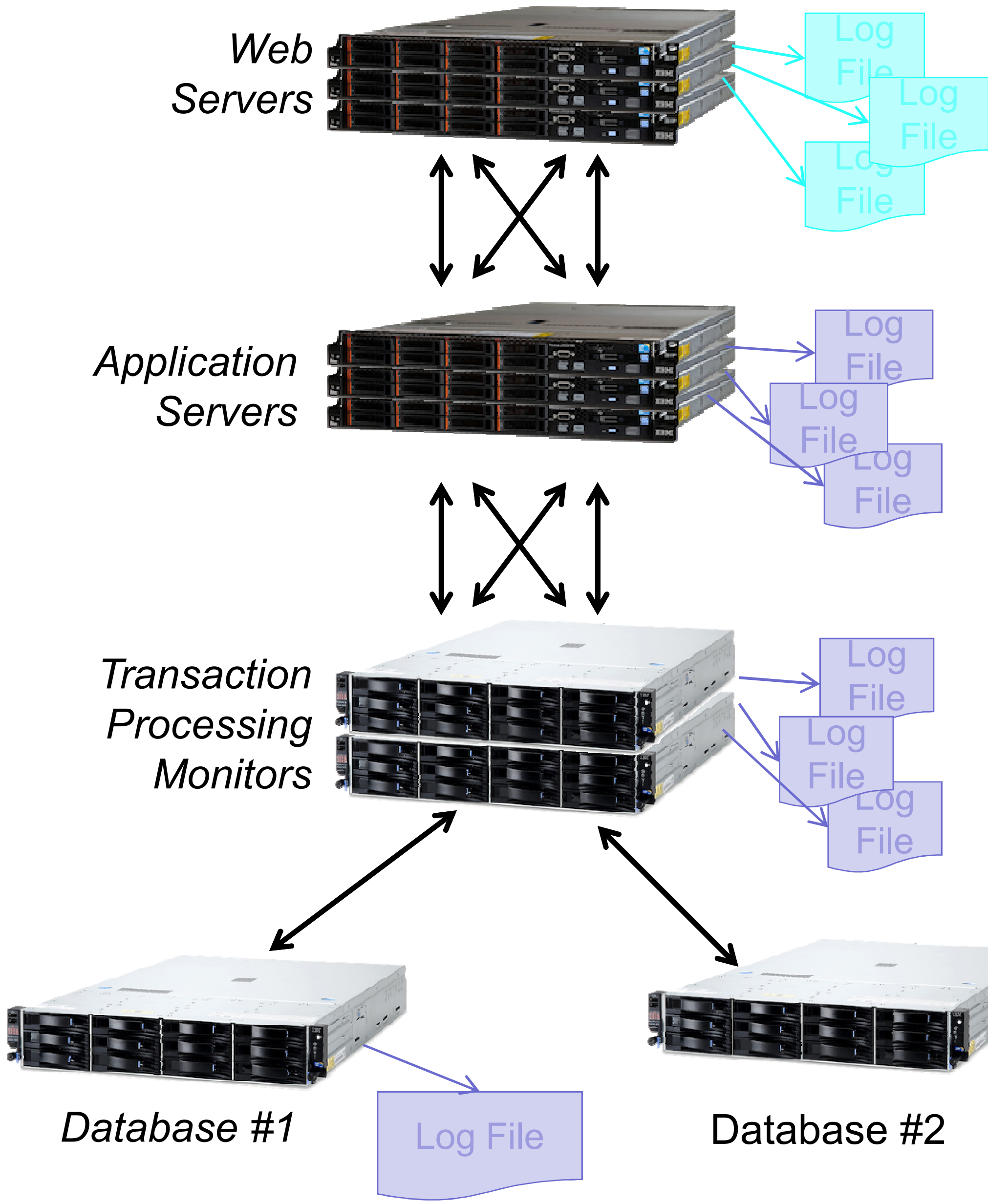
Complexity
Sarcasm, wishful thinking, …

Breadth
Buzz, intent, sentiment, life events, personal atts, …

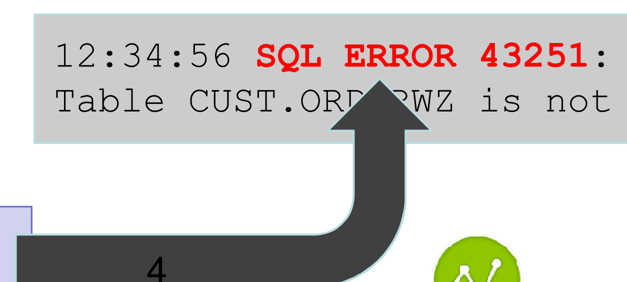
Scale
450M+ tweets a day, 100M+ consumers, …

Customer 360º

Case Study 2: Server Logs

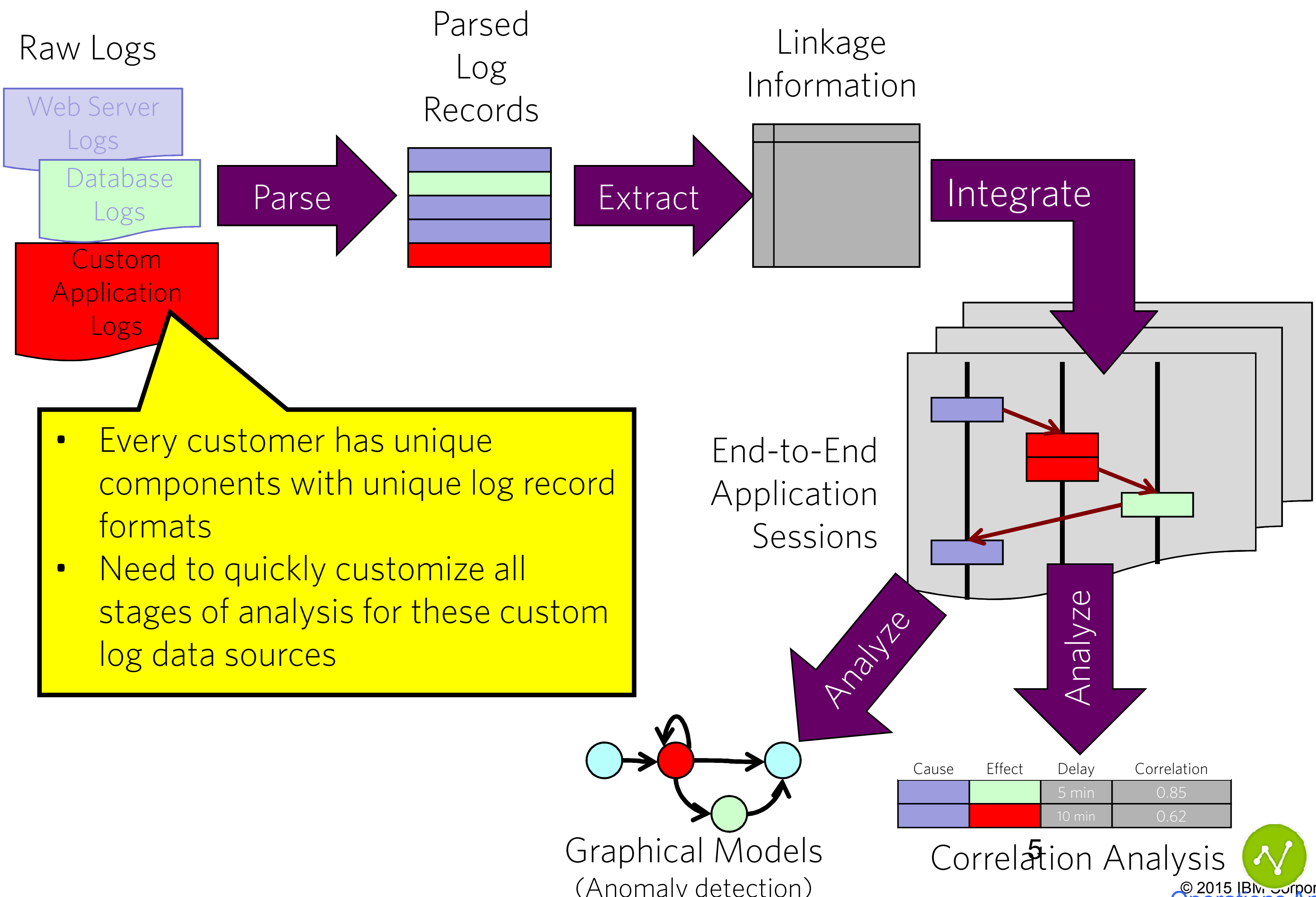


- Web site with multi-tier architecture
- Every component produces its own system logs
- An error shows up in the log for Database #2
- What sequence of events led to this error?



Operations Analysis

Case Study 2: Server Logs



- Every customer has unique components with unique log record formats
- Need to quickly customize all stages of analysis for these custom log data sources

Case Study 3: Sentiment Analysis for Analyst Research Reports

- Determine the sentiments expressed towards a financial entity or its aspects in financial research reports

Sentiment Mention	Sentiment Target	Sentiment Polarity	Entity Type	Sentiment Category	Aspect
We prefer HK Telecom from a long term perspective	HK Telecom	Positive	Company	Direct	n/a
Sell EUR/CHF at market for a decline to 1.31000	EUR	Negative	Currency	Direct	n/a
Sell EUR/CHF at market for a decline to 1.31000	CHF	Positive	Currency	Direct	n/a
Intel's 2013 capex is elevated relative to historical norms	Intel	Positive	Company	Indirect	Capex

- Handle different categories of sentiment mentions
  - **Direct**: Explicit recommendations
    - *Our current neutrals are on China/Hong Kong, Singapore, Indonesia and Thailand; underweight on Malaysia, Korea, Taiwan and India.*
    - *We prefer HK Telecom from a long term perspective.*
  - **Indirect**: Mention of a change in a key indicator that can be directly linked to a recommendation
    - *Intel's 2013 capex is elevated relative to historical norms*
    - *FHLMC reported a net loss of $2.5bn net loss for the quarter.*
  - **Implicit**: other sentiment mentions that are not direct recommendations or statements about a key economic indicator
    - *Taiwan is making continuous progress on trade and investment liberalization, which bodes well for its long-term economic prospects*
    - *Export outlook remains lackluster for the next 1-3 months.*

Requirements for IE in the Enterprise

- Scalability

Scalability Examples

- Social Media
  - Twitter has 450M+ messages per day; 1TB+ per day → 400+ TB per year
  - Add to it enterprise-specific Facebook, Tumblr, and tens of thousands of blogs/forums
- Financial Data
  - Regulatory filings can be in tens of millions and several TBs
- Machine data
  - One application server under moderate load at medium logging level → 1GB of app server logs per day
  - A medium-size data center has tens of thousands of servers → Tens of Terabytes of system logs per day

Requirements for IE in the Enterprise

- Scalability
- Expressivity

IBM

Expressivity Example: Varied Input Data

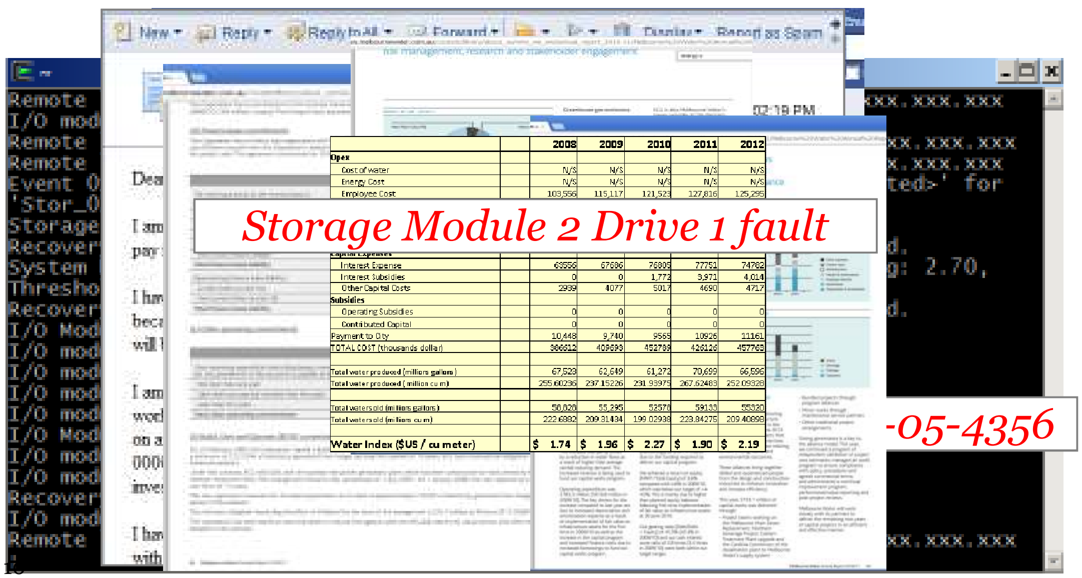
Customer 360º

Security & Privacy

Operations Analysis

Financial Analytics

Product:Hotel	Location:Orlando
Type:SSN	Value:400054356
Event:DriveFail	Loc:mod2.Slot1



Social Media

Medical Records

Email

Patents

Machine Data

News

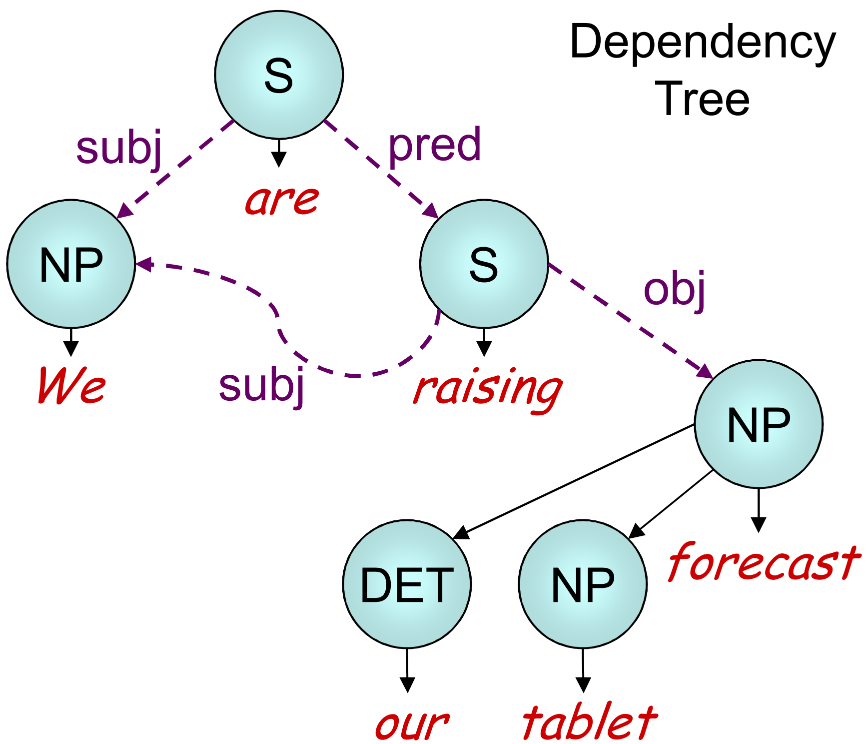
Financial Statements

CRM

Expressivity Example: Different Kinds of Parses

Natural Language

We are raising our tablet forecast.



Machine Log

`Oct 1 04:12:24 9.1.1.3 41865: %PLATFORM_ENV-1-DUAL_PWR: Faulty internal power supply B detected`

Time	Oct 1 04:12:24
Host	9.1.1.3
Process	41865
Category	%PLATFORM_ENV-1-DUAL_PWR
Message	Faulty internal power supply B detected

Expressivity Example: Fact Extraction (Tables)

IBM

PUBLIC UTILITIES BOARD AND ITS SUBSIDIARIES

STATEMENTS OF COMPREHENSIVE INCOME

Year ended 31 March 2012

	Note	GROUP 31 March 2012 S$'000	GROUP 31 March 2011 S$'000	BOARD 31 March 2012 S$'000	BOARD 31 March 2011 S$'000
Operating income	3	1,037,549	1,010,737	1,030,740	1,005,476
Operating expenses	4	(1,037,056)	(998,773)	(1,030,647)	(993,502)
Net operating income		493	11,964	93	11,974
Non-operating income	5	26,000	19,768	27,881	19,721
Net income before financing expenses and operating grants		26,493	31,732	27,974	31,695
Financing expenses	6	(108,030)	(103,608)	(108,030)	(103,608)
Net Loss before operating grants		(81,537)	(71,876)	(80,056)	(71,913)
Operating grants from government	13.1	199,035	185,218	199,035	185,218
Net income after grants and before contribution to government consolidated fund and taxation		117,498	113,342	118,979	113,305
Contribution to government consolidated fund and taxation	7	(20,230)	(19,269)	(20,231)	(19,261)
Net income after grants and after contribution to government consolidated fund and taxation		97,268	94,073	98,748	94,044
Other comprehensive income		–	–	–	–
Total comprehensive income for the year		97,268	94,073	98,748	94,044
Attributable to:					
Shareholder of the Board	20.3	97,268	94,073	98,748	94,044

Identify line item for Operating expenses from Income statement (financial table in pdf document)

Singapore 2012 Annual Report
(136 pages PDF)

4 OPERATING EXPENSES

	Note	GROUP 31 March 2012 S$'000	GROUP 31 March 2011 S$'000	BOARD 31 March 2012 S$'000	BOARD 31 March 2011 S$'000
Direct operating expenses					
- electricity		147,427	126,539	147,427	126,539
- manpower		177,901	185,272	177,852	185,128
- depreciation		264,431	254,436	264,431	253,753
- plant rental		10,071	24,801	10,071	24,801
- property tax		15,014	14,365	15,014	14,365
- maintenance and others	4.1	293,002	266,880	286,642	262,436
Indirect operating expenses					
- service departments' costs	4.2	129,210	126,480	129,210	126,480
	4.3	1,037,056	998,773	1,030,647	993,502

Identify note breaking down Operating expenses line item, and extract opex components

106 /

Expressivity Example: Sentiment Analysis

Analyst Research Reports

- Intel's 2013 capex is elevated at 23% of sales, above average of 16%
- IBM announced 4Q2012 earnings of $5.13 per share, compared with 4Q2011 earnings of $4.62 per share, an increase of 11 percent
- We continue to rate shares of MSFT neutral.
- FHLMC reported $4.4bn net loss and requested $6bn in capital from Treasury.
- Sell EUR/CHF at market for a decline to 1.31000…

Customer Surveys

- Not a pleasant client experience. Please fix ASAP.
- I'm still hearing from clients that Company A's website is better.
- X… fixing something that wasn't broken

Social Media

- Makin chicken fries at home bc everyone sucks!
- Bank X got me ****ed up today!
- Mcdonalds mcnuggets are fake as shit but they so delicious.
- You are never too old for Disney movies.
- We should do something cool like go to Z (kidding).

Requirements for IE in the Enterprise

- Scalability
- Expressivity
- Ease of comprehension

Ease of Comprehension: What not to do (1)

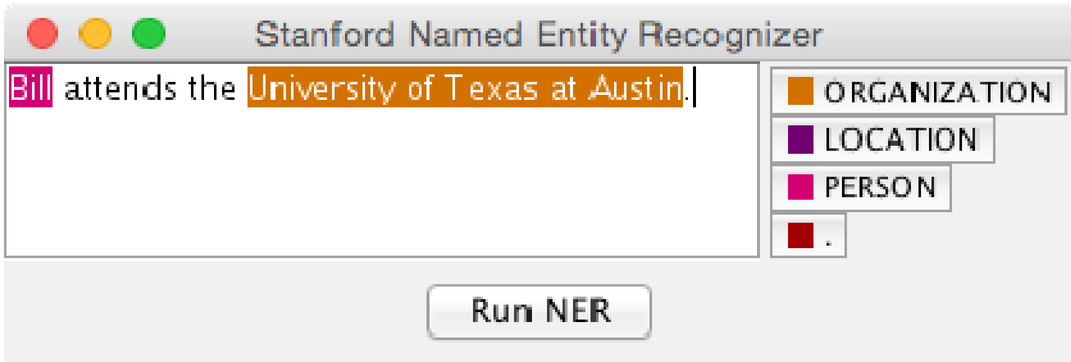
```
package com.ibm.avatar.algebra.util.sentence;

import java.io.BufferedWriter;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.regex.Matcher;

public class SentenceChunker
{
  private Matcher sentenceEndingMatcher = null;

  public static BufferedWriter sentenceBufferedWriter = null;

  private HashSet<String> abbreviations = new HashSet<String> ();

  public SentenceChunker ()
  {

  }

  /** Constructor that takes in the abbreviations directly. */
  public SentenceChunker (String[] abbreviations)
  {

    // Generate the abbreviations directly.
    for (String abbr : abbreviations) {
      this.abbreviations.add (abbr);
    }
  }

  /**
   * @param doc the document text to be analyzed
   * @return true if the document contains at least one sentence boundary
   */
  public boolean containsSentenceBoundary (String doc)
  {

    String origDoc = doc;

    /*
     * Based on getSentenceOffsetArrayList()
     */

    // String origDoc = doc;
    // int dotpos, quepos, exclpos, newlinepos;
    int boundary;
    int currentOffset = 0;

    do {

      /* Get the next tentative boundary for the sentenceString */
      setDocumentForObtainingBoundaries (doc);
      boundary = getNextCandidateBoundary ();

      if (boundary != -1) {doc.substring (0, boundary + 1);
        String remainder = doc.substring (boundary + 1);


        String candidate = /*
         * Looks at the last character of the String. If this last
         * character is part of an abbreviation (as detected by
         * REGEX) then the sentenceString is not a fullSentence and
         * "false" is returned
         */
        // while (!(isFullSentence(candidate) &&
        // doesNotBeginWithCaps(remainder))) {

          if (nextBoundary == -1) {
            break;
          }
           boundary = nextBoundary;
          candidate = doc.substring (0, boundary + 1);
          remainder = doc.substring (boundary + 1);

        }
```

```
        if (candidate.length () > 0) {
          // sentences.addElement(candidate.trim().replaceAll("\n", "
          // "));
          // sentenceArrayList.add(new Integer(currentOffset + boundary
          // + 1));
          // currentOffset += boundary + 1;

          // Found a sentence boundary. If the boundary is the last
          // character in the string, we don't consider it to be
          // contained within the string.
          int baseOffset = currentOffset + boundary + 1;
          if (baseOffset < origDoc.length ()) {
            // System.err.printf("Sentence ends at %d of %d\n",
            // baseOffset, origDoc.length());
            return true;
          }
          else {
            return false;
          }
        }
        // origDoc.substring(0,currentOffset));
        // doc = doc.substring(boundary + 1);
        doc = remainder;
      }
    }
    while (boundary != -1);

    // If we get here, didn't find any boundaries.
    return false;

  }

  public ArrayList<Integer> getSentenceOffsetArrayList (String doc)
  {
    ArrayList<Integer> sentenceArrayList = new ArrayList<Integer> ();

    // String origDoc = doc;
    // int dotpos, quepos, exclpos, newlinepos;
    int boundary;
    int currentOffset = 0;
    sentenceArrayList.add (new Integer (0));

    do {

      /* Get the next tentative boundary for the sentenceString */
      setDocumentForObtainingBoundaries (doc);
      boundary = getNextCandidateBoundary ();

      if (boundary != -1) {
        String candidate = doc.substring (0, boundary + 1);
        String remainder = doc.substring (boundary + 1);

        /*
         * Looks at the last character of the String. If this last character
         * is part of an abbreviation (as detected by REGEX) then the
         * sentenceString is not a fullSentence and "false" is returned
         */
        // while (!(isFullSentence(candidate) &&
        // doesNotBeginWithCaps(remainder))) {
        while (!(doesNotBeginWithPunctuation (remainder) &&
                 isFullSentence (candidate))) {

          /* Get the next tentative boundary for the sentenceString */
          int nextBoundary = getNextCandidateBoundary ();
          if (nextBoundary == -1) {

        }

        if (candidate.length () > 0) {
          sentenceArrayList.add (new Integer (currentOffset + boundary + 1));
          currentOffset += boundary + 1;
        }
        // origDoc.substring(0,currentOffset));
        // doc = doc.substring(boundary + 1);
```

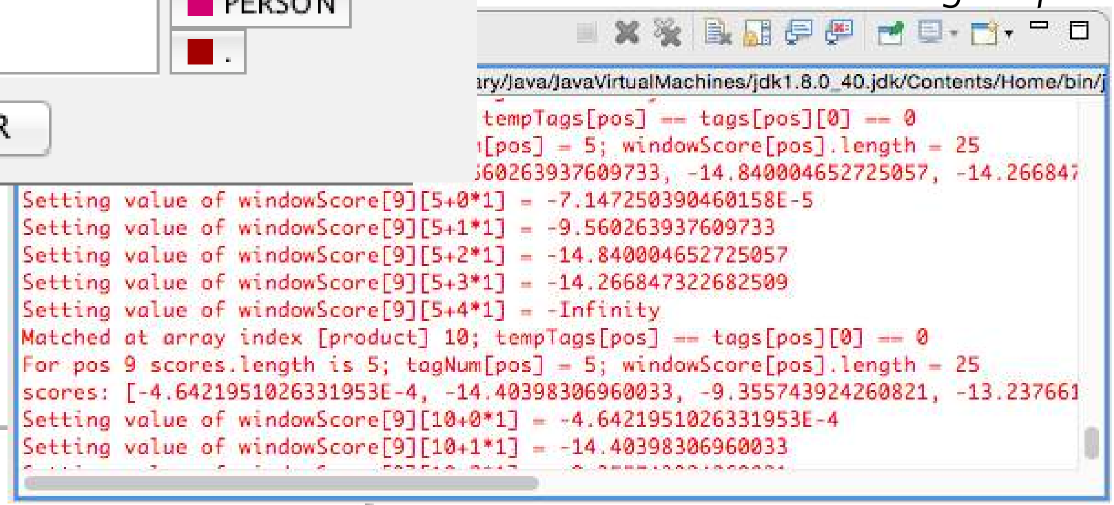
```
          doc = remainder;
        }
      }
      while (boundary != -1);

      if (doc.length () > 0) {
        sentenceArrayList.add (new Integer (currentOffset + doc.length ()));
      }

      sentenceArrayList.trimToSize ();
      return sentenceArrayList;
    }

    private void setDocumentForObtainingBoundaries (String doc)
    {
  sentenceEndingMatcher = SentenceConstants.
          sentenceEndingPattern.matcher (doc);
    }

    private int getNextCandidateBoundary ()
    {
      if (sentenceEndingMatcher.find ()) {
        return sentenceEndingMatcher.start ();
      }
      else
        return -1;

    }

    private boolean doesNotBeginWithPunctuation (String remainder)
    {
      Matcher m = SentenceConstants.punctuationPattern.matcher (remainder);
      return (!m.find ());
    }

    private String getLastWord (String cand)
    {
      Matcher lastWordMatcher = SentenceConstants.lastWordPattern.matcher (cand);
      if (lastWordMatcher.find ()) {
        return lastWordMatcher.group ();
      }
      else {
        return "";
      }
   }

    /*
     * Looks at the last character of the String. If this last character is
     * par of an abbreviation (as detected by REGEX)
     * then the sentenceString is not a fullSentence and "false" is returned
     */
    private boolean isFullSentence (String cand)
    {

      // cand = cand.replaceAll("\n", " "); cand = " " + cand;

      Matcher validSentenceBoundaryMatcher  =
          SentenceConstants.validSentenceBoundaryPattern.matcher (cand);
      if (validSentenceBoundaryMatcher.find ()) return true;

      Matcher abbrevMatcher = SentenceConstants.abbrevPattern.matcher (cand);

      if (abbrevMatcher.find ()) {
        return false; // Means it ends with an abbreviation

         e {
         / Check if the last word of the sentenceString has an entry in the
         / abbreviations dictionary (like Mr etc.)
         tring lastword = getLastWord (cand);

        if (abbreviations.contains (lastword)) { return false; }

      }

      return true;
    }
  }
```

Java Implementation of Sentence Boundary Detection

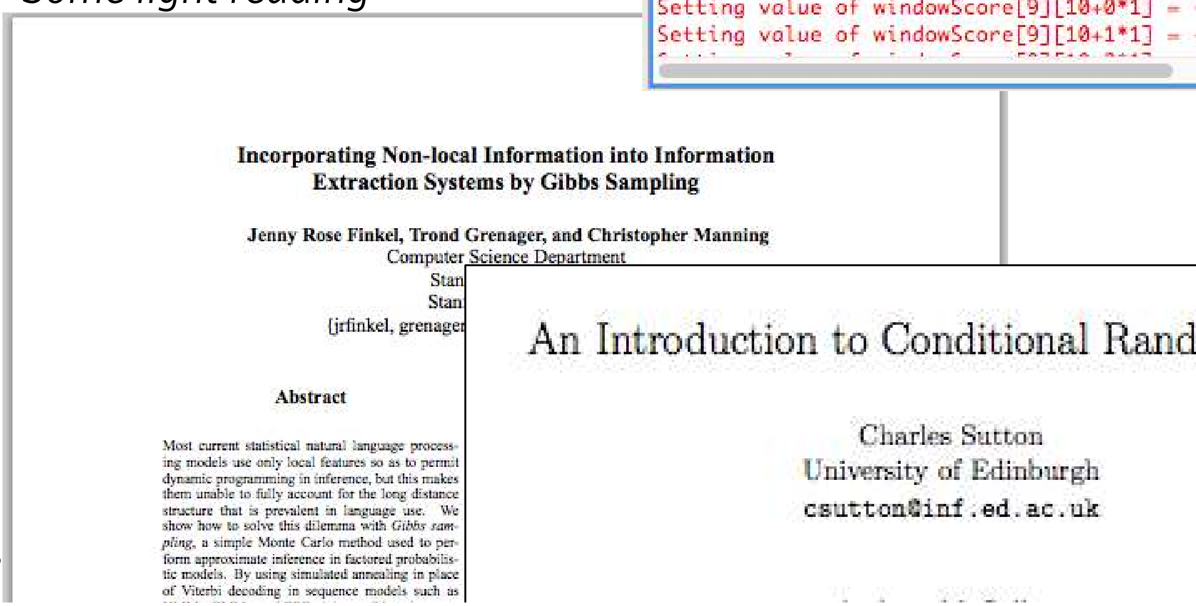
Ease of Comprehension: What not to do (2)



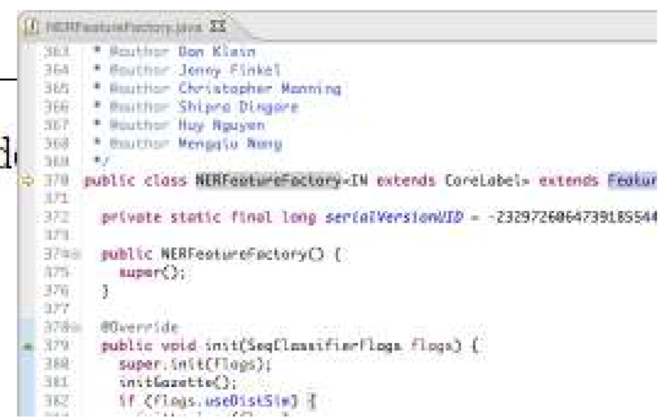
Viterbi debug output



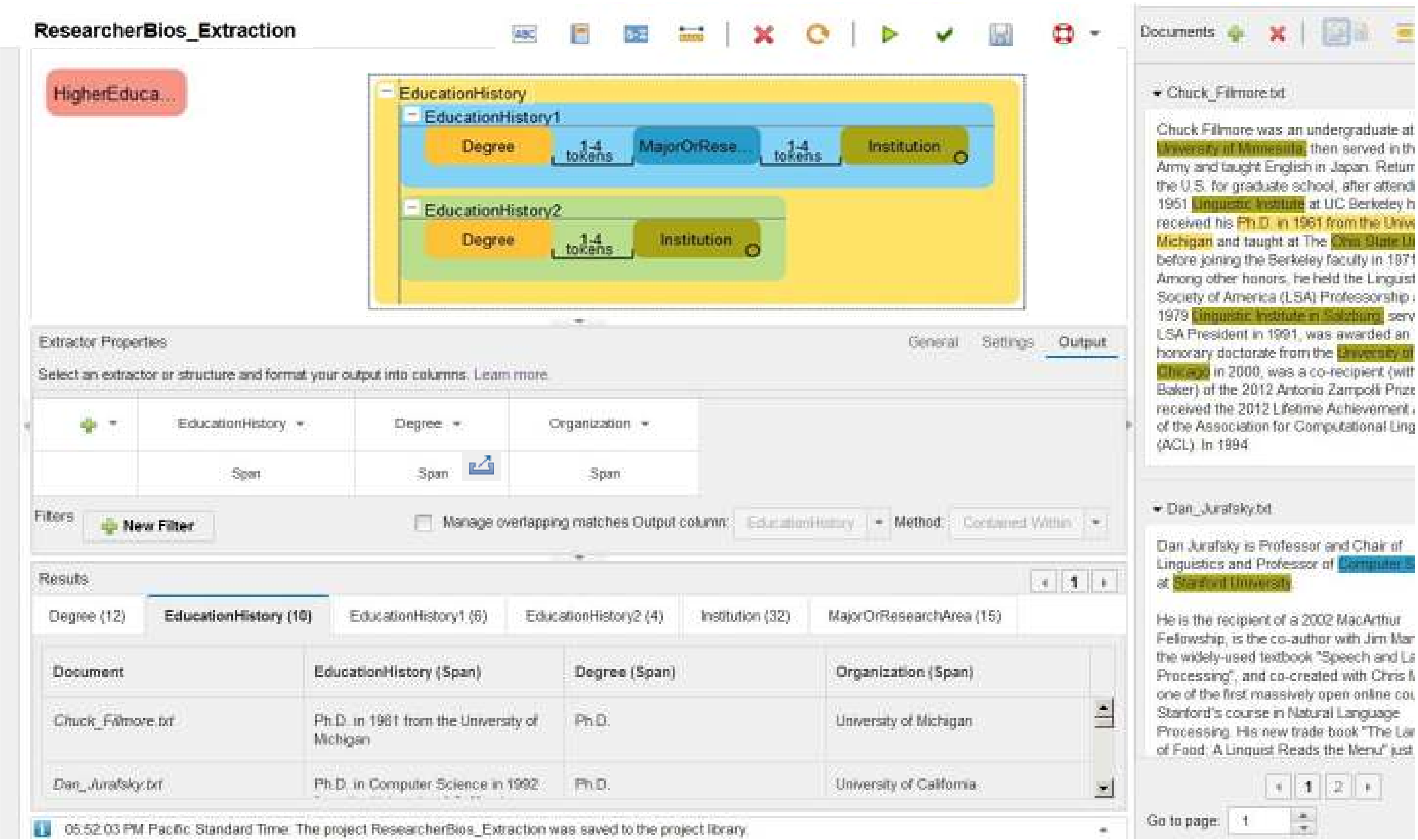
Some light reading



Feature extraction (2200 lines)



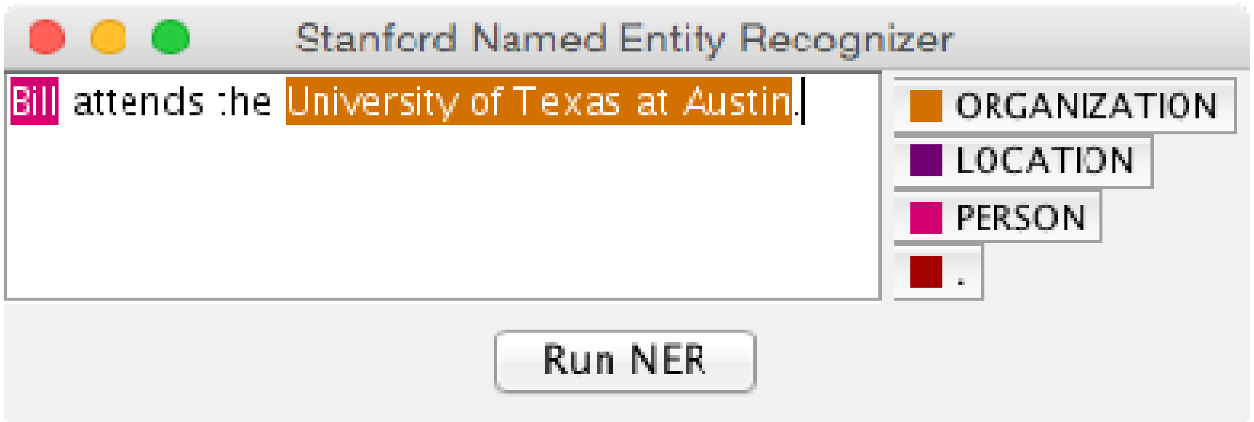
Ease of Comprehension Example



Requirements for IE in the Enterprise

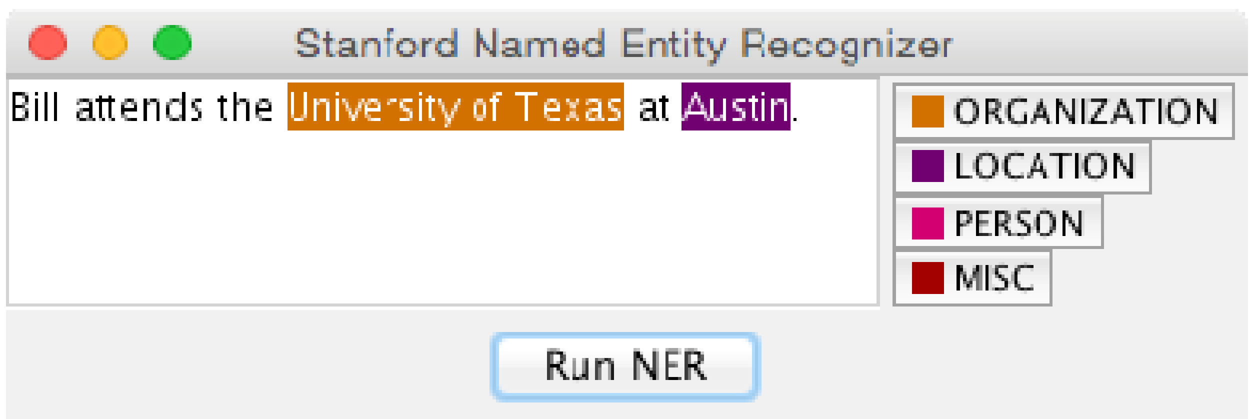
- Scalability
- Expressivity
- Ease of comprehension
- Ease of debugging

Ease of Debugging: What not to do



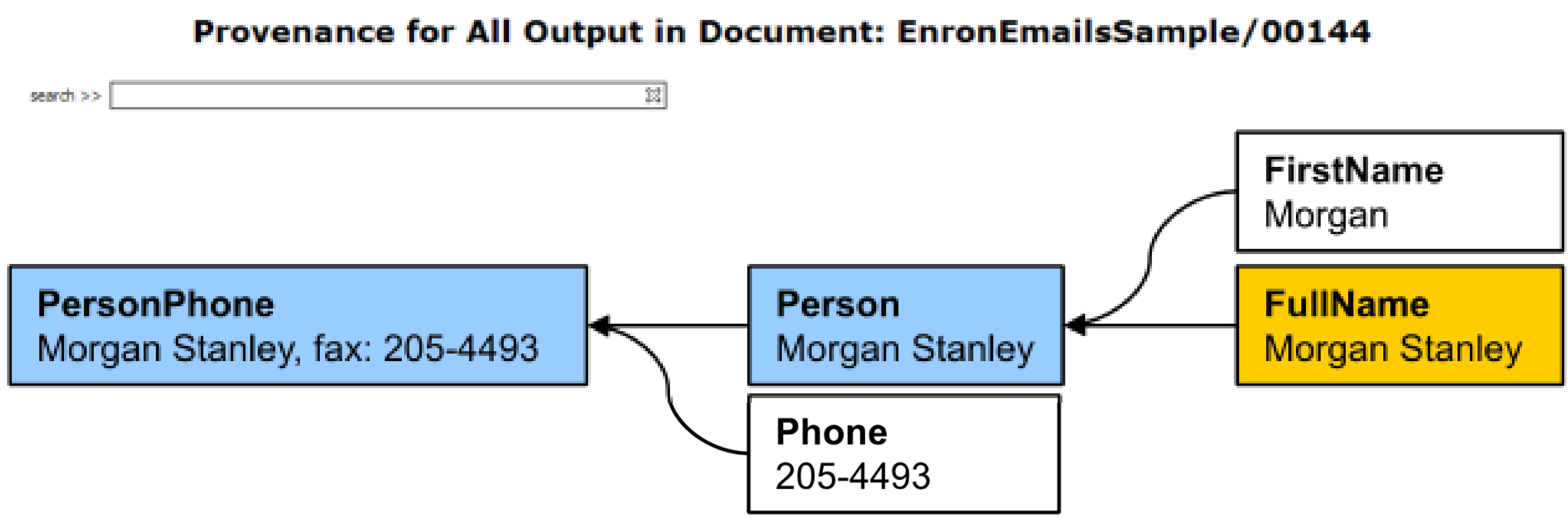
English.all.3class

Same features.
Same entities.
Slightly different training data.
Wrong answer.



English.CoNLL.4class

Ease of Debugging Example



- **Type:** **FullName**
- **Operation:** **EXTRACT DICTIONARY**
- **Annotation:**

name: *Span over Doc.text*
Doc.text[478-492]: *Morgan Stanley*

- **AQL rule:**

```
create view FullName as
extract
    dictionary ‘fullNames.dict’
    on D.text as name
from Doc D;
```

Requirements for IE in the Enterprise

- Scalability
- Expressivity
- Ease of comprehension
- Ease of debugging
- Ease of enhancement

Example: Sentiment Analysis

FHLMC reported $4.4bn net loss and requested $6bn in capital from Treasury.

Entity of interest

Good or bad?

Intel's 2013 capex is elevated at 23% of sales, above average of 16%

I'm still hearing from clients that Merrill's website is better.

I need to go back to Walmart, Toys R Us has the same toy $10 cheaper!

Customer or competitor?

Requirements for IE in the Enterprise

- Scalability
- Expressivity
- Ease of comprehension
- Ease of debugging
- Ease of enhancement

Transparency

Road map

- Focus of this tutorial:
  - Achieving transparency…
  - …while leveraging machine learning
- Parts that will follow:
  - Part 2: Intro to Transparent Machine Learning
  - Part 3: State of the Art in Transparent ML
  - Part 4: Case study
  - Part 5: Research Challenges and Future Directions

Transparent ML: Intro

A Brief History of IE

Rule-Based

- 1978-1997: MUC (Message Understanding Conference) – DARPA competition 1987 to 1997
  - FRUMP [DeJong82]
  - FASTUS [Appelt93],
  - TextPro, PROTEUS
- 1998: Common Pattern Specification Language (CPSL) standard [Appelt98]
  - Standard for subsequent rule-based systems
- 1999-2010: Commercial products, GATE
- 2006 – Declarative IE started in Universities and Industrial Labs

Machine Learning

- At first: Simple techniques like Naive Bayes
- 1990's: Learning Rules
  - AUTOSLOG [Riloff93]
  - CRYSTAL [Soderland98]
  - SRV [Freitag98]
- 2000's: More specialized models
  - Hidden Markov Models [Leek97]
  - Maximum Entropy Markov Models [McCallum00]
  - Conditional Random Fields [Lafferty01]
  - Automatic feature expansion

A False Dichotomy

Regarded as lacking in research opportunities

Lots of research focuses here

Rule-Based

Humans involved in all aspects

Opaque

Machine Learning

Humans not involved at all

	Rule-Based	Machine Learning
Model of representation	Rules	The more complex, the better
Learning algorithm	None	Completely automatic; the more complex the better
Incorporation of domain knowledge	Manual, by writing rules	The least, the better

IE system traditionally perceived as either completely Rule-based or completely ML-based.

The Reality Is Much More Nuanced !

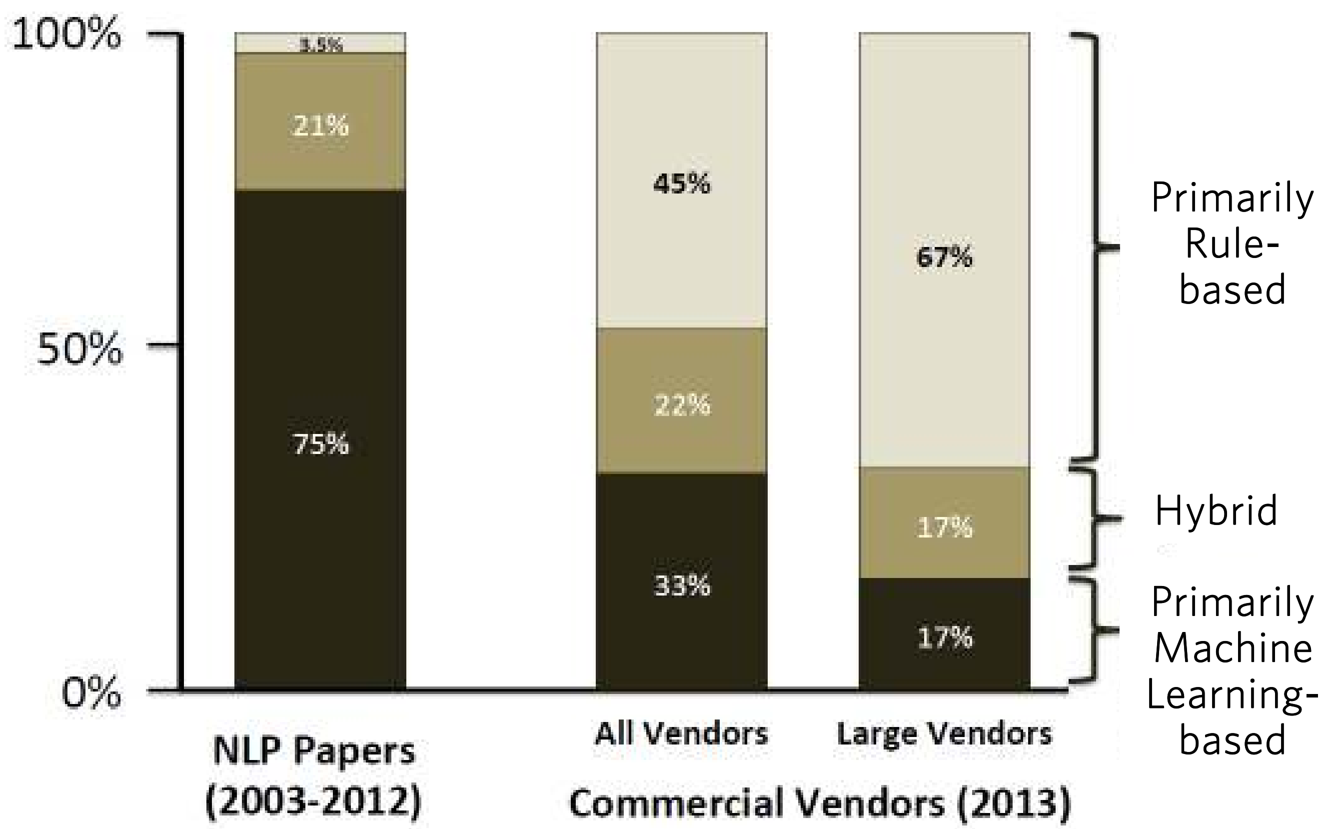
Spectrum of Techniques

Opaque

	Rule-Based *Humans involved in all aspects*	**Real Systems**	**Machine Learning** *Humans not involved at all*
Model of representation	Rules		The more complex, the better
Learning algorithm	None		Completely automatic; the more complex the better
Incorporation of domain knowledge	Manual, by writing rules		The least, the better

Real Systems: A Practical Perspective

- Entity extraction
- EMNLP, ACL, NAACL, 2003-2012
- 54 industrial vendors (Who's Who in Text Analytics, 2012)



[Chiticariu, Li, Reiss, EMNLP 2013]

Why Do Real Systems Use Rules ?

Rule-Based

PROs

- Easy to comprehend
- Easy to debug
- Easy to enhance

CONs

- Heuristic
- Requires tedious manual labor

Machine Learning

PROs

- Trainable
- Adapts automatically
- Reduces manual effort

CONs

- Requires labeled data
- Requires retraining for domain adaptation
- Requires ML expertise to use or maintain
- Opaque

Why Do Real Systems Use Rules ?

Rule-Based

PROs

- Easy to comprehend
- Easy to debug
- Easy to enhance

REQUIREMENTS in practice

Machine Learning

PROs

- Trainable
- Adapts automatically
- Reduces manual effort

NICE TO HAVE in practice

Transparent ML: meet the REQUIREMENTs, while retaining as many of the NICE TO HAVEs !

Transparent Machine Learning (Transparent ML)

- An ideal Transparent ML technique is one that:
  1. Produces models that a typical real world user can read, understand, and edit
     → Easy to comprehend, debug, and enhance
  2. Uses algorithms that a typical real world user can understand and influence
     → Easy to comprehend, debug, and enhance
  3. Allows a real world user to incorporate domain knowledge when generating the models
     → Easy to enhance

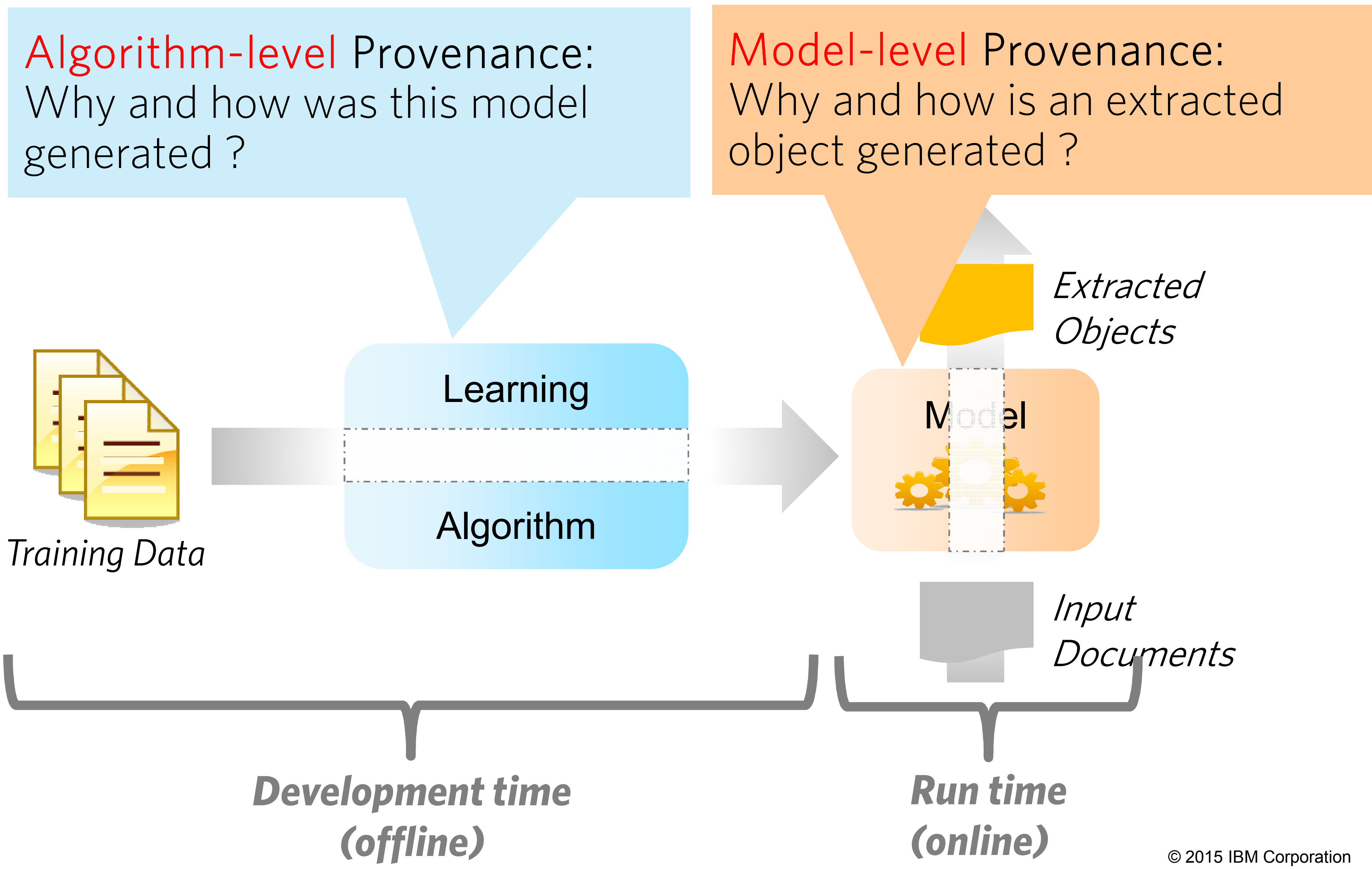
The Reality Is Much More Nuanced !

Spectrum of Techniques

Rule-Based	**Transparent ML**	**Opaque Machine Learning**
Humans involved in all aspects	Real Systems	*Humans not involved at all*

Model of representation	Rules	?	The more complex, the better
Learning algorithm	None	?	Completely automatic; the more complex the better
Incorporation of domain knowledge	Manual, by writing rules	?	The least, the better

Provenance



Algorithm-level Provenance:
Why and how was this model generated ?
Model-level Provenance:
Why and how is an extracted object generated ?
Extracted Objects
Training Data
Learning
Algorithm
Model
Input Documents
Development time (offline)
Run time (online)

Key Dimension 1: Models of Representation

Simple → Complex

Simple					Complex
• *Dictionary* • *Regular Expression*	• *Single rule (pattern)*	• *Rule Program*	• *Rules + Classifier*	• *Classification rules*	• *Decision Tree* • *SVM* • *CRF* • *HMM* • *Deep Learning* • *...*

- Simple models, e.g., dictionaries, regular expressions ...
- ... to more expressive models such as sequence patterns, dependency path patterns, rule programs ...
- ... to more complex models e.g., classifiers, or a combination of the above

Spectrum of Models of Representation (1/4): Sequence Pattern Rules

- A rule matches a linear sequence of tokens

- E.g., CPSL-style sequence rules [Appelt 1998]

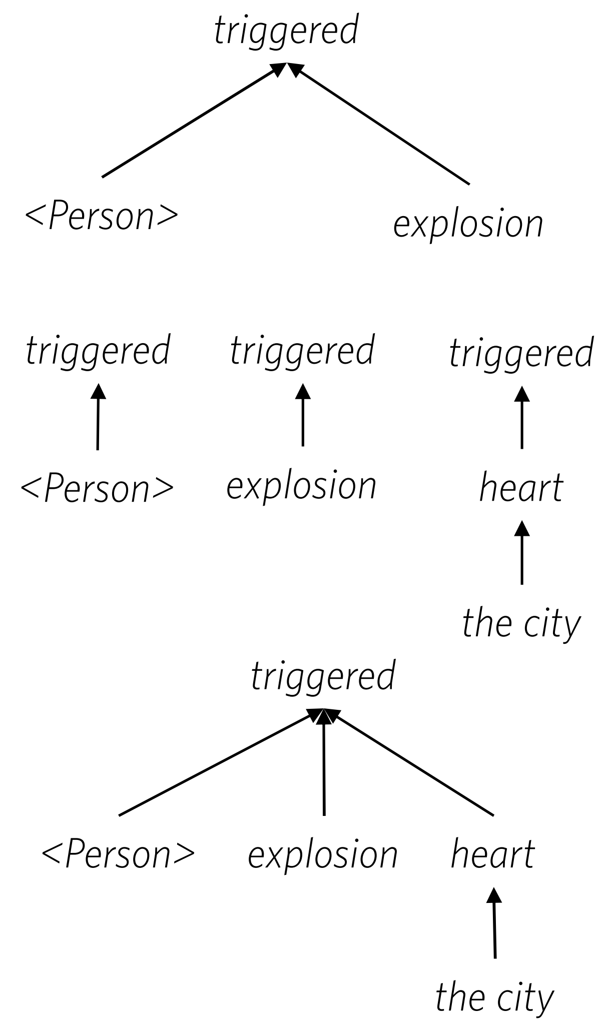
Organization Candidate

Token	**Token**	**Token**
Dictionary='Org. Prefix'	*string='of'*	*Dictionary='City Name'*

- Components include:
  - Orthographic features: e.g., matches for a regular expression
  - Lexical features: e.g., matches of a dictionary of terms
  - Syntactic features. e.g., Part of Speech (POS) tags, Noun Phrase (NP) chunks
  - Semantic features: e.g., named entity tags

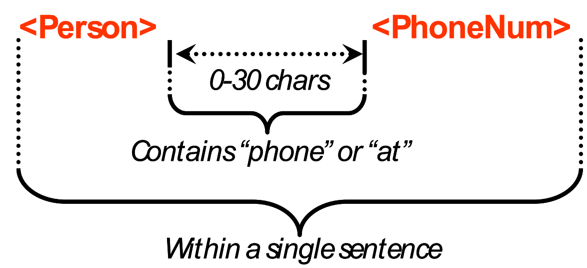
Spectrum of Models of Representation (2/4): Path Pattern Rules

- A rule matches a subgraph of a parse tree [Sudo et al., 2003]
- Predicate-argument (PA) structure
  - Based on direct relation with a predicate
- Chain Model
  - Based on a chain of modifiers of a predicate
- Subtree Model
  - Any connected subtree of a dependency parse
  - Provide reliable contexts (like PA model)
  - Captures long-distance relationship (like Chain model)



Spectrum of Models of Representation (3/4): Predicate-based Rules

- Rule program expressed using first order logic
- SQL-like [Krishnamurthy et al., ICDE 2008]




```
create view Person as ...; create view PhoneNum as ...;
create view Sentence as ...;

create view PersonPhone as
select P.name as person, N.number as phone
from Person P, PhoneNum N, Sentence S
where
   Follows(P.name, N.number, 0, 30)
   and Contains(S.sentence, P.name) and Contains(S.sentence, N.number)
   and ContainsRegex(/\b(phone|at)\b/, SpanBetween(P.name, N.number));
```

- Prolog-like [Shen et al., 2007]

```
Person(d, person) ⇐ ...; PhoneNum(d, phone) ⇐ ...; Sentence(d, person) ⇐ ...;

PersonPhone(d, person, phone) ⇐ Person(d, person), PhoneNum(d, phone), Sentence(d, sentence),
                                 before(person, phone, 0, 30),
                                 match(spanBetween(person, phone), /\b(phone|at)\b/),
                                 contains(sentence, person), contains(sentence, phone);
```

Spectrum of Models of Representation (4/4)

- Classifiers
  - Decision trees, logistic regression, Support Vector Machines (SVM), ...

- Graphical models
  - Conditional Random Fields (CRF), Hidden Markov Model (HMM), ...

Key Dimension 1: Models of Representation

Simple → Complex

- *Dictionary*
- *Regular Expression*
- *Single rule (pattern)*
- *Rule Program*
- *Rules + Classifier*
- *Classification rules*
- *Decision Tree*
- *SVM*
- *CRF*
- *HMM*
- *Deep Learning*

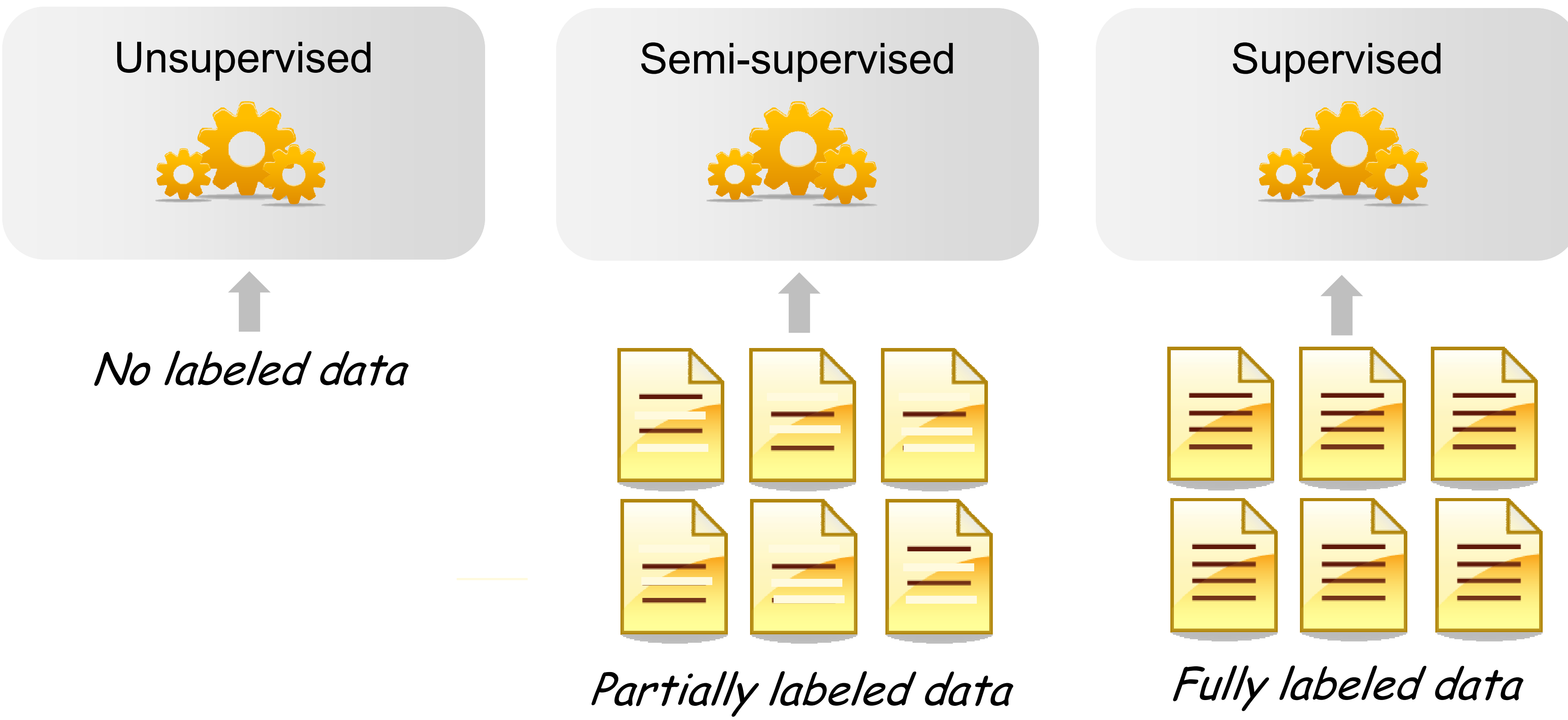
Transparent → Opaque

- Transparency: Does the model generate explainable output (i.e., extracted objects) ?
- Transparency is determined by the presence or absence of Model-level Provenance
- Model-level Provenance: ability to connect an extracted object to a subset of the input data and a part of the model that generated it

  → critical to comprehending and debugging the extracted objects
- The simpler the model, the more likely to have Model-level Provenance

  → the more transparent the model
- Range of transparency cutoff on this spectrum, depending on the application

Key Dimension 2: Learning Algorithms (1/2)



Key Dimension 2: Learning Algorithms (2/2)

Unsupervised | Semi-supervised | Supervised

- **Transparency:** Does the learning algorithm generate explainable output, i.e., model?
- **Transparency** is determined by the presence or absence of **Algorithm-level Provenance**
- **Algorithm-level Provenance:** ability to connect the model or part of the model with a subset of the input data to the learning algorithm that produces the model

  → Critical for comprehending, debugging and maintaining the model

Key Dimension 3: Incorporation of Domain Knowledge (1/3)

- Why do we need to incorporate domain knowledge ?
  - In a contest/competition environment (e.g., MUC, TAC), the model is trained on one domain and tested on the same domain
  - Hardly the case in practice: the model is deployed in an environment usually different from that where the model was trained

Customer or competitor?

I'm still hearing from clients that Merrill's website is better.

Debt reduction indicates sentiment for Country, but not Company

U.S to Reduce Debt Next Quarter After Borrowing Needs Fall.

We remain confident Computershare will generate sufficient earnings and operating cash flow to gradually reduce debt.

Key Dimension 3: Incorporation of Domain Knowledge (2/3)

- Types of domain knowledge
  - Complete labeled data
  - Seed examples (e.g. dictionary terms, patterns)
  - Type of extraction task
  - Choice of features and parameters
  - Metadata (e.g., knowledge base)

- Stages during learning when domain knowledge is incorporated
  - Offline: model is learned once and incorporates the domain knowledge all at once
  - Iterative: model is learned through a set of iterations, each iteration receiving more domain knowledge
    - Interactive: Human actively involved in each iteration to provide more domain knowledge
  - Deployment: learnt model customized for the domain/application where it is deployed

Key Dimension 3: Incorporation of Domain Knowledge (3/3)

- Transparency is determined by both:
  1. Model-level Provenance
     - Can extraction results be explained by the model?

       The more explainable the results

       → The easier to incorporate domain knowledge in the model to influence the results
     - Is the incorporation of domain knowledge to the model easy and intuitive?

       The easier and more intuitive

       → The easier it is to adapt the model to a new domain

  2. Algorithm-level Provenance
     - What changes to the model does the domain knowledge result in ?

       The more explainable the changes to the model

       → The easier to incorporate domain knowledge in the algorithm to influence the model
     - Are the parameters intuitive and do they have clear semantics ?

       The more intuitive parameters

       → The easier it is to adapt the model to a new domain

Recap

- The false dichotomy
- Transparent Machine Learning
- Provenance: Model and algorithm-level
- Ensuring provenance in
  - Model
  - Learning algorithm
  - Domain adaptation

Transparent ML:
State of the Art

Objective

- Highlight some existing techniques exhibiting Transparent ML
  - Breath over depth

- Mix of techniques: Recent or/and influential
  - Not an exhaustive list !

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

Dictionaries

- A dictionary (gazetteer) contains terms for a particular concept

- Very important for IE tasks
  - E.g. list of country names, common first names, organization suffixes
  - Highly data dependent → Crucial for domain adaptation

General Approaches for Dictionary Learning

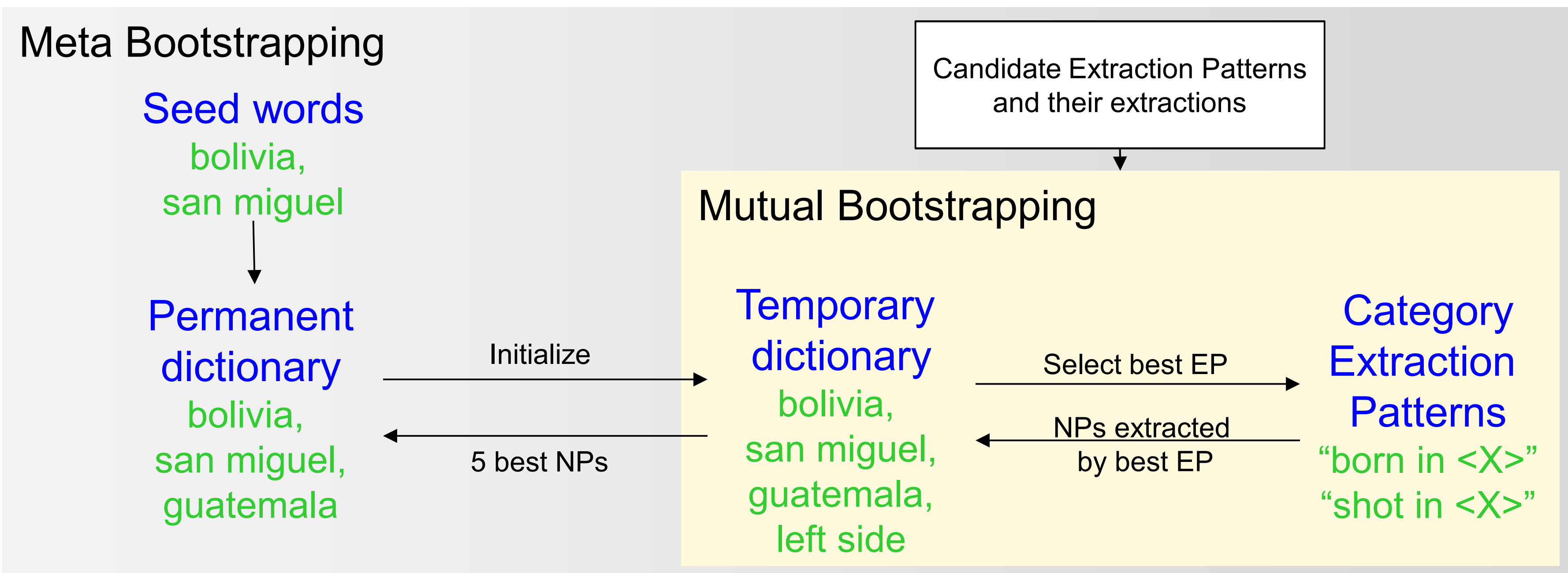
- **Dictionary Learning/Lexicon Induction:** learn a new dictionary
  - Semi-supervised (also known as Set Expansion)
    - Often used in practice because it allows for targeting specific entity classes
    - Dominant approach: Bootstrapping: e.g. [Riloff & Jones AAAI 1999]

      Seed entries → (semi-)automatically expand the list based on context
  - Unsupervised: Cluster related terms
    - Use targeted patterns or co-occurrence statistics, e.g. [Gerow 2014]

- **Dictionary Refinement:** update an existing dictionary
  - E.g., by removing ambiguous terms (e.g., [Baldwin et al., ACL 2013])
  - **Related problem:** Dictionary refinement in the context of a rule program (see later)

Dictionary Learning: Bootstrapping [Riloff & Jones AAAI 1999]

- Input: Corpus, Candidate Extraction Patterns, Seed Words
- Mutual Bootstrapping: find the Extraction Pattern (EP) that is most useful to extracting known category members; add all its extracted NPs to the dictionary
  – Scoring heuristic tries to balance pattern reliability and number of known terms extracted
- Meta Bootstrapping: guard against semantic drift due to few bad words extracted by "Best EP"
  – Scoring heuristic rewards NPs extracted by many category EPs



Dictionary Learning: Semi-supervised

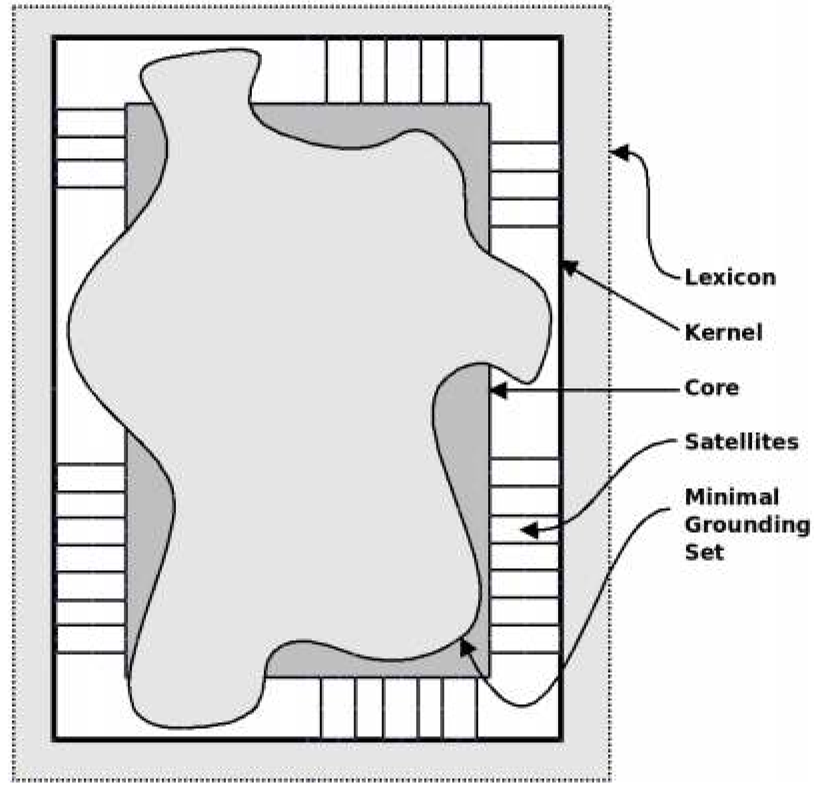
- Reducing semantic drift
  - Multi-category bootstrapping, e.g., BASILISK [Thellen & Riloff EMNLP 2002]
  - Distributional similarity to detect terms that could lead to semantic drift, e.g., [McIntosh & Curran, ACL 2009]
  - Discover negative categories, e.g., [McIntosh EMNLP 2010]
  - Hybrid: bootstrapping + semantic tagger + coreference, e.g., [Qadir & Riloff, *SEM 2012]
  - Incorporate user interaction: [Coden et al., Sem. Web Eval. Challenge 2014]

- Exploit the Web, e.g., [Downey et al., IJCAI 2007]

- Multi-word expressions, e.g., [Qadir et al. AAAI 2015]

Dictionary Learning: Unsupervised [Gerow, ACL 2014]

- **Input**: a corpus
- **Goal**: extract qualifiable sets of specialist terms found in the corpus
- Algorithm
  - Construct co-occurrence graph of all words in the corpus
    - Two words are connected if they are observed in a n-word window
  - Identify communities in the graph using a community detection algorithm
  - Rank words by their centrality in the community
- Minimal preprocessing
  - No document structure
  - No semantic relationship
  - No threshold



Communities from NIPS Proceedings

model	1.00	university	1.00	nuclear	1.00
learning	0.99	science	0.85	weapons	0.66
data	0.96	computer	0.83	race	0.57
neural	0.94	department	0.74	countries	0.40
using	0.85	engineering	0.30	rights	0.37
network	0.85	report	0.30	india	0.27
training	0.73	technical	0.29	russia	0.26
algorithm	0.66	institute	0.26	philippines	0.26
function	0.63	abstract	0.25	brazil	0.25
networks	0.62	california	0.23	waste	0.22

Term Ambiguity Detection (TAD) [Baldwin et al, ACL 2013]

- Movie night watching **brave** with Cammie n Isla n loads munchies
- This **brave** girl deserves endless retweets!
- Watching **brave** with the kiddos!
- watching Bregor playing Civ 5: **Brave** New World and thinking of getting it

- Perform term disambiguation at the term, not instance level
  - Given term T and its category C, do *all* the mentions of the term reference a member of that category?
- Motivation for IE
  - Simpler model if the term unambiguous
  - More complex model otherwise

Term	Category
Brave	**Movie**
Skyfall 007	**Movie**
A New Beginning	**Video Game**
EOS 5D	**Camera**

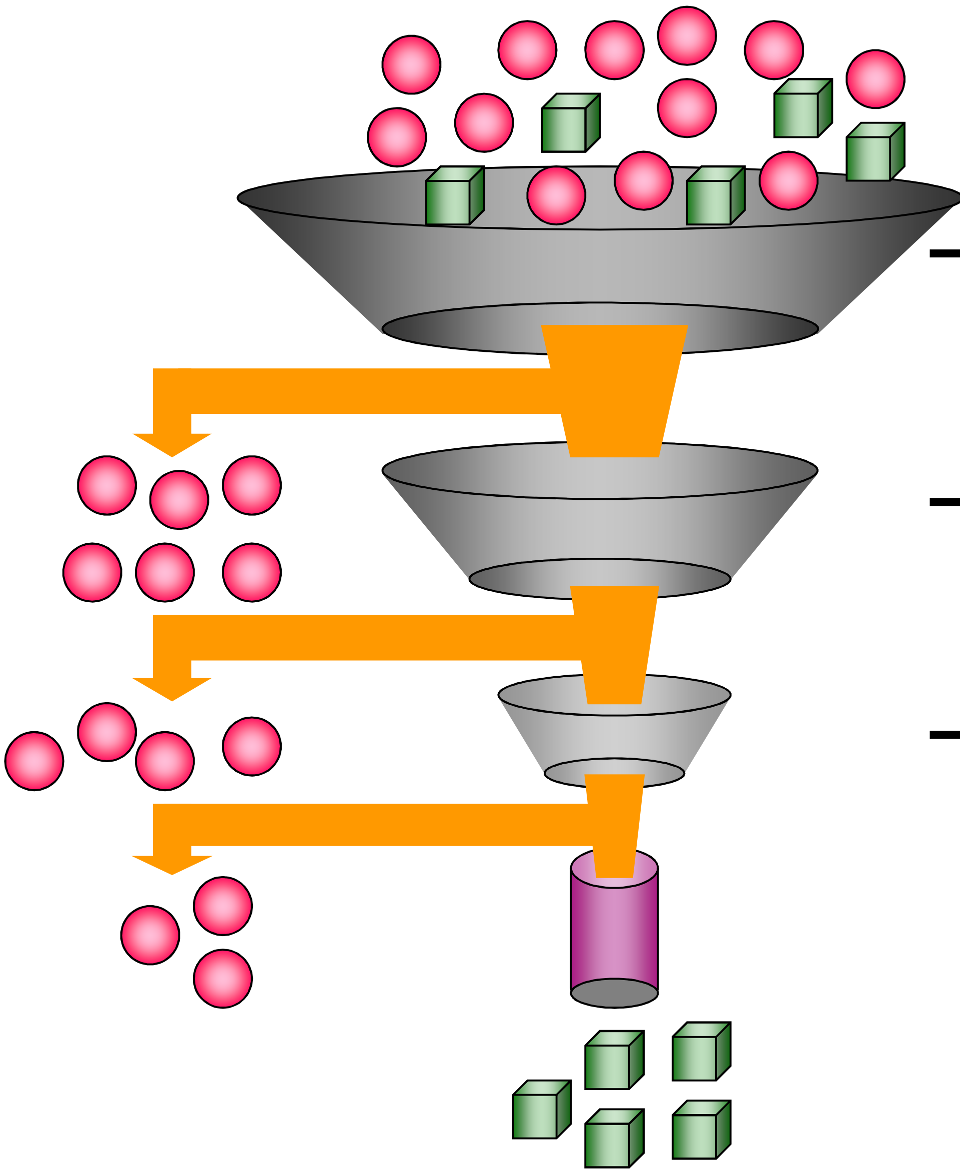
Term	Category
Brave	**Movie**
A New Beginning	**Video Game**

Ambiguous

Term	Category
Skyfall 007	**Movie**
EOS 5D	**Camera**

Unambiguous

Term Ambiguity Detection (TAD) [Baldwin et al, ACL 2013]



Step 1: N-gram
Does the term share a name with a common word/phrase?

Step 2: Ontology
Wiktionary + Wikipedia

Step 3: Clustering
Cluster the contexts in which the term appears

Transparent ML in Dictionary Learning/Refinement

- Transparency in Model of Representation
  - Very simple
  - Model-level Provenance: trivial to connect an extracted object with the input text and the part of the model that determined it

- Transparency in Learning Algorithm
  - Bootstrapping [Riloff & Jones, AAAI 1999] → Algorithm-level Provenance
    - Every change in the model can be justified by the extraction pattern that extracts it
    - In turn, the extraction pattern can be explained by the seed terms matching the pattern
  - TAD [Baldwin et al., ACL 2013] → Some transparency
    - Coarse granularity of transparency in terms of each level of filtering
    - Finer granularity of transparency within some of the filters, e.g., based on Wikipedia/Wiktionary
  - [Gerow 2014] → No transparency

- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline, for majority of techniques
  - But, easy to incorporate DK at deployment (by further modifying the dictionary)
  - Interactive techniques potentially fruitful to explore in semi-supervised settings

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

Regular Expressions (Regex)

- Regexes are essential to many IE tasks
  - Email addresses
  - Software names
  - Credit card numbers
  - Social security numbers
  - Gene and Protein names
  - ....

Web collections

Email compliance

Bioinformatics

- But writing regexes for IE is not straightforward !

- **Example**: Simple regex for **phone number** extraction:

blocks of digits separated by non-word character:

R_0 = (\d+\W)+\d+

Identifies valid phone numbers (e.g. *800-865-1125, 725-1234*)

Produces invalid matches (e.g. *123-45-6789, 10/19/2002, 1.25* ...)

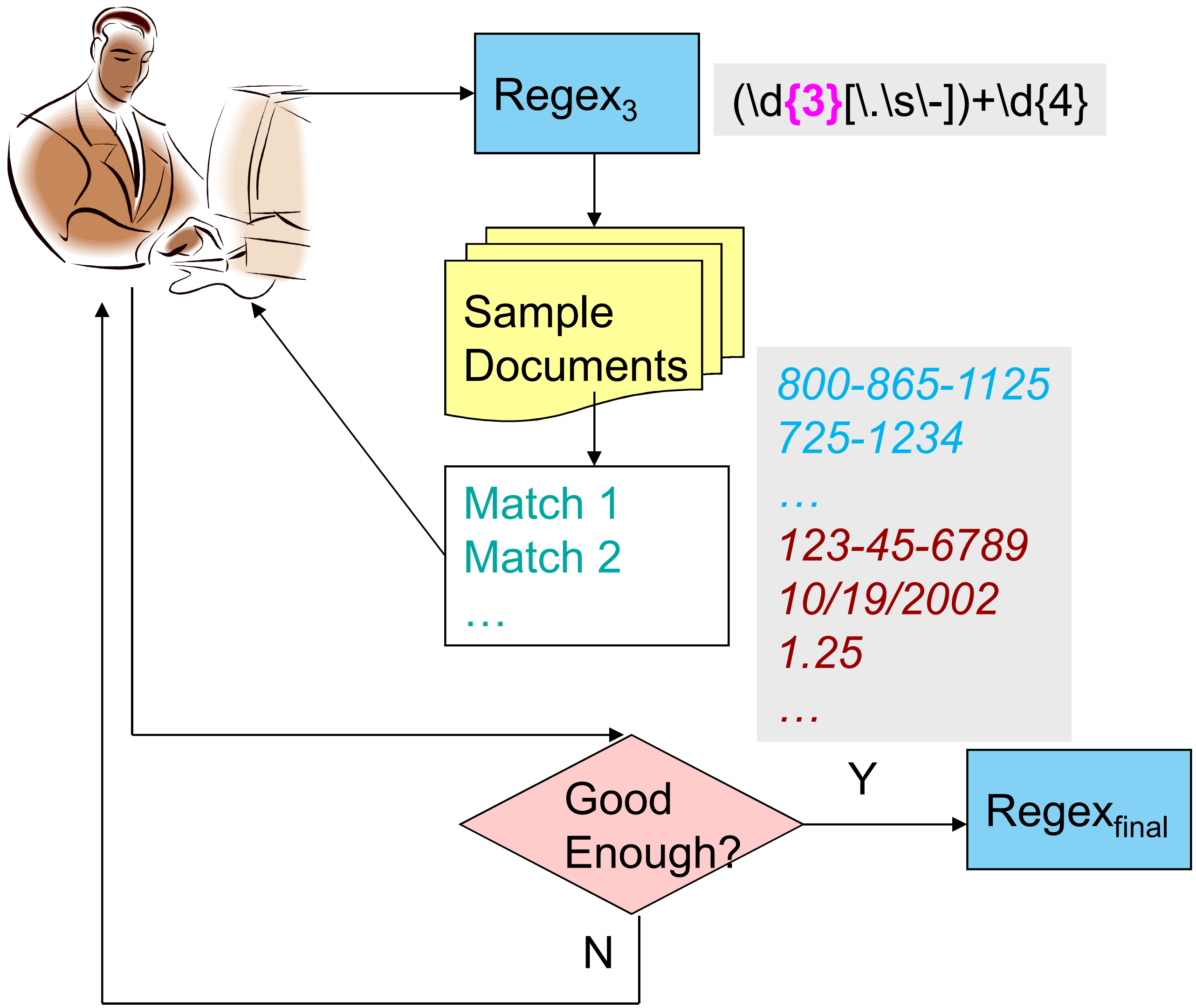
Misses valid phone numbers (e.g. *(800) 865-CARE*)

Learning Regular Expressions

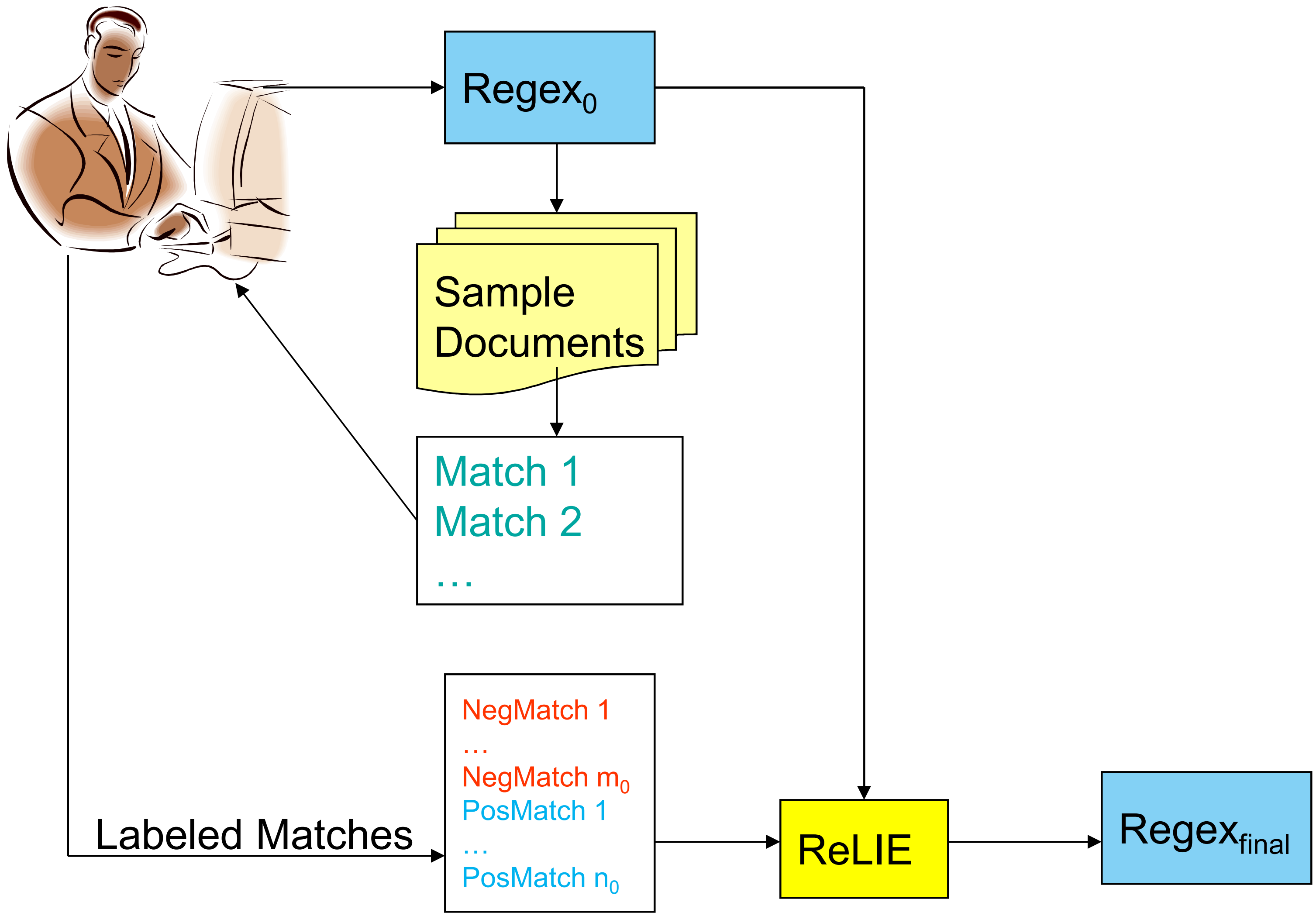
- Supervised
  - Refine regex given positive and negative examples [Li et al., EMNLP 2008]

- Semi-supervised
  - Learning regex from positive examples [Brauer et al., CIKM 2011]

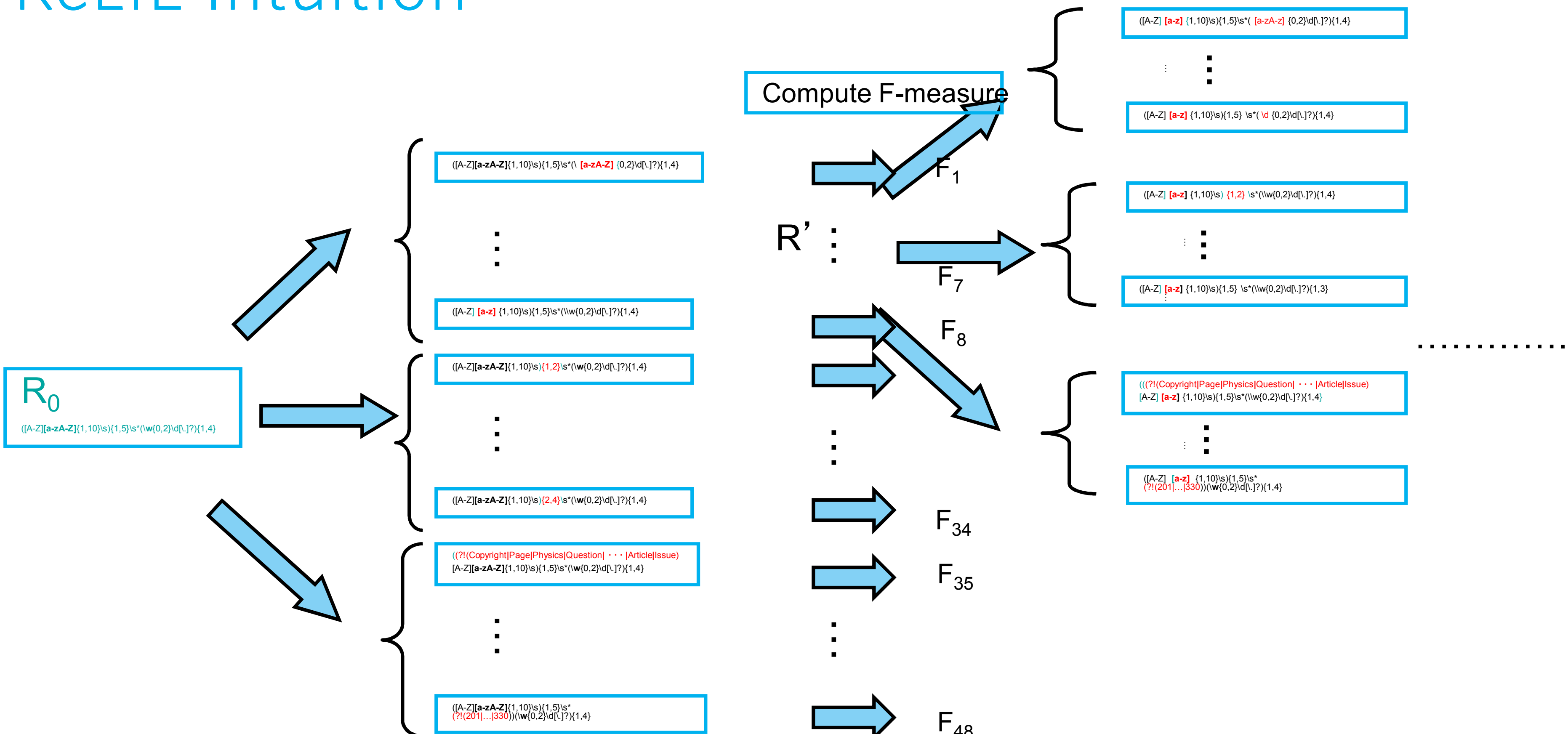
Conventional Regex Writing Process for IE



Learning $Regex_{final}$ automatically in ReLIE [Li et al., EMNLP 2008]



ReLIE Intuition



- Generate candidate regular expressions by modifying current regular expression
- Select the "best candidate" R'
- If R' is better than current regular expression, repeat the process
- Use a validation set to avoid overfitting

Regex Learning Problem

– Find the best R_f among all possible regexes
  – Best = Highest F-measure over a document collection D
  – Can only compute F-measure based on the labeled data → Limit R_f such that any match of R_f is also a match of R_0

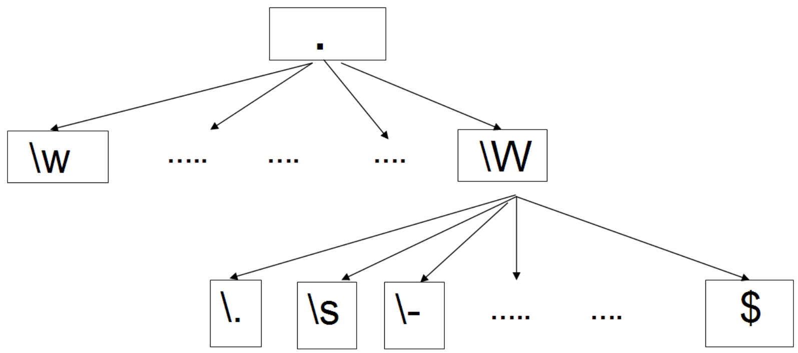
- Two Regex Transformations
  – Drop-disjunct Transformation:

$$R = R_a(R_1 | R_2 | \ldots R_i | R_{i+1} | \ldots | R_n) R_b \rightarrow R' = R_a (R_1 | \ldots R_i | \ldots) R_b$$

  – Include-Intersect Transformation

$$R = R_a X R_b \rightarrow R' = R_a (X \cap Y) R_b, \text{ where } Y \neq \emptyset$$

Applying Drop-Disjunct Transformation



- Character Class Restriction

  E.g. To restrict the matching of non-word character

  (\d+\W)+\d+ → (\d+[\.\s\-])+\d+

- Quantifier Restriction

  E.g. To restrict the number of digits in a block

  (\d+\W)+\d+ → (\d{3}\W)+\d+

Applying Include-Intersect Transformation

- Negative Dictionaries
  - Disallow certain words from matching specific portions of the regex

E.g. a simple pattern for software name extraction:

blocks of capitalized words followed by version number:

$$R_0 = ([A\text{-}Z]\backslash w^*\backslash s^*)+[Vv]?(\backslash d+\backslash .?)+$$

- Identifies valid software name (e.g. *Eclipse 3.2, Windows 2000*)
- Produces invalid matches (e.g. *ENGLISH 123, Room 301, Chapter 1.2*)

$$R_f = \textbf{(?! ENGLISH|Room|Chapter)}\ ([A\text{-}Z]\backslash w^*\backslash s^*)+[Vv]?(\backslash d+\backslash .?)+$$

Learning regex from positive examples [Brauer et al. 2011]

- **Input**: set of examples
- **Output**: one regex

Notebook models

z800
z800 AAB
d700 ASE
z40y
d50t ATX

(d|z)([0-9]0{2}|[0-9]0[a-z]) ([A-Z]+)?

Learning a Regex from Positive Examples [Brauer et al. CIKM 2011]

Step 1: Build automata to capture all features of the examples

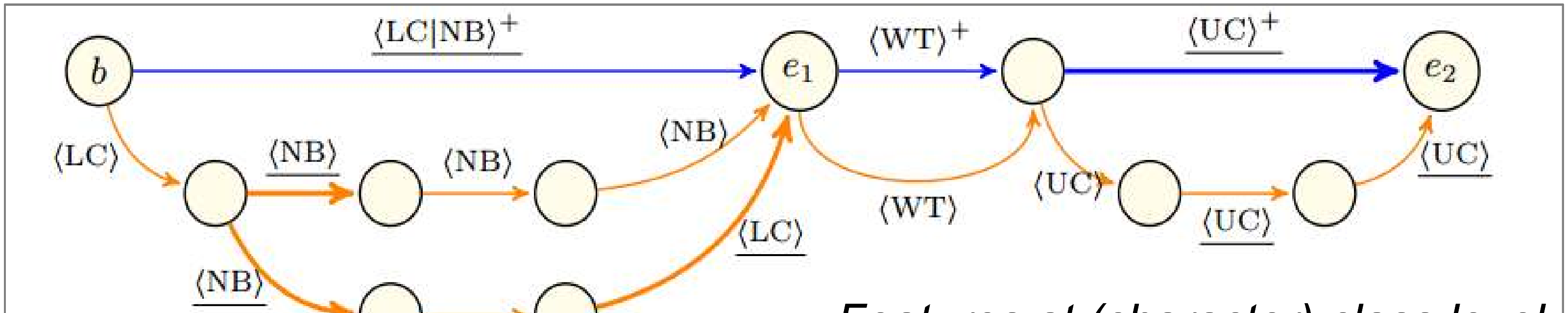
- Features: class vs. instance level and token vs. character level
- Transitions encode the sequential ordering of features in the examples

Instances

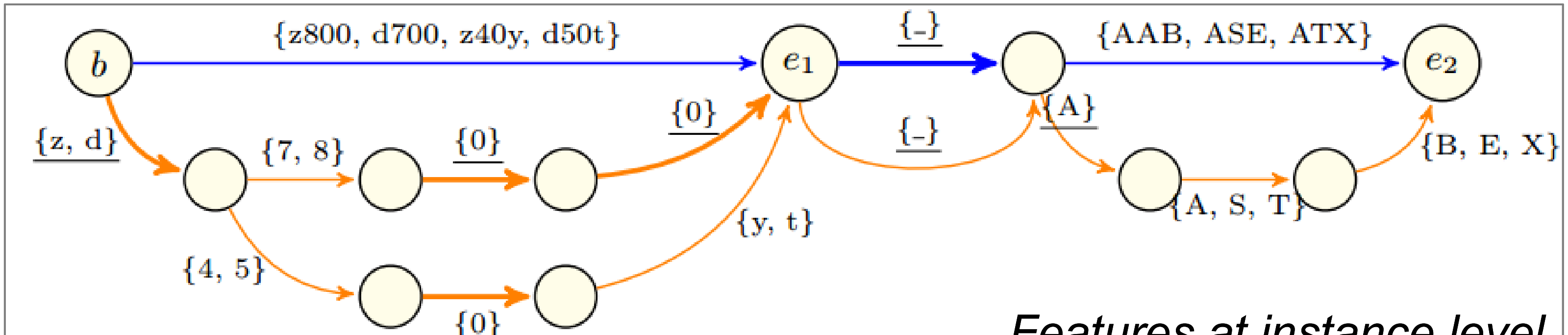
z800
z800 AAB
d700 ASE
z40y
d50t ATX

Token features

Character features



Features at (character) class level



Features at instance level

Learning a Regex from Positive Examples [Brauer et al. CIKM 2011]

Step 2: Choose among class vs. instance feature

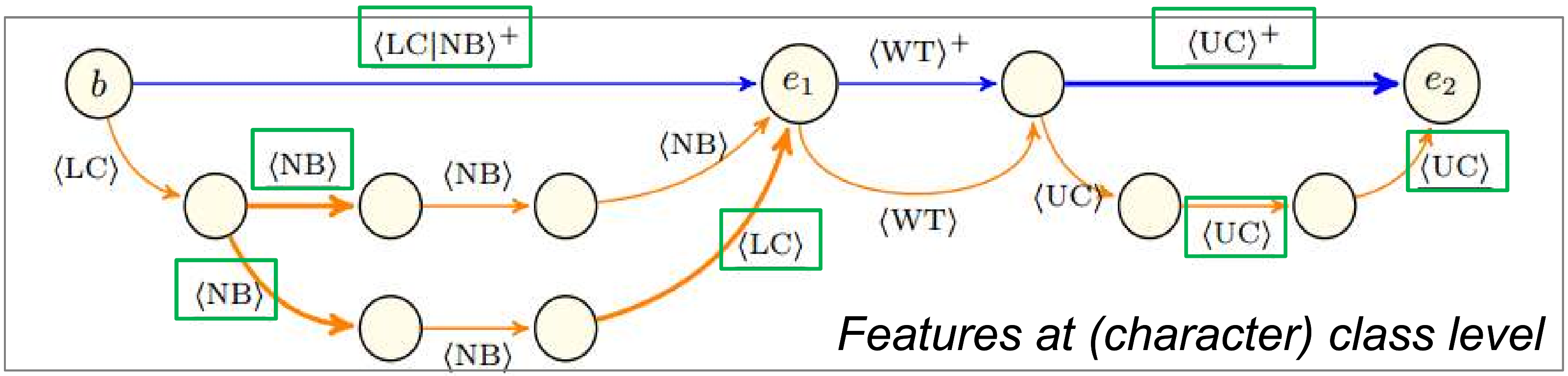
- Prefer instance feature if very common in the examples
- Parameter β to further influence the feature selection towards class features (for higher recall) vs. instance (for higher precision)

Instances

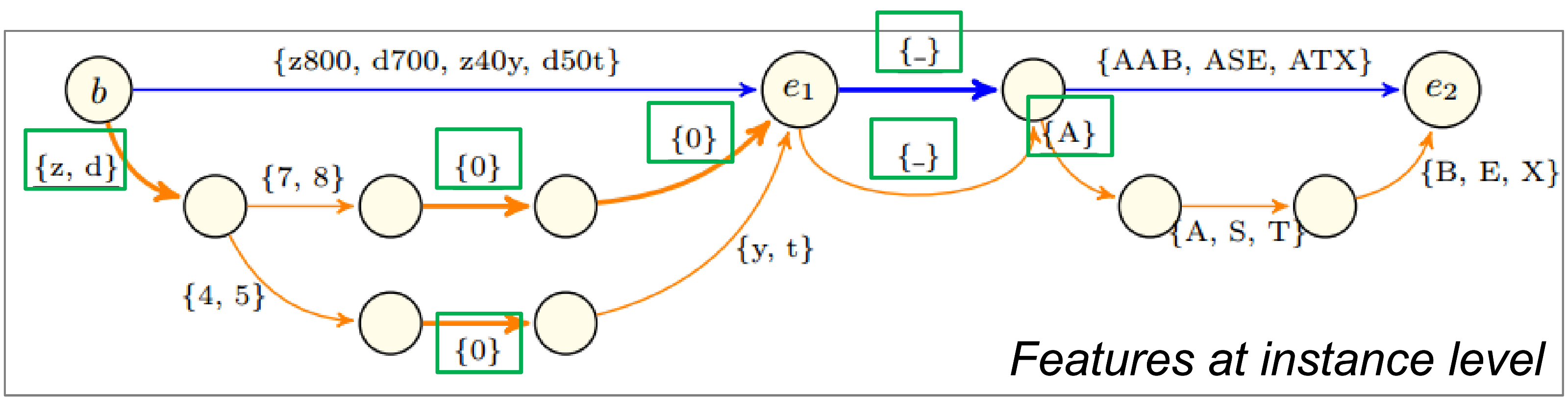
z800
z800 AAB
d700 ASE
z40y
d50t ATX

Token features

Character features



Features at (character) class level



Features at instance level

Learning a Regex from Positive Examples [Brauer et al. CIKM 2011]

Step 3: Choose among token vs. character feature

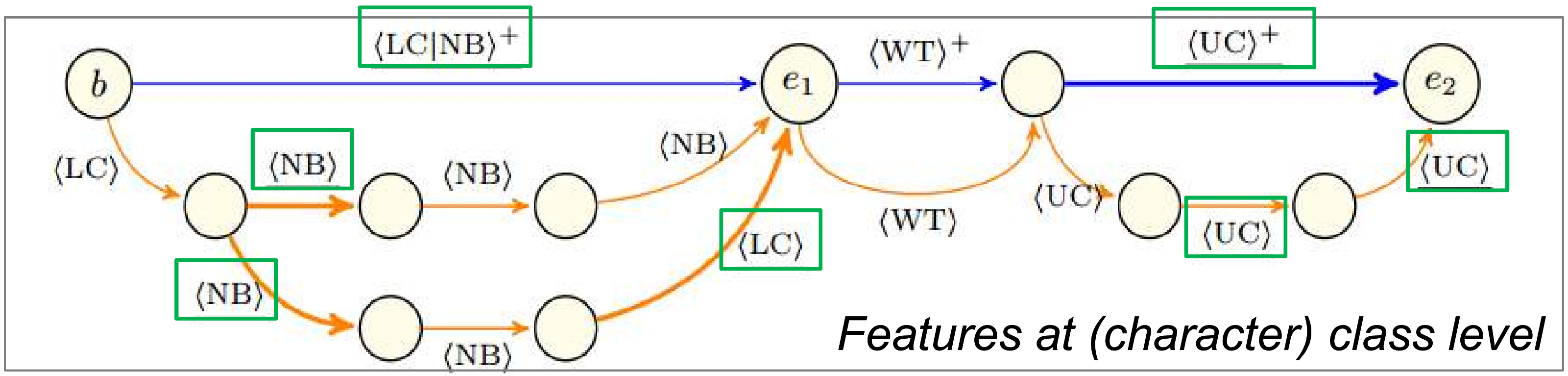
- Use the Minimum Description Length (MDL) principle to choose most promising abstraction layer
- To balance model complexity with its fitness to encode the data

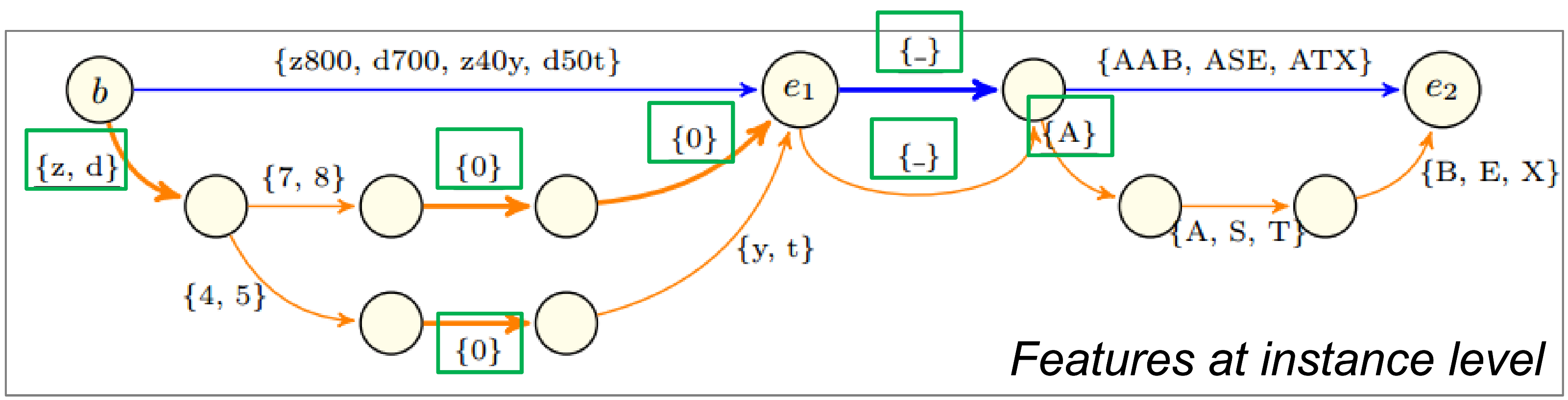
Instances

z800
z800 AAB
d700 ASE
z40y
d50t ATX

Token features

Character features





Learning a Regex from Positive Examples [Brauer et al. CIKM 2011]

Step 4: Generate regular expressions for each end state

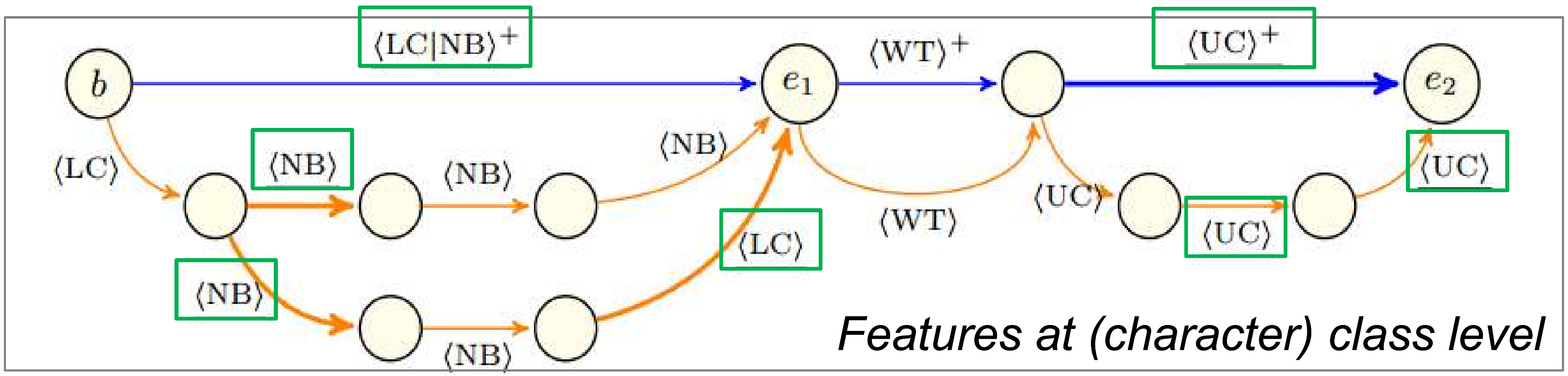
- Pick the expression with smallest MDL from begin to end state
- Apply some simplification rules, e.g. cardinality
- Final regex: (z|d) ((<NB>0{2}) | (<NB>0<LC>)) (_<UC>+){0,1}

Instances

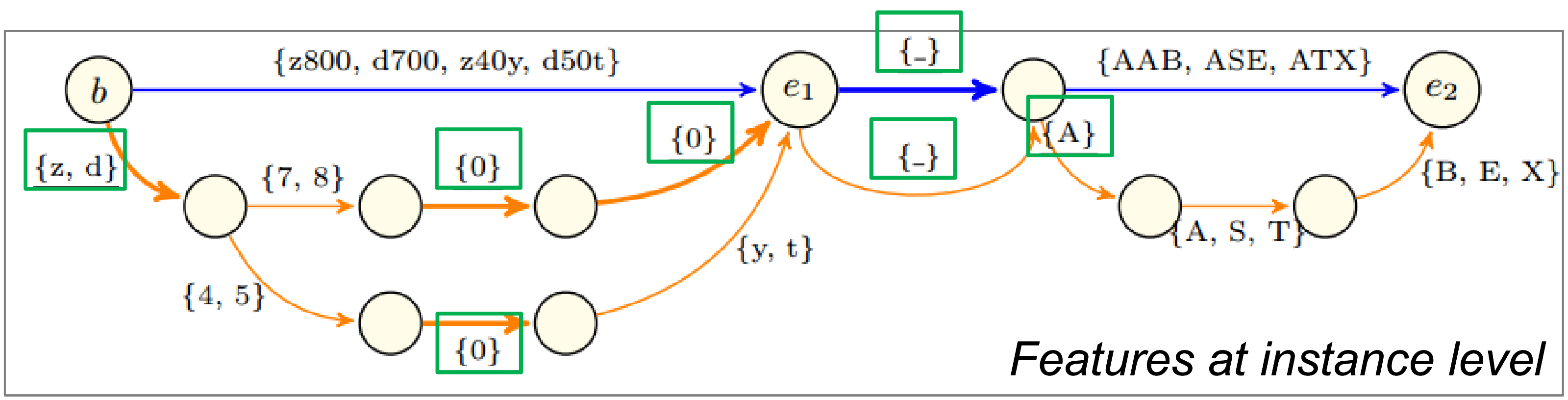
z800
z800 AAB
d700 ASE
z40y
d50t ATX

Token features

Character features



Features at (character) class level



Features at instance level

Transparent ML in Regex Learning/Refinement

- Transparency in Model of Representation
  - Simple
  - Model-level provenance: easy to connect a result of the model with the input text that determined it
- Transparency in Learning Algorithm
  - No algorithm-level provenance
  - RELIE [Li et al., EMNLP 2008] → some transparency in terms of influencing the model via the initial regular expression
  - [Brauer et al., CIKM 2011] → some transparency in influencing feature selection
- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline
  - But, easy to incorporate DK at deployment (by modifying the regex)
  - Interactive techniques potentially useful

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

Fact Extraction

Fact (or concept): can be an entity, relation, event, …

Several papers, and two tutorials in this EMNLP:
- Knowledge Acquisition for Web Search (now)
- Learning Semantic Relations from Text (Friday morning)

	Traditional IE	Open IE [Banko et el., 2007]
Input	Corpus (+ labeled data)	Corpus
Type	Specified in advance	Discovered automatically, or specified via ontology
Extractor	Type-specific	Type-independent

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

Fact Extraction: Supervised

- **Fact (or concept):** can be an entity, relation, event, …
- **Context:** Traditional IE
- **Input:** Document collection, labeled with the target concept
- **Goal:** induce rules that capture the target concept

- **Earlier work:** Sequence patterns (CPSL-style) as target language

- **Recent work:** Predicate-based rule program as target language

Fact Extraction: Supervised

- **Fact (or concept):** can be an entity, relation, event, …
- **Context:** Traditional IE
- **Input:** Document collection, labeled with the target concept
- **Goal:** induce rules that capture the target concept

- **Earlier work:** Sequence patterns (CPSL-style) as target language

- **Recent work:** Predicate-based rule program as target language

Supervised Learning of Sequence Patterns

- Input:
  - Collection of text documents, labeled with target concept
  - Available basic features: tokens, orthography, parts of speech, dictionaries, entities, …

- Goal: Define the smallest set of rules that cover the maximum number of training cases with high precision

- Model of Representation: unordered disjunction of sequence pattern rules

- General framework: greedy hill climbing strategy to learn one rule at a time
  1. S is the set of rules, initially empty
  2. While there exists a training concept not covered by any rule in S
     - Generate new rules around it
     - Add new rules to S
  3. Post process rules to prune away redundant rules

- Techniques: Bottom-up and top-down

- Surveys: [Muslea, AAAI Workshop on ML in IE 1999]

  [Sarawagi, Foundations and Trends in Databases, 2008]

Bottom-up Techniques: Generalize a Specific Rule

- Start with a specific rule covering a single instance (100% precision)
- Generalize the rule to increase its coverage, with a possible loss of precision
  - Many strategies: e.g., dropping a token, or replacing a token by a more general feature
- Remove instances covered by the rule from the training set
- Example systems: RAPIER [Califf & Mooney AAAI 1999, JML 2003], $(LP)^2$ [Ciravegna IJCAI 2001]

Bottom-up Technique Example: $(LP)^2$ [Ciravegna IJCAI 2001]

- Example text: *I am studying at University of Chicago.*

- Initial rule: snippet of *w* tokens to the left and right of the labeled instance

  <Token>[string="studying"] <Token>[string="at"]
  (<Token>[string="University"] <Token>[string="of"] <Token>[string="Chicago"]):ORG

- Some generalizations of the initial rule:
  - Two tokens generalized to orthography type

    <Token>[string="studying"] <Token>[string="at"]
    (<Token>[orth="CapsWord"] <Token>[string="of"] <Token>[orth="CapsWord"]):ORG

  - Two tokens are dropped, two tokens generalized by whether they appear in dictionaries

    (<Token>[Lookup="OrgPrefix"] <Token>[string="of"] <Token>[Lookup="CityName"]):ORG

- Exponential number of generalizations → heuristics to reduce the search space
  - Greedily select the best single step of generalization
  - User-specified maximum number of generalizations retained

- Top-k "best" generalizations are added to the "best rules pool"
  - Based on a combination of measures of quality of rules, including precision, overall coverage, and coverage of instances not covered by other rules

Top-down Techniques: Specialize a Generic Rule

- Start with a generic rule covering all instances (100% coverage)
- Specialize the rule in various ways to get a set of rules with high precision (inductive logic – style)
- Example systems: WHISK [Soderland, ML 1999], [Aitken, ECAI 2002]

Top-down Technique Example: WHISK [Soderland, ML 1999]

- Seed labeled instance: *Capitol Hill – 1 br townhome, all inclusive $675*

- Initial rule: * (*) * (*) * (*)

- Some specializations of the initial rule:
  - First slot anchored inside: * (Neighborhood) * (*) * (*)
  - First slot anchored outside: @start (*) '-' * (*) * (*)

- Greedily select the best single step of generalization
  - Capture the seed and minimize error on training set
  - Heuristics to prefer the least restrictive rule that fits the data, e.g., choose semantic class and syntactic tags over literals

- Semi-supervised and interactive
  - Start with a random sample of unlabeled instances, possibly satisfying some keywords
  - In each iteration, automatically select instances from 3 sets for the user to label
    - Covered by an existing rule → increase support for the rule or provide counter example
    - "Near" misses of existing rules
    - Not covered by any rule

Transparent ML in Learning of CPSL-style Patterns

- Transparency in Model of Representation
  - Relatively simple representation
  - Model-level Provenance: easy to connect an extracted object with the input text and a part of the model (i.e., a rule) that determined it

- Transparency in Learning Algorithm
  - No transparency

- Transparency in Incorporation of Domain Knowledge (DK)
  - Most systems → offline (fully supervised)
  - WHISK → interactive
    - Active learning techniques used to select examples for the user to label
  - Easy to incorporate domain knowledge at deployment (by further modifying the rules)

Fact Extraction: Supervised

- Earlier work: Sequence patterns (CPSL-style) as target language
- Recent work: Predicate-based rule program as target language

Supervised Learning of Predicate-based Rules

- Rule Induction: generate a rule program from basic features

- Rule refinement: refine an existing rule program

Supervised Learning of Predicate-based Rules

- Rule Induction: generate a rule program from basic features
  - E.g., [Nagesh et al., 2012]

- **Rule refinement:** refine an existing rule program

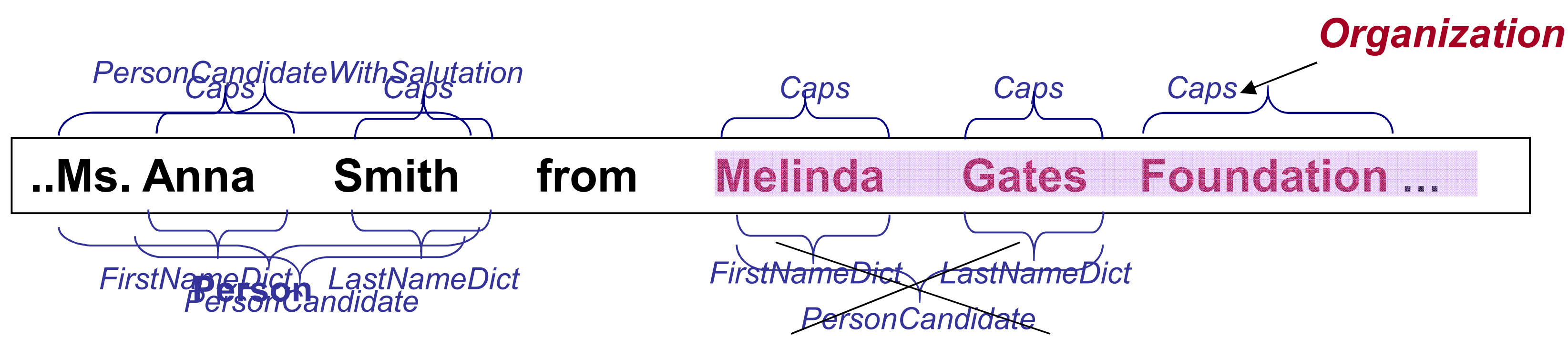
NER Rule Induction [Nagesh et al., EMNLP 2012]

- Input:
  - Basic features (dictionaries & regular expressions)
  - Fully labeled document collection (PER, ORG, LOC)
- Goal: Induce an initial set of named-entity rules that can be refined / customized by domain-expert

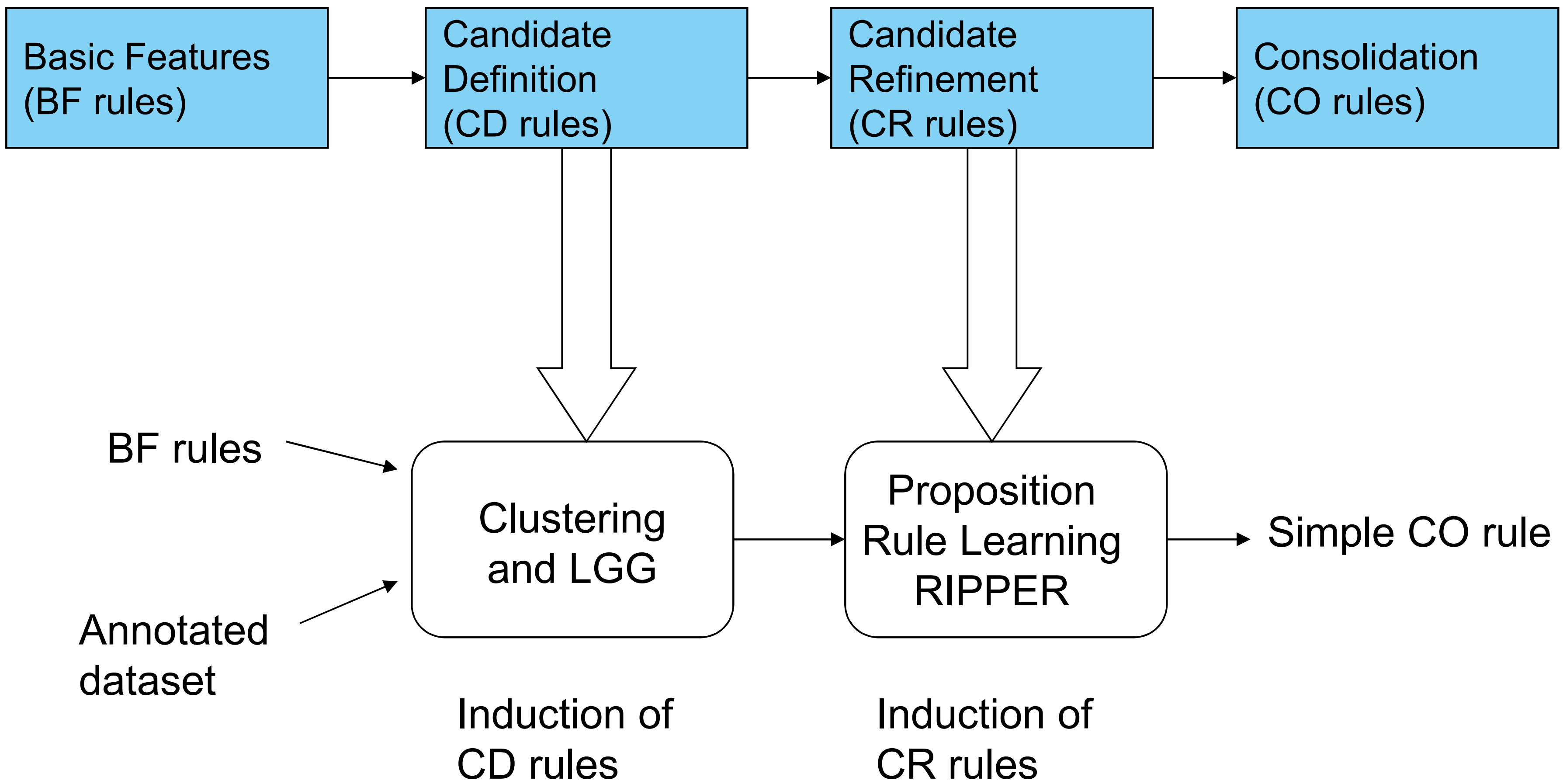
Anatomy of a Named Entity Extractor

Basic Features (BF rules) → Candidate Definition (CD rules) → Candidate Refinement (CR rules) → Consolidation (CO rules)

Document **… we met Ms. Anna Smith from Melinda Gates Foundation…**



Overview of Rule Induction System



First order representation of labeled data

<PER> $\overbrace{\underbrace{\textbf{M.}}_{X1}\ \underbrace{\textbf{Waugh}}_{X2}}^{X}$ </PER>

BF rules

--

Caps
LastNameDict
InitialDict

Textual Spans generated

--

Caps → *Waugh*
LastNameDict → *Waugh*
InitialDict → *M.*
...

First Order Logic predicates

--

```
Caps(X2), LastNameDict(X2),
InitialDict(X1)
```

+

Glue predicates

```
startsWith(X, X1)
endsWith(X, X2)
immBefore(X1, X2)
contains(Y, Y3)
equals(Z1, Z2)
```

First order representation

```
person(X, d1) :- startsWith(X, X1), InitialDict(X1),
                 endsWith(X, X2), immBefore(X1, X2), Caps(X2), LastNameDict(X2)
```

Induction of CD rules:
Least general generalisation (LGG) of annotations

PER: **john Smith** `person(X,D1) :- startsWith(X, X1), FirstNameDict(X1), endsWith(X, X2), immBefore(X1,X2), Caps(X2).`

PER: **John Doe** `person(Y,D2) :- startsWith(Y, Y1), FirstNameDict(Y1), Caps(Y1), endsWith(Y, Y2), immBefore(Y1,Y2), Caps(Y2).`

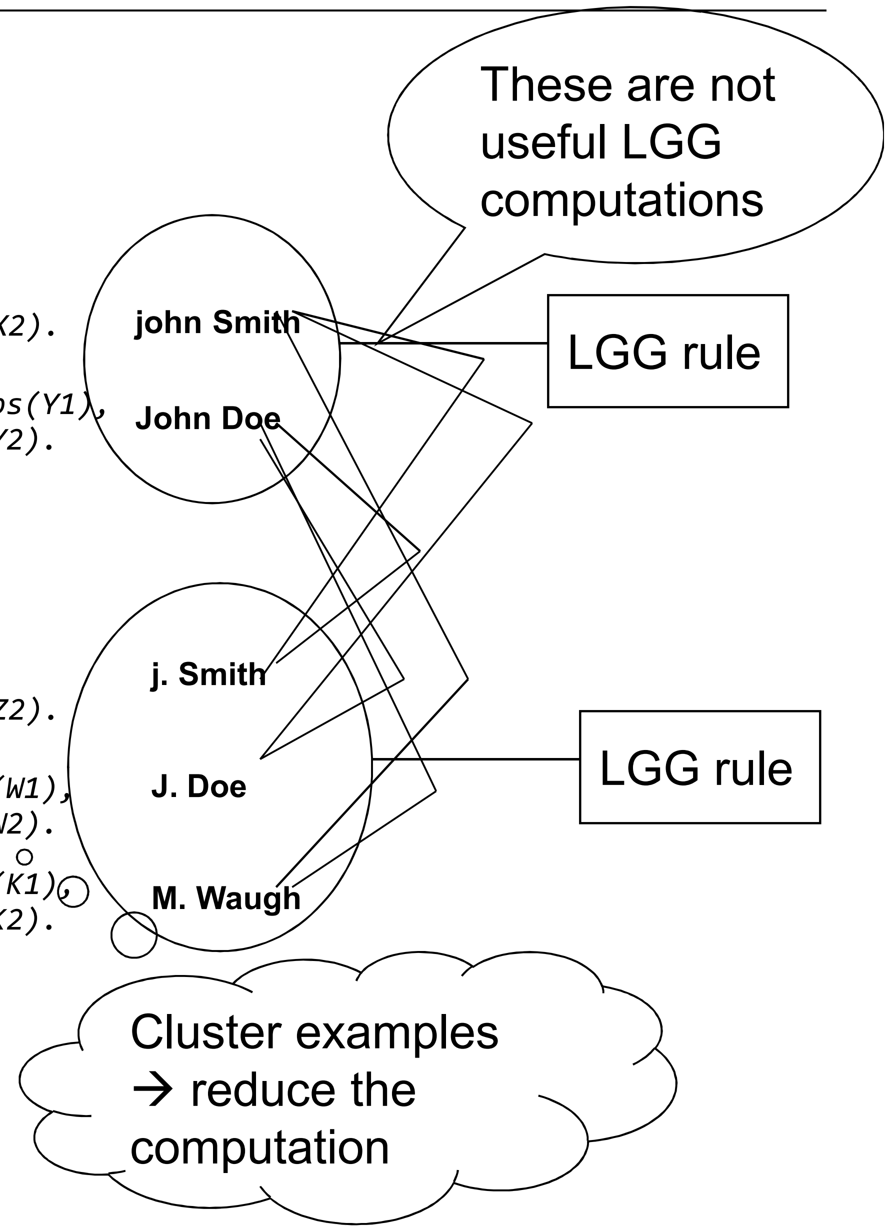
LGG of the above

`person(Z,D) :- startsWith(Z, Z1), FirstNameDict(Z1), endsWith(Z, Z2), immBefore(Z1,Z2), Caps(Z2)`

Clustering of Annotations

```
person(X,D1) :-  startsWith(X, X1), FirstNameDict(X1),
                 endsWith(X, X2), immBefore(X1,X2), Caps(X2).

person(Y,D2) :-  startsWith(Y, Y1), FirstNameDict(Y1), Caps(Y1),
                 endsWith(Y, Y2), immBefore(Y1,Y2), Caps(Y2).

....
....
....

person(Z,D1) :-  startsWith(Z, Z1), InitialDict(Z1),
                 endsWith(Z, Z2), immBefore(Z1,Z2), Caps(Z2).

person(W,D3) :-  startsWith(W, W1), InitialDict(W1), Caps(W1),
                 endsWith(W, W2), immBefore(W1,W2), Caps(W2).

person(K,D3) :-  startsWith(K, K1), InitialDict(K1), Caps(K1),
                 endsWith(K, K2), immBefore(K1,K2), Caps(K2).

....
....
```

Features for clustering are obtained from RHS of example clauses



Induction of CR rules

- Build a table encoding whether a span generated by one CD rule matches (M) or overlaps (O) with a span generated by any other CD rule
- Learn compositions of CD rules via the RIPPER propositional learner [Furnkranz and Widmer, 1994]

$$\mathtt{LOC} \leftarrow \boxed{\mathtt{(locCD_i\ =\ M)}}\ \mathtt{AND}\ \boxed{\mathtt{(orgCD_j\ !=\ O)}}$$

A span of text is a LOC

matches a Loc-CD rule

does not overlap with a org-CD rule

"***Washington***" in ***Washington Post*** will be filtered due to this rule

- Inductive Bias to model rule developer expertise and restrict the size of generated rules
  1. Disallow the BFs for one entity type from appearing in CD rules for another type
     - Avoids: ***PerCD*** *← [FirstNameDict][CapsPerson ^ CapsOrg]*
  2. Restriction of type of CD views that can appear in a CR
     - Avoids: ***PerCR*** *← (OrgCD = M) AND (LocCD != O)*

Supervised Learning of Predicate-based Rules

- Rule Induction: generate a rule program from basic features

- Rule refinement: refine an existing rule program
  - Refine rules [Liu et al., 2010]
  - Refine dictionaries used by the rules [Roy et al., 2013]

Rule Refinement [Liu et al. VLDB 2010]

R1: **create view** Phone **as**
Regex(‘d{3}-\d{4}’ , Document, text);

R2: **create view** Person **as**
Dictionary(‘first_names.dict’ , Document, text);

Dictionary file *first_names.dict*:
anna, james, john, peter…

R3: **create table** PersonPhone(match *span*);

insert into PersonPhone
select Merge(F.match, P.match) **as** match
from Person F, Phone P
where Follows(F.match, P.match, 0, 60);

- Rules expressed in SQL
  - Select, Project, Join, Union all, Except all
  - Text-specific extensions
    - Regex, Dictionary table functions
    - New selection/join predicates
  - Can express core functionality of IE rule languages
    - AQL, CPSL, XLog
- Relational data model
  - Tuples and views
  - New data type *span*: region of text in a document

Document:

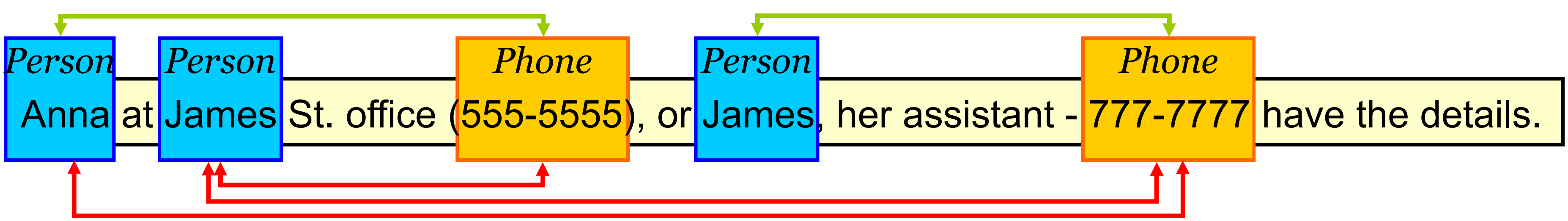
text
Anna at James St. office (555-5555), or James, her assistant – 777-7777 have the details.

Phone:

match
555-5555
777-7777

Person:

match
Anna
James
James



Rule Refinement [Liu et al. VLDB 2010]

R1: **create view** Phone **as**
Regex(‘d{3}-\d{4}’ , Document, text);

R2: **create view** Person **as**
Dictionary(‘first_names.dict’ , Document, text);

Dictionary fil
anna, james, j

R3: **create table** F

insert into Pe
select Merge(
from Perso
where Follow

- Rules expressed in SQL
  - Select, Project, Join, Union all, Except all
  - Text-specific extensions
    - Regex, Dictionary table functions
    - New selection/join predicates
  - Can express core functionality of IE rule languages

a document

Challenges

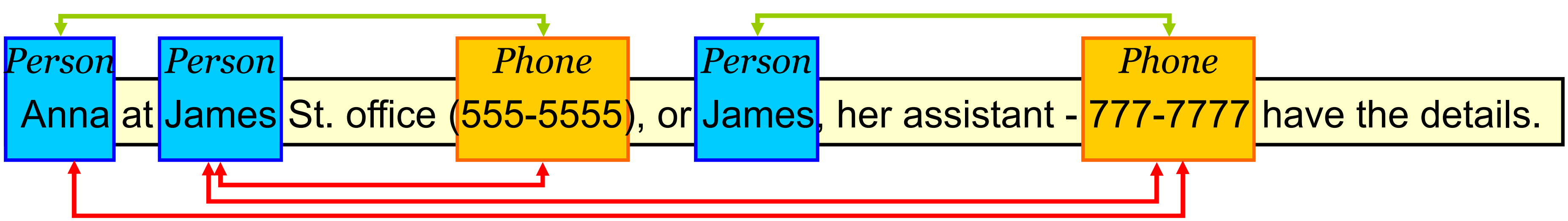
- Which rule to refine and how?
- What are the effects and side-effects?

– 777-7777 have the details.

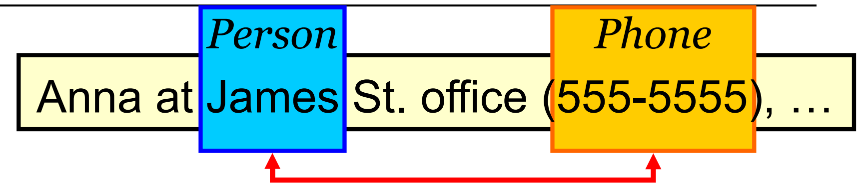
555
777

Person:

match
Anna
James
James



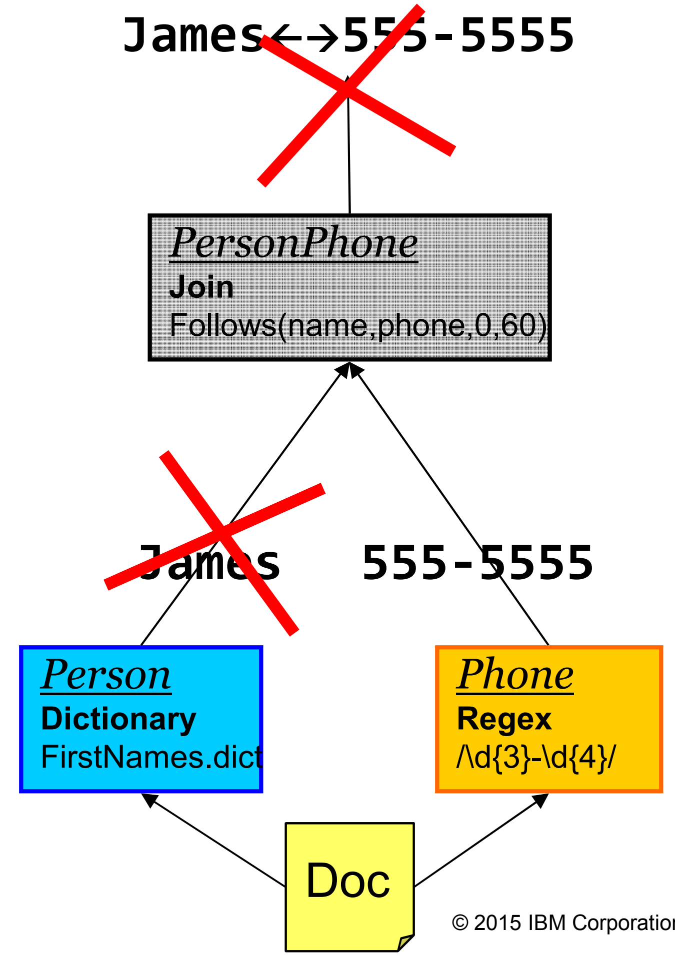
Method Overview



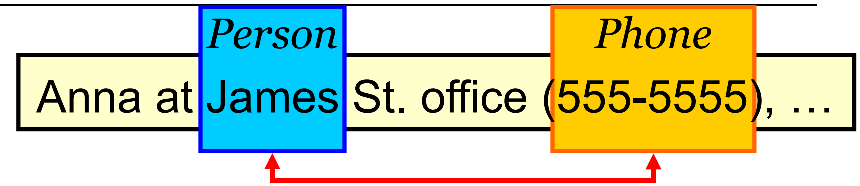
- Framework for systematic exploration of multiple refinements geared towards improving precision
- Input: Extractor P

  Results of P, fully labeled
- Goal: Generate refinements of P that remove false positives, while not affecting true positives
- Basic Idea:
  Cut any provenance link → wrong output disappears

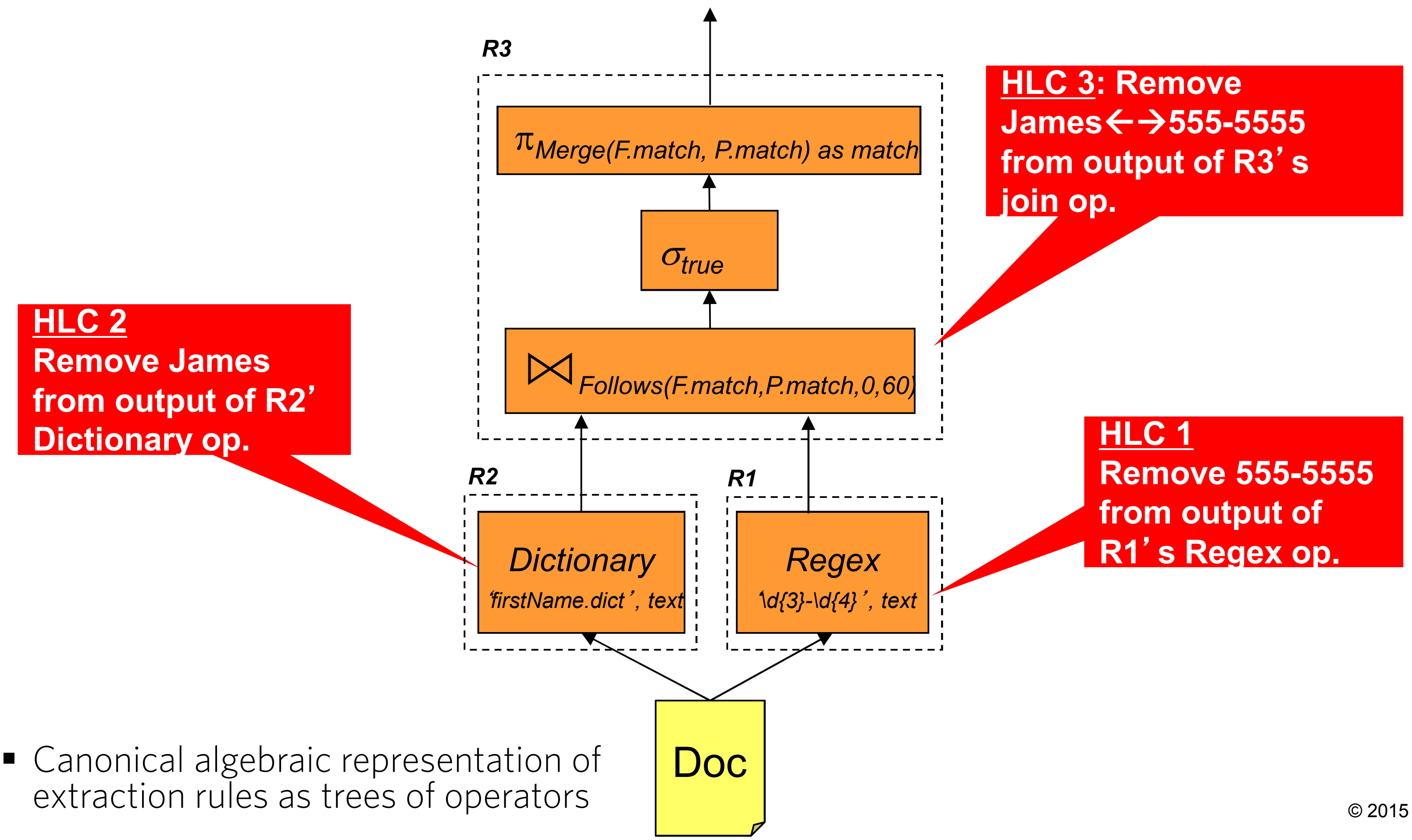
(Simplified) provenance of a wrong output



High Level Changes: What Operator to Modify ?

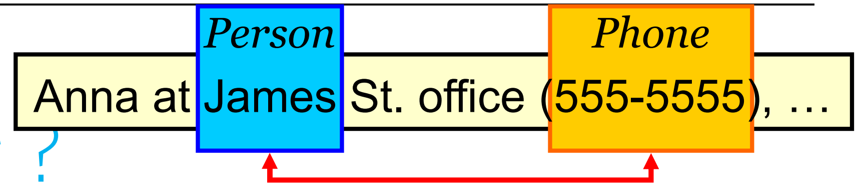


Goal: remove "James ←→ 555-5555" from output

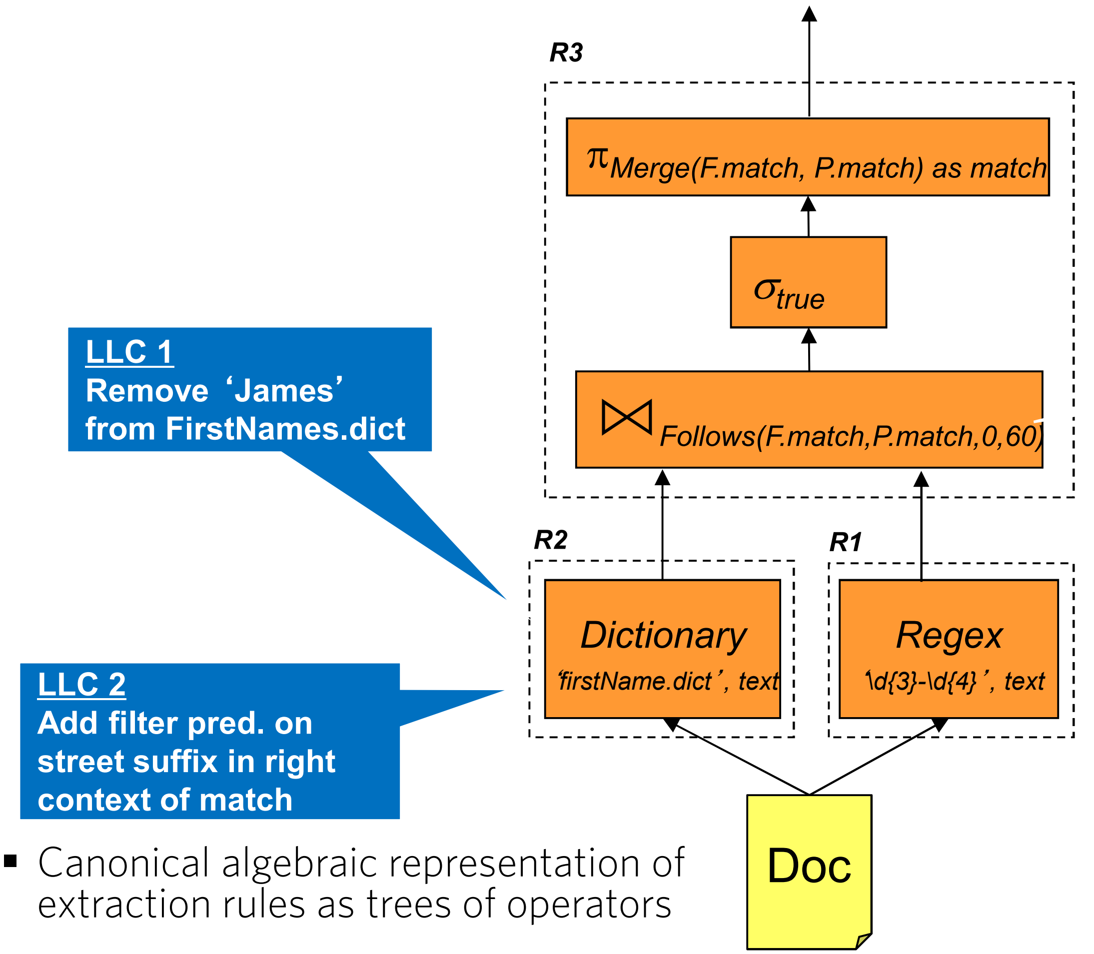


- Canonical algebraic representation of extraction rules as trees of operators

Low-Level Changes: How to Modify the Operator ?



Goal: remove "James ←→ 555-5555" from output



- Canonical algebraic representation of extraction rules as trees of operators

Types of Low-Level Changes

1. Modify numerical join parameters - implements HLCs for $\bowtie$

2. Remove dictionary entries - implements HLCs for **Dictionary**, $\sigma_{ContainsDict()}$
   - More on this later

3. Add filtering dictionary - implements HLCs for σ
   - Parameters: target of filter (match, or left/right context)

4. Add filtering view - applies to an entire view
   - Parameters: filtering view, filtering mode (*Contains*, *IsContained*, *Overlaps*)
   - E.g., "Subtract from the result of rule R3 *PersonPhone* spans that are strictly contained within another *PersonPhone* span"

- Other LLC generation modules can be incorporated

Computing Model-level Provenance

- (Model-level) Provenance: Explains output data in terms of the input data, the intermediate data, and the transformation (e.g., SQL query, ETL, workflow)
  - Surveys: [Davidson & Freire, SIGMOD 2008] [Cheney et al., Found. Databases 2009]
- For predicate-based rule languages (e.g., SQL), can be computed automatically!

PersonPhone rule:

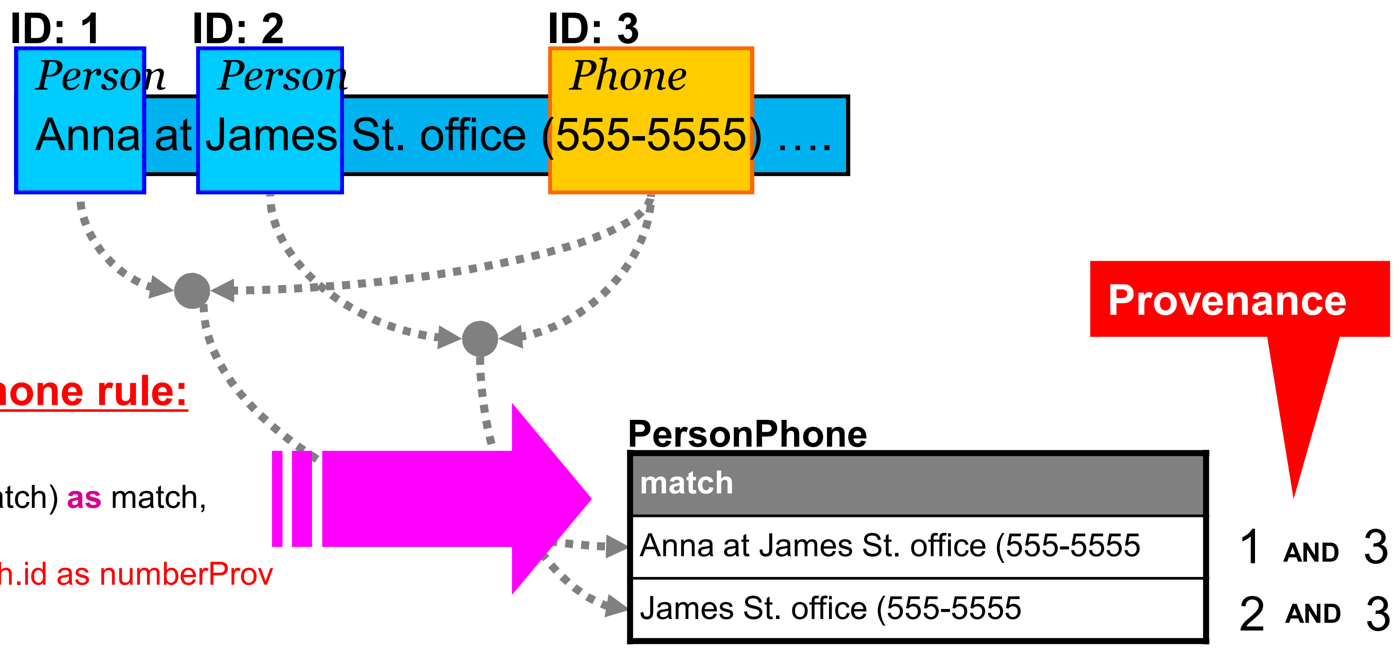
```
insert into PersonPhone
select Merge(F.match, P.match) as match
from    Person F, Phone P
where Follows(F.match, P.match, 0, 60);
```

PersonPhone

match
Anna at James St. office (555-5555
James St. office (555-5555

Computing Model-level Provenance

- (Model-level) Provenance: Explains output data in terms of the input data, the intermediate data, and the transformation (e.g., SQL query, ETL, workflow)
  – Surveys: [Davidson & Freire, SIGMOD 2008] [Cheney et al., Found. Databases 2009]
- For predicate-based rule languages (e.g., SQL), can be computed automatically!

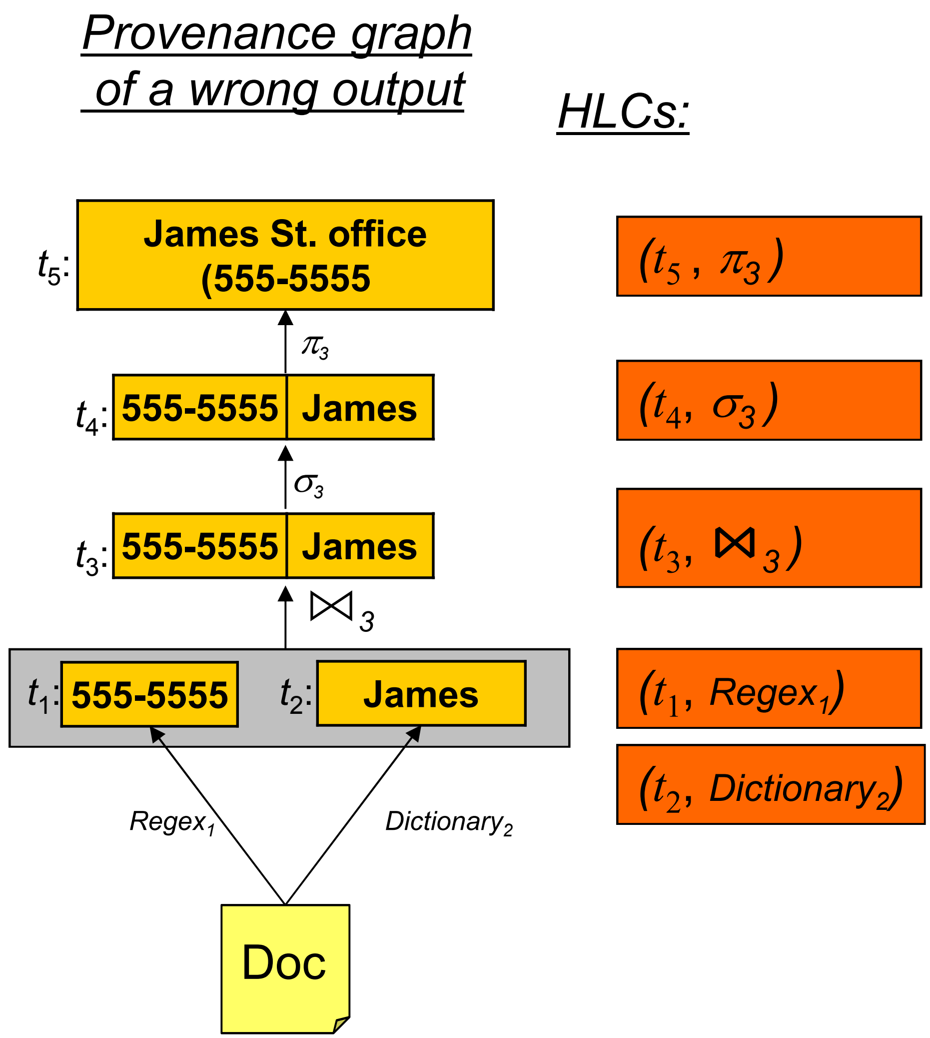


Rewritten PersonPhone rule:

```
insert into PersonPhone
select Merge(F.match, P.match) as match,
       GenerateID() as ID,
       P.id as nameProv, Ph.id as numberProv
       'AND' as how
from   Person F, Phone P
where Follows(F.match, P.match, 0, 60);
```

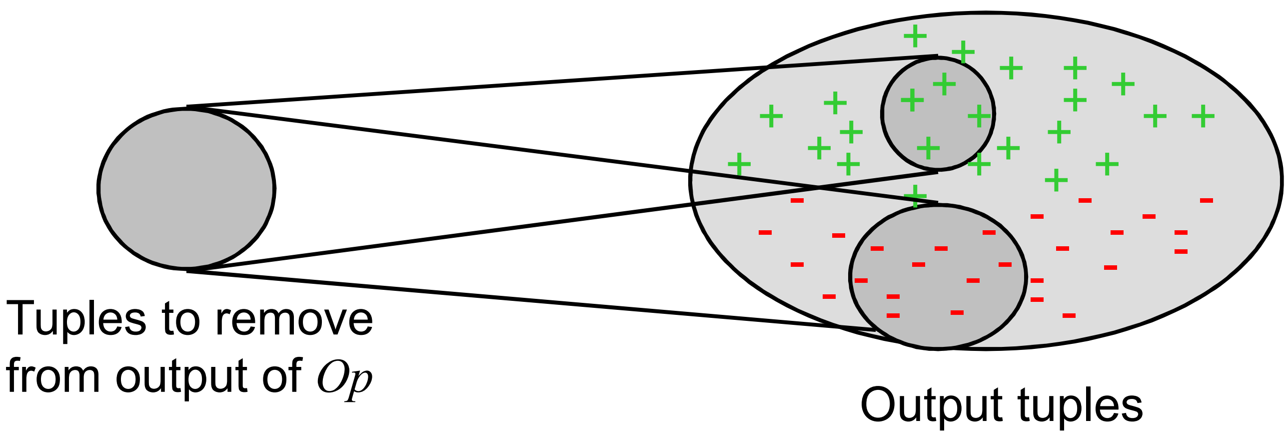
Generating HLCs and LLCs

- **HLCs:** compute directly from provenance graph and negative examples
- **LLCs:** Naive approach
  - For each HLC *(t_i, Op)*, enumerate all possible LLCs
  - For each LLC:
    - Compute set of local tuples it removes from the output of *Op*
    - Propagate removals up the provenance graph to compute the effect on end-to-end result
  - Rank LLCs based on improvement in F1



Problems with the Naïve Approach

- Problem 1: Given an HLC, the number of possible LLCs may be large
  - E.g., HLC is (*t*, ***Dictionary***), 1000 dictionary entries → 2^{999}-1 possible LLCs !
- Solution: Limit the LLCs considered to a set of tractable size, while still considering all feasible combinations of HLCs for ***Op***
  - Generate a single LLC for each of ***k*** promising combinations of HLCs for ***Op***
  - ***k*** is the number of LLCs presented to the user
- Problem 2: Traversing the provenance graph is expensive
  - $O(n^2)$, where ***n*** is the size of the operator tree
- Solution: For each ***Op*** and tuple t_i in the output of ***Op***, remember mapping t_i → {set of affected output tuples}



LLC Generation: Learning a Filter Dictionary

Output of σ operator		Final output of *Person* extractor
James	→	**James St**
Morgan	→	**Morgan Ave**
June	→	**June Blvd**
Anna	→	**Anna Karenina Blvd**
Hall	→	**Hall St**

Common token in right context		Effects of filtering with the token
‘st’	→	**James St**
		Hall St
‘blvd’	→	**June Blvd**
		Anna Karenina Blvd
‘ave’	→	**Morgan Ave**

Generated LLCs:

Add ***ContainsDict(‘SuffxDict’, RightContextTok(match,2))*** **to σ operator, where** ***SuffixDict*** **contains:**

1. ‘st’
2. ‘st’ ,’ blvd’
3. ‘st’ , ‘blvd’ ,’ ave’

Supervised Learning of Predicate-based Rules

- Rule Induction: generate a rule program from basic features

- Rule refinement: refine an existing rule program
  - Refine rules [Liu et al., 2010]
  - Refine dictionaries used by the rules [Roy et al., 2013]

Dictionary Refinement Problem [Roy et al, SIGMOD 2013]

"………This **April**, mark your calendars for the first derby of the season: Arsenal at **Chelsea**. ……………………,..**April Smith** and **John Lee** reporting live from …….. **David** said that……"

~~April~~		w_3
~~Chelsea~~	×	w_1
April Smith	✓	$w_5 + w_3\, w_4$
John Lee	✓	$w_2\, w_6$
David	✓	w_7

Input:

- Predicate-based rule program (SQL-like)
- Boolean model-level provenance of each result
- ✓ / × Label of each result

We also studied the incomplete labeling case

Goal: Maximize F-score

Select a set ***S*** of entries to remove from dictionaries

… that maximizes the new F-score

… subject to $|S| \le k$

new recall $\ge r$

Size Constraint
(limit #deleted entries)

Recall Constraint
(limit #true positives deleted)

Possible output

S = { w1: chelsea, w3: april }

New F-score = 1 ☺

Dictionary Refinement Problem [Roy et al, SIGMOD 2013]

"………This **April**, mark your calendars for the first derby of the season: Arsenal at **Chelsea**. ……………………,..**April Smith** and **John Lee** reporting live from …….. **David** said that……"

~~April~~		w_3
~~Chelsea~~	✗	w_1
April Smith	✓	$w_5 + w_3 w_4$
John Lee	✓	
David		

Input:

- Predicate-based rule program (SQL-like)
- Boolean model-level provenance of each result
- ✓ / ✗ Label of each result

Challenges

- Complex input-output dependencies
- Complex objective function

Possible output

$$S = \{ \text{w1: chelsea}, \text{w3: april} \}$$

New F-score = 1 ☺

Goal: Maximize F-score

Select a set ***S*** of entries to remove from dictionaries

… that maximizes the new F-score

… subject to **|*S*| ≤ *k***

… new recall ≥ ***r***

Size Constraint
(limit #deleted entries)

Recall Constraint
(limit #true positives deleted)

Complex Objective Function

Both numerator and denominator depend on S

(even if we try to rewrite the expression)

$$\text{New F-score after deleting S} = \frac{2 * G_{-s}}{G_o + G_{-s} + B_{-s}}$$

G_o = original #true positives

G_{-s} = remaining #true positives after deleting S

B_{-s} = remaining #false positives after deleting S

Results: Simple Rules

- Provenance has a simple form
- One input to many results

	Simple Rules **Provenance: w**
Size constraint **$\|S\| \leq k$**	Optimal Algorithm
Recall constraint (remaining true positives after deleting $S \geq r$)	NP-hard (reduction from the subset-sum problem) "Near optimal" Algorithm (simple, provably close to optimal)

Some details next

Sketch of Optimal Algorithm

for Simple Rules, Size Constraint |S| ≤ k

1. Guess the optimal F-score θ

2. Verify if there exists a subset S, |S|≤ k, giving this F-score θ

3. Repeat by binary-search in [0, 1] until the optimal θ is found

$$F_{-s} = \frac{2 * G_{-s}}{G_o + G_{-s} + B_{-s}} \geq \theta$$

$$G_{-s}(2 - \theta) - \theta B_{-s} - \theta G_o \geq 0$$

$G_{-s} = G_o - \Sigma_{w \in S} G_w$

$B_{-s} = B_o - \Sigma_{w \in S} B_w$

Binary search on real numbers in [0, 1] **(still poly-time)**

$$\Sigma_{w \in S} f(G_w, B_w) \geq Const, \quad \text{where } |S| \leq k$$

Top-k problem, **poly-time!**

Does not work for general case (many-to-many)

Results: Complex Rules

- Arbitrary extraction rules
- Arbitrary provenance
- Many to many dependency

	Simple Rules Provenance: w	**Complex Rules** Provenance: $w_1 + w_2 w_3 + w_4$
Size constraint $\|S\| \leq k$	Optimal Algorithm	NP-hard even for two dictionaries (reduction from the k-densest subgraph problem)
Recall constraint (bound on the true positives retained)	NP-hard "Near optimal" Algorithm	

- **Efficient Heuristics**

•Sketch:

•Find an initial solution

•Improve solution by hill-climbing

So far we assumed all results are labeled as true positive / false positive

What if not all the results are labeled?

…ignoring unlabeled results may lead to over-fitting

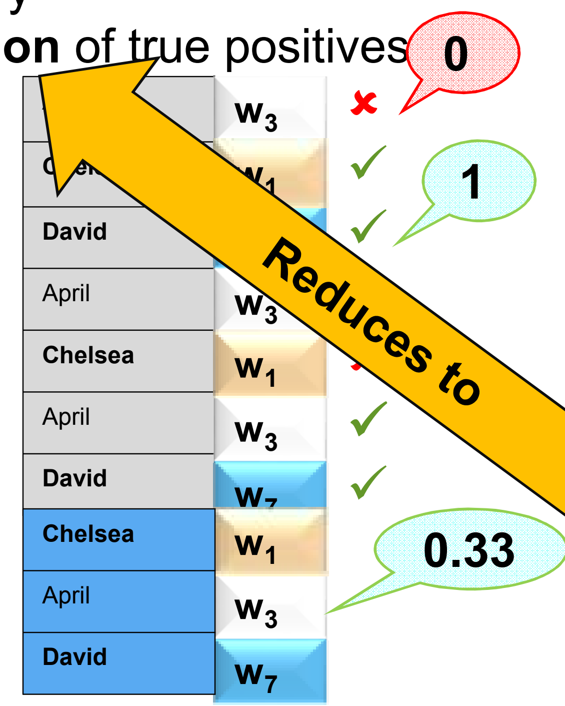
Estimating Missing Labels

Simple Rules

Possible approach:

Label of an entry =
Empirical **fraction** of true positives

w_3 april:	**0.33**
w_1 chelsea:	0.50
w_7 david:	1.00



Complex Rules

April Smith	$w_5 + w_3 w_4$
John Lee	$w_2 w_6$
David	w_3

Empirical estimation does not work!

- Arbitrary monotone Boolean expressions
- Very few or no labels available!

We assume a statistical model and estimate labels using **Expectation-Maximization** algorithm

Transparent ML in Learning of Predicate-based Rules

- Transparency in Model of Representation
  - Predicate-based rules, completely declarative
  - Model-level provenance computed automatically
  - Interesting issue: Interpretability of program
    - Induced program is declarative, but there is a more subjective aspect of "code quality"
      → Two equivalent programs may have very different levels of "interpretability"
    - Applies primarily to Rule Induction
    - Applies to Rule Refinement to a considerable smaller extent because: (1) learning is constrained by the initial program, and (2) user guides the learning interactively
    - Initial investigation [Nagesh et. Al, 2012]; more work is needed

- Transparency in Learning Algorithm
  - Some transparency in terms of the user influencing the model
    - Rule Induction → inductive bias
    - Rule Refinement → user selects among suggested refinements

- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline (Rule Induction) or Interactive (Rule Refinement)
  - Easy to incorporate DK at deployment (by further modifying the rules)

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

FlashExtract [Le & Gulwani, PLDI 2014]

- **Goal**: Data Extraction from semi-structured text documents
- **User Interaction**: Positive/negative examples of rectangular regions on a document
  - Interactive
- Different colors & nested regions enables data extraction into a data structure with struct/sequence constructs

Seq([blue] Struct(Name: [green] String,
City: [yellow] String))

- Techniques borrowed from program synthesis

Label 1	Label 2
Ana Trujillo	Redmond
Antonio Moreno	Renton

FlashExtract: Learning Algorithm

- **Model of Representation:** Program consisting of core operations:
  - Map, Filter, Merge, Pair
- **Learning Algorithm:** Inductive on the grammar structure
  - Learn programs from positive examples
  - Discard those that capture the negative examples
- Learn city extractor = learn a Map operator
  - The **lines** that hold the city
  - The **pair** that identifies the city within a line
- Learn lines = learn a Boolean filter

Ana Trujillo
357 21th Place SE
Redmond, WA
(757) 555-1634

Antonio Moreno
515 93th Lane
Renton, WA
(411) 555-2786

FlashExtract: City Extractor

1. Filter lines that end with "WA"

Ana Trujillo
357 21th Place SE
Redmond, WA
(757) 555-1634

Antonio Moreno
515 93th Lane
Renton, WA
(411) 555-2786

FlashExtract: City Extractor

1. Filter lines that end with "WA"

2. Map each selected line to a pair of positions

FlashExtract: City Extractor

1. Filter lines that end with “WA”
2. Map each selected line to a pair of positions
3. Learn two leaf expressions for the start/end positions
   - Begin of line
   - ‘,’

Transparent ML in FlashExtract

- Transparency in Model of Representation
  – Simple domain-specific language → easy to comprehend
  – Language is imperative → no model-level provenance
    - Output can be explained only by watching program execution

- Transparency in Learning Algorithm
  – No transparency

- Transparency in Incorporation of Domain Knowledge (DK)
  – Interactive
  – Can incorporate DK at deployment (by further modifying the program)

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

Rule Learning: Unsupervised

- Traditional IE: Pattern Discovery [Li et al., CIKM 2011]

- Open IE: ClauseIE [DelCorro & Gemulla, WWW 2013]

Rule Learning: Unsupervised

- Traditional IE: Pattern Discovery [Li et al., CIKM 2011]

- Open IE: ClauseIE [DelCorro & Gemulla, WWW 2013]

Pattern Discovery [Li et al., CIKM 2011]

- Manually identify patterns → tedious + time consuming
  - ⟨PERSON⟩ .* at .* ⟨PHONE_NUMBER⟩
  - ⟨PERSON⟩'s (cell|office|home)? number is ⟨PHONE_NUMBER⟩

- Basic idea:
  - Group similar strings together to facilitate pattern discovery

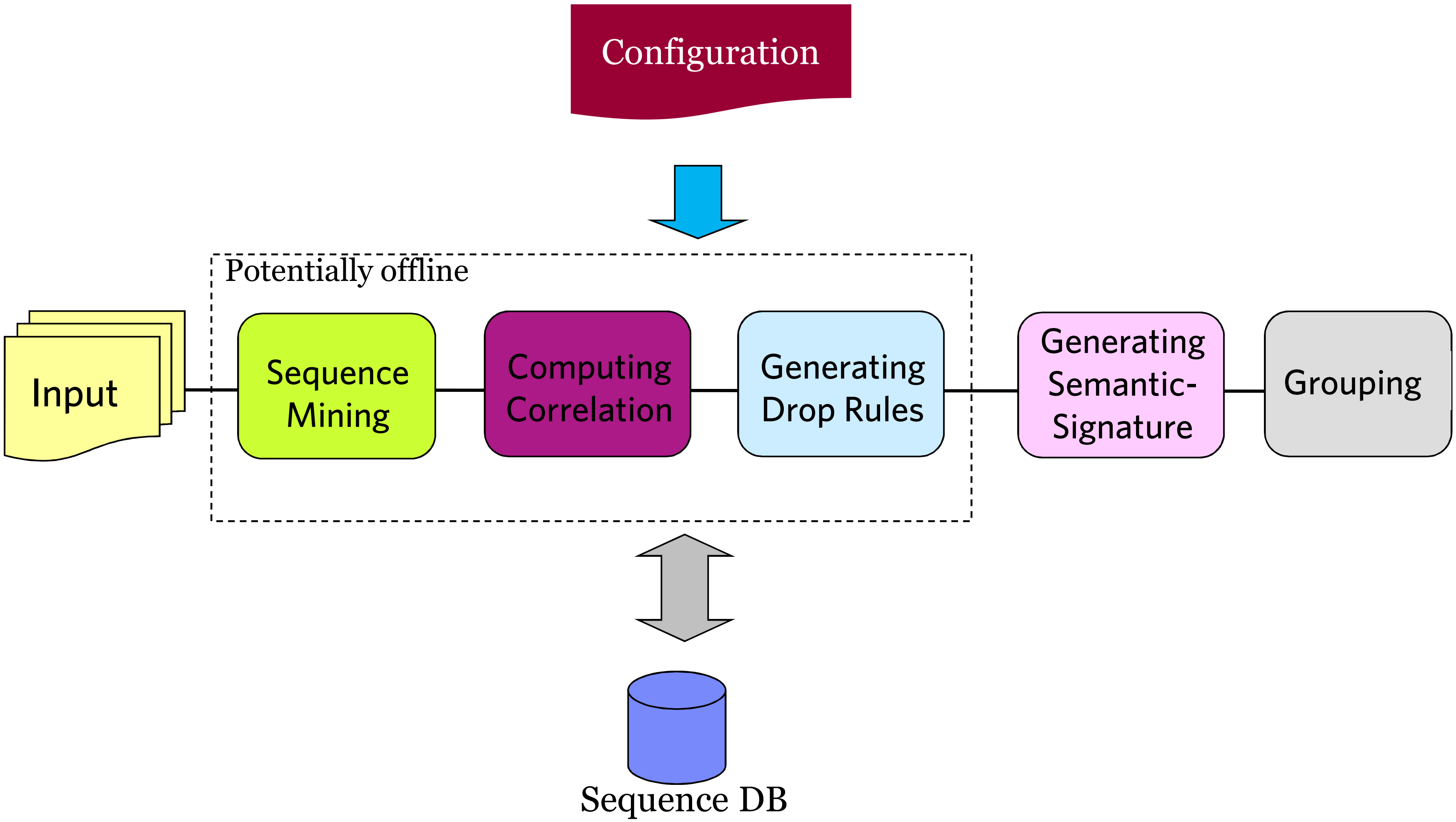
Kristen's phone number is (281)584-1405
Andrea Walter's office number is x345763
...

→
⟨PERSON⟩'s (cell|office|home)? Number is ⟨PHONE_NUMBER⟩

Practical Requirements

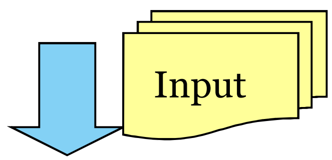
- Configurable
  - Grouping may be done along multiple aspects of the data
- Declarative
  - Providing justification for group membership for debugging
- Scalable
  - We expect to have many instances and possibly many groups

Overview: Clustering based on Semantic-Signature



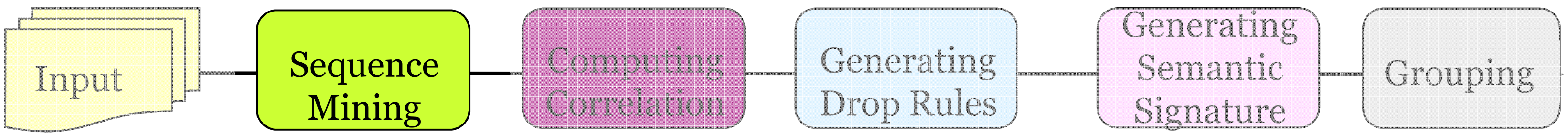
Running Example: Person Phone

- John can be reached at (408)123-4567
- Jane can be reached at her cell (212)888-1234
- Mr. Doe can also be reached at (123)111-2222
- Mary may be reached at her office # (111)222-3333



ID	Input Contextual String
1	can be reached at
2	can be reached at her cell
3	can also be reached at
4	may be reached at her office #

Step 1. Sequence Mining



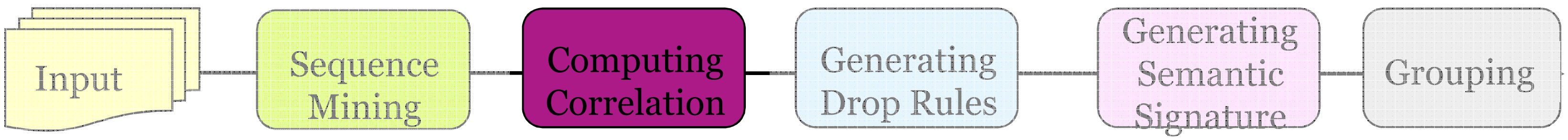
- Configurable by
  - f_{min}: Minimum support of the sequence
  - l_{min}: Minimum sequence length
  - l_{max}: Maximum sequence length

Example: Given $f_{min}=3$, $l_{min}=1$, $l_{max}=2$

ID	Input Contextual String
1	can be reached at
2	can be reached at her cell
3	can also be reached at
4	may be reached at her office #

Sequence
can
be reached
reached at
be
at

Step 2. Computing Correlation



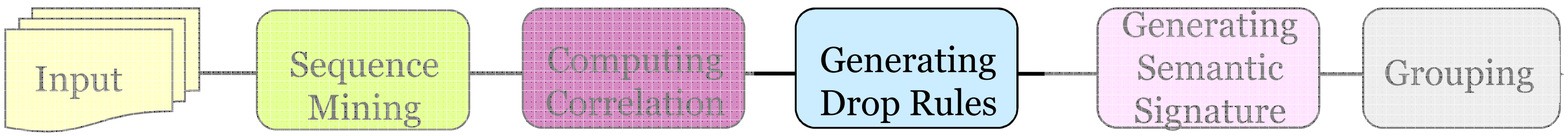
- Different measures of correlation can be used
  - The presence of one sequence predicates the other
  - Uncertainty Coefficient

$$U(x|y) = I(x, y)/H(x)$$

Example

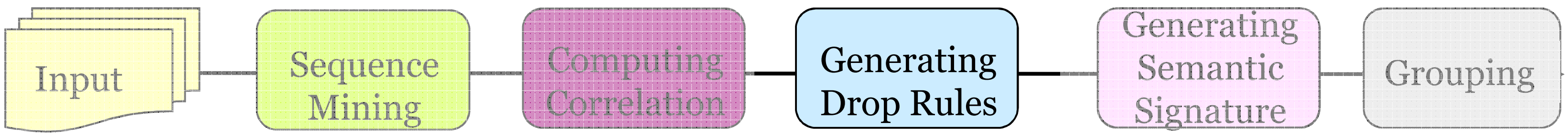
Sequence X	Sequence Y	U(X|Y)	U(Y|X)
can	be reached	0.946	0.750
be reached	at	0.022	0.277
can	at	0.029	0.293

Step 3. Generating Drop Rules - I



- Rule format:
  - DROP sequence X IF sequence X AND sequence Y (present in the same contextual string)
- Generated based on threshold over correlation measure

Step 3. Generating Drop Rules - II



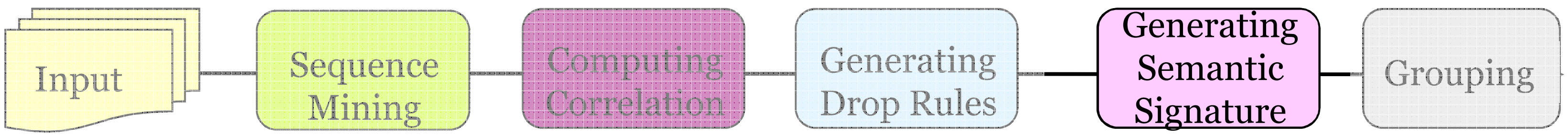
Example: If U(X|Y) > 0.25 or U(Y|X) > 0.25, generate a drop rule

Sequence X	Sequence Y	U(X|Y)	U(Y|X)
can	be reached	0.946	0.750
be reached	at	0.022	0.277
can	at	0.029	0.293

DROP “can” IF “can” AND “be reached”
DROP “be reached” IF “can” AND “be reached”
DROP “at” IF “be reached” AND “at”
DROP “at” IF “can” AND “at”

Confidence score

Step 4. Generating Semantic Signature



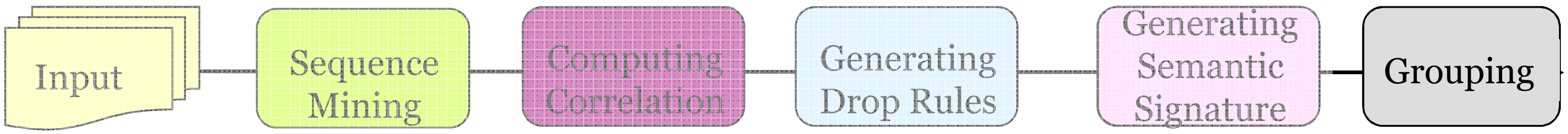
- Applying drop rules in the decreasing order of their associated confidence score

Example:

DROP “can” IF “can” AND “be reached”
DROP “be reached” IF “can” AND “be reached”
DROP “at” IF “be reached” AND “at”
DROP “at” IF “can” AND “at”

~~can~~; be reached; ~~at~~

Step 5. Grouping



- Step 1: Sequences with the same semantic signature form a group
- Step 2: Further merge groups of small size with similar semantic signatures to those of the larger ones
  - → reduce the number of clusters to be examined

Transparent ML in Pattern Discovery

- Transparency in Model of Representation
  - Sequence Patterns
  - Model-level Provenance

- Transparency in Learning Algorithm
  - Some algorithm-level provenance: final sequences can be explained through the chain of drop rules
  - User can influence the model through the initial configuration

- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline
  - But, easy to incorporate domain knowledge at deployment (by further modifying the rules)

Rule Learning: Unsupervised

- Traditional IE: Pattern Discovery [Li et al., CIKM 2011]

- Open IE: ClauseIE [DelCorro & Gemulla, WWW 2013]

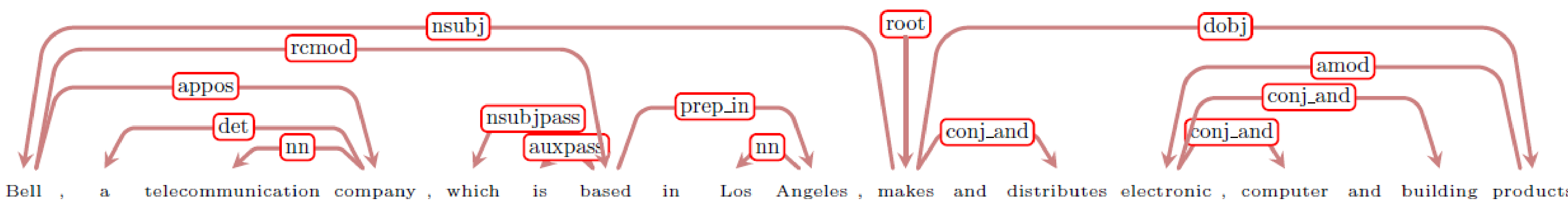
ClausIE [Del Corro & Gemulla, WWW 2013]

- **Goal:** Separate the identification of information from its representation

- Identifies essential and optional arguments in a clause
  - 7 essential clauses: SV, SVA, SVO, SVC, $SVOO_{ind}$, SVOA, SVOC
  - A minimal clause is a clause without the optional adverbials (A)

- Algorithm
  1. Clause Identification: Walk the dependency tree and identify clauses using a deterministic flow chart of decision questions
  2. Proposition Generation: For each clause, generate one or more propositions

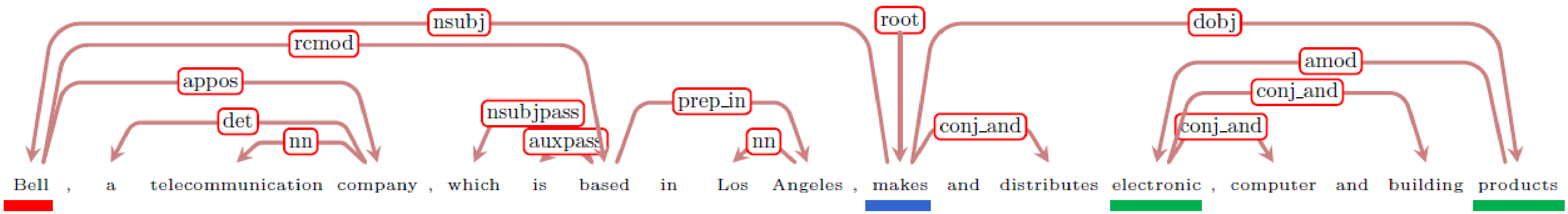
ClausIE: Example



Bell, a telecommunication company, which is based in Los Angeles, makes and distributes electronic and building products.

(S: Bell, V: 'is', C: a telecommunication company)
(S: Bell, V: is based, A: in Los Angeles)
(S: Bell, V: makes, O: electronic products)
(S: Bell, V: makes, O: computer products)
(S: Bell, V: makes, O: building products)
(S: Bell, V: distributes, O: electronic products)
(S: Bell, V: distributes, O: computer products)
(S: Bell, V: distributes, O: building products)

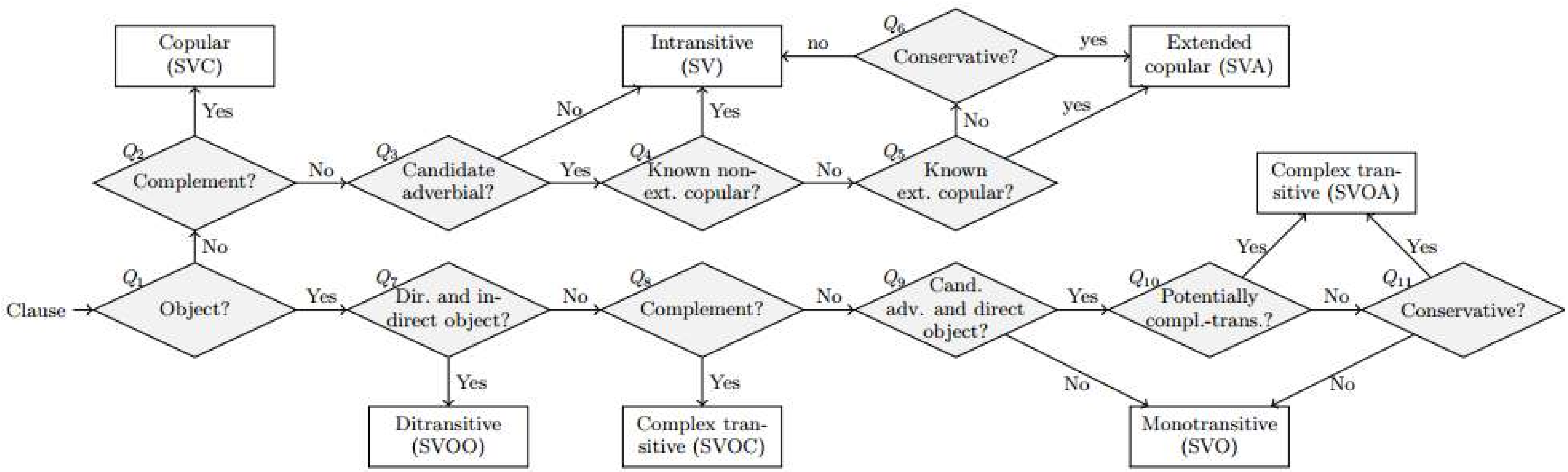
ClausIE: Example



Bell, a telecommunication company, which is based in Los Angeles, makes and distributes electronic and building products.

(S: Bell,	V: 'is',	C: a telecommunication company)
(S: Bell,	V: is based,	A: in Los Angeles)
(S: Bell,	V: makes,	O: electronic products)
(S: Bell,	V: makes,	O: computer products)
(S: Bell,	V: makes,	O: building products)
(S: Bell,	V: distributes,	O: electronic products)
(S: Bell,	V: distributes,	O: computer products)
(S: Bell,	V: distributes,	O: building products)

Clause Identification Flow Chart



Transparent ML in ClausIE

- Transparency in Model of Representation
  - Essential clauses = abstraction of dependency path patterns
  - Easier to comprehend compared to path patterns
  - Model-level provenance (partial):
    - Can connect an extracted object with the part of the model (i.e., clause) that determined it
    - Comprehending why the clause matches the parse tree of the input text requires reasoning about the clause identification flow chart

- Transparency in Learning Algorithm
  - User can influence the model through customizing the types of generated propositions
    - Type of relations: *Messi plays in Barcelona* → *plays* or *plays in*
    - Triples or n-ary propositions: (Messi, plays football in, Barcelona) or (Messi, plays, football, in Barcelona)

- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

Fact Extraction: Supervised

- AutoSlog-SE [Choi et al., EMNLP 2005]: Identifying sources of opinions with CRF and extraction patterns

AutoSlog heuristics — Set of extraction patterns that, collectively, can extract every NP in the training corpus.

↓

Semantic restriction — Semantically constrain the types of noun phrases that are legitimate extractions for opinion sources

↓

Apply patterns to corpus; gather statistics — Count number of correct and incorrect extractions for each pattern; estimate probability that the pattern will extract an opinion source in new texts

↓

Basic features: orthographic, lexical, syntactic, semantic → **CRF** — Incorporate extraction patterns as features to increase recall of CRF model.

Transparent ML in AutoSlog-SE

- Transparency in Model of Representation
  - Path patterns + CRF
  - Model-level provenance (partial)
    - Provenance at the level of patterns
    - No provenance at the level of the CRF → overall, cannot explain an extracted object

- Transparency in Learning Algorithm
  - CRF training is not transparent

- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline

Transparent ML Techniques

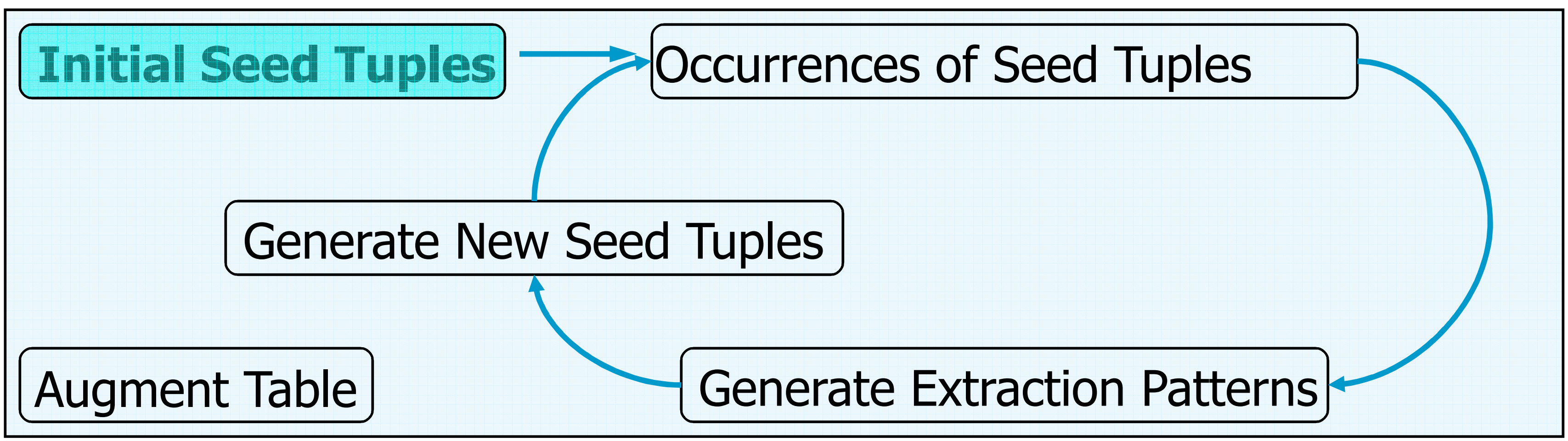
	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

Semi-supervised (using Bootstrapping) Relation Extraction

Example Task: Organization "located in" Location

Initial Seed Tuples:

ORGANIZATION	***LOCATION***
MICROSOFT	REDMOND
IBM	ARMONK
BOEING	SEATTLE
INTEL	SANTA CLARA



Slide from Eugene Agichtein

Semi-supervised (using Bootstrapping) Relation Extraction

Occurrences of seed tuples:

ORGANIZATION	LOCATION
MICROSOFT	REDMOND
IBM	ARMONK
BOEING	SEATTLE
INTEL	SANTA CLARA

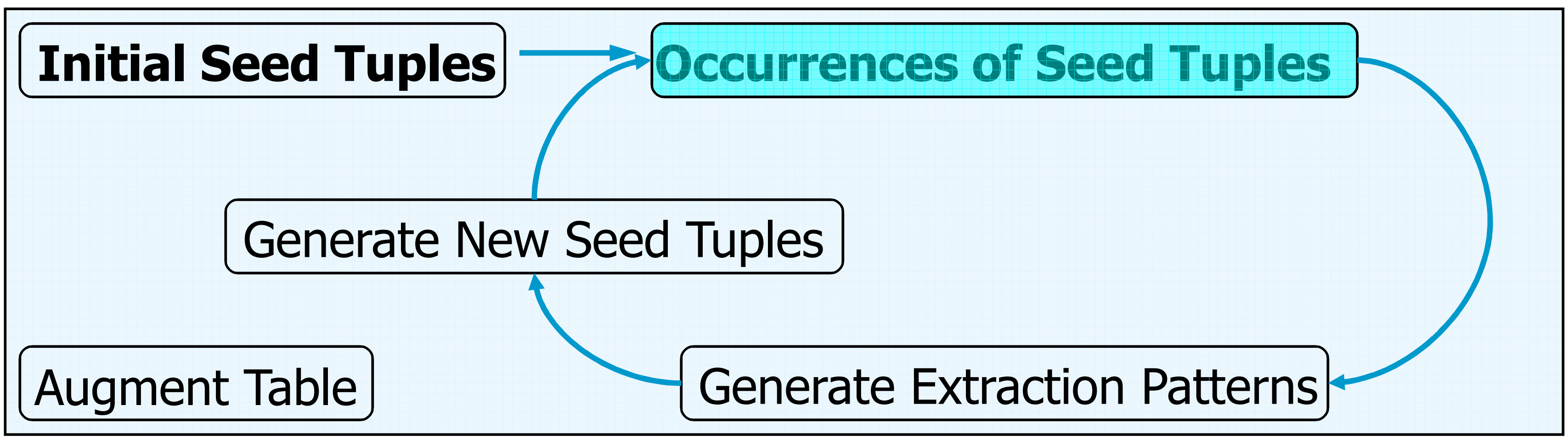
Computer servers at **Microsoft**'s headquarters in **Redmond**…

In mid-afternoon trading, share of **Redmond**-based **Microsoft** fell…

The **Armonk**-based **IBM** introduced a new line…

The combined company will operate from **Boeing**'s headquarters in **Seattle**.

Intel, **Santa Clara**, cut prices of its Pentium processor.

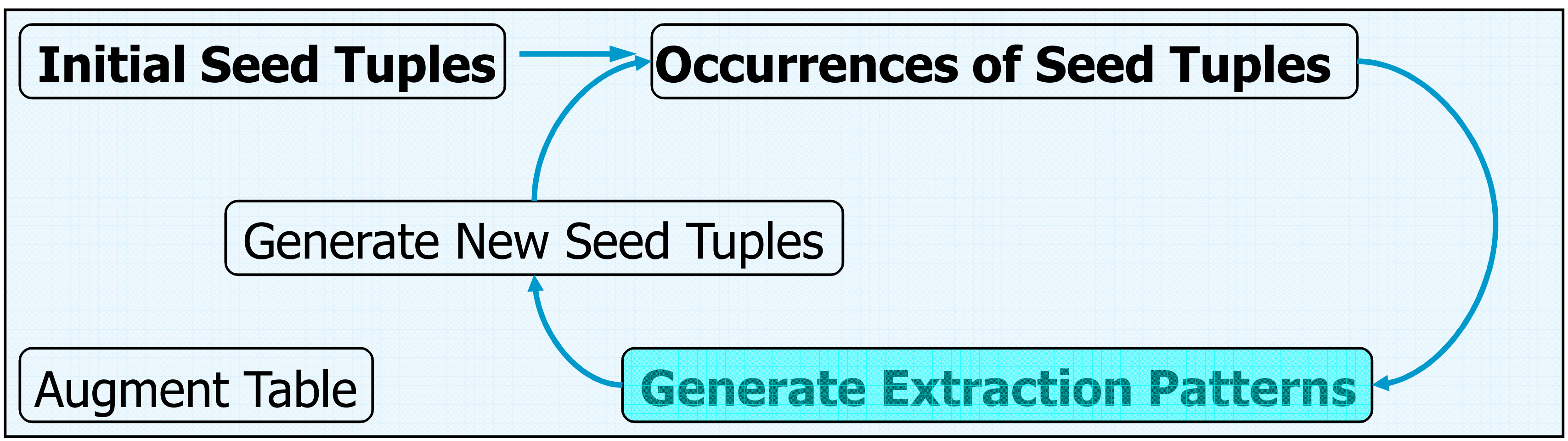


Slide from Eugene Agichtein

Semi-supervised (using Bootstrapping) Relation Extraction

DIPRE Patterns
[Brin, WebDB 1998]

- <*STRING1*>'s headquarters in <*STRING2*>
- <*STRING2*> -based <*STRING1*>
- <*STRING1*> , <*STRING2*>

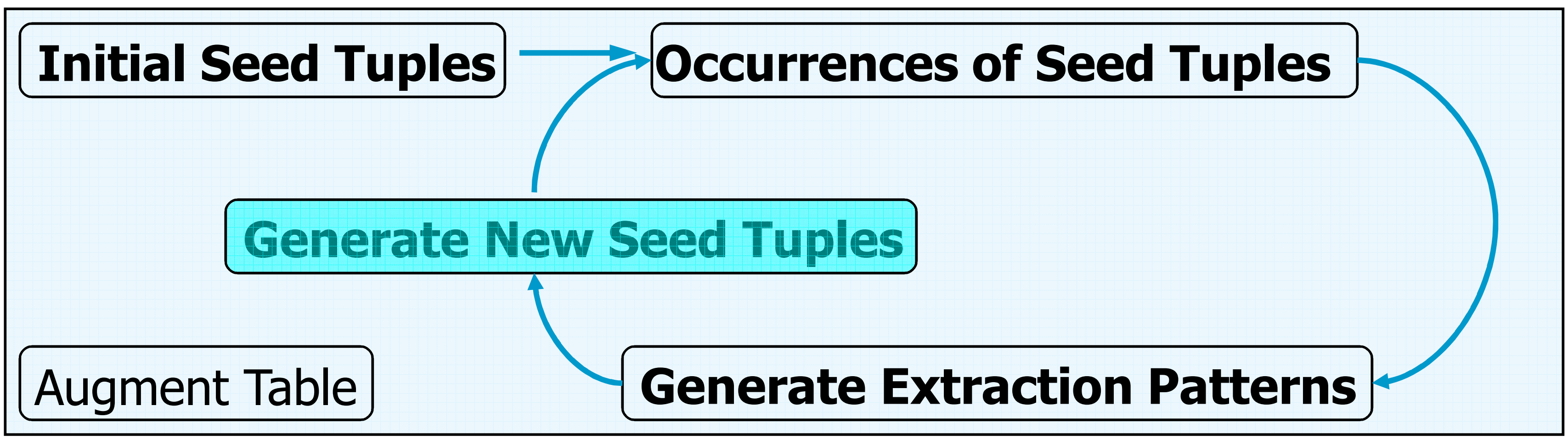


Slide from Eugene Agichtein

Semi-supervised (using Bootstrapping) Relation Extraction

Generate new seed tuples; start new iteration

ORGANIZATION	*LOCATION*
AG EDWARDS	ST LUIS
157TH STREET	MANHATTAN
7TH LEVEL	RICHARDSON
3COM CORP	SANTA CLARA
3DO	REDWOOD CITY
JELLIES	APPLE
MACWEEK	SAN FRANCISCO



Slide from Eugene Agichtein

Fact Extraction: Semi-supervised and Unsupervised

Systems differ in:

- Model of Representation

- Learning Algorithm and Incorporation of Domain Knowledge:
  - Bootstrapping → initial set of seeds grown iteratively, over multiple iterations
  - Distant supervision → a single iteration
  - Unsupervised → no seeds

Fact Extraction: Semi-supervised and Unsupervised

- Bootstrapping → initial set of seeds grown iteratively, over multiple iterations
- Distant supervision → a single iteration
- Unsupervised → no seeds

Bootstrapping: Example Systems

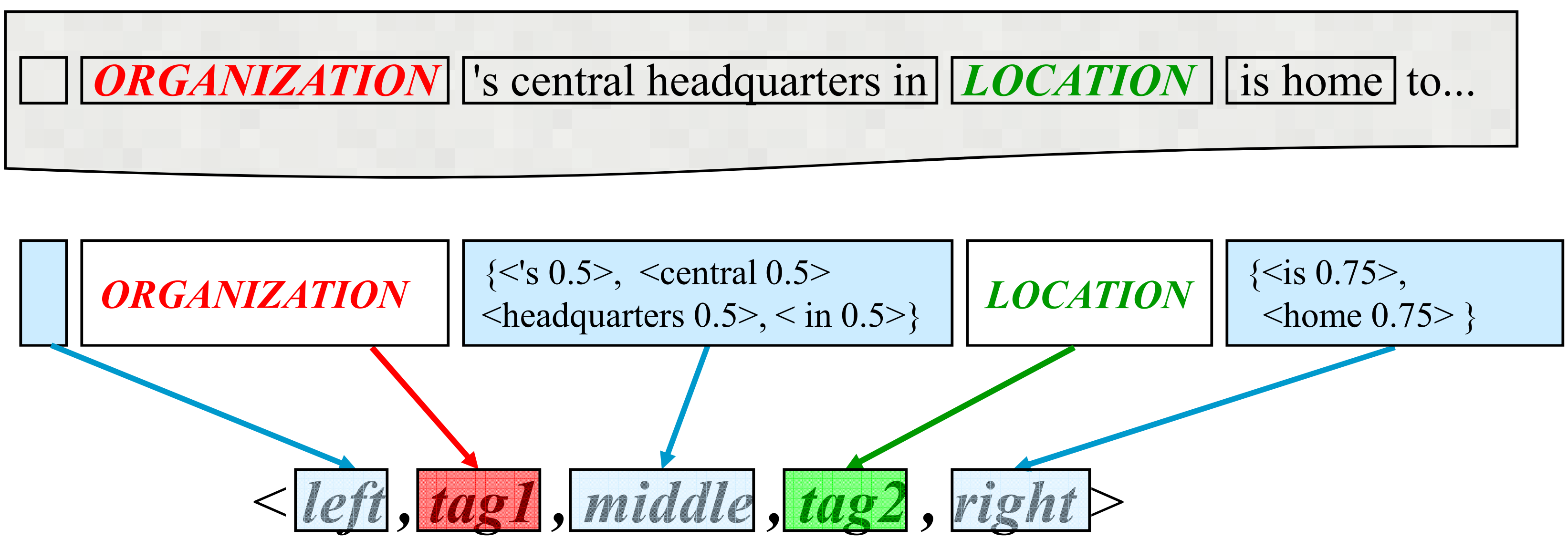
- AutoSlog-TS [Riloff, AAAI 1996]
- DIPRE [Brin, WebDB 1998]
- Snowball [Agichtein & Gravano, DL 2000]
- KnowItAll [Etzioni et al., J. AI 2005]
- KnowItNow [Cafarella et al., HLT 2005]
- Fact Extraction on the Web [Pasca et al., ACL 2006]
- Coupled Pattern Learning (part of NELL) [Carlson et al., WSDM 2010]
- [Gupta & Manning, ACL 2014]
- INSTAREAD [Hoffman et al., CoRR abs. 2015]
- ...

Bootstrapping: Example Systems

- AutoSlog-TS [Riloff, AAAI 1996]
- DIPRE [Brin, WebDB 1998]
- Snowball [Agichtein & Gravano, DL 2000]
- KnowItAll [Etzioni et al., J. AI 2005]
- KnowItNow [Cafarella et al., HLT 2005]
- Fact Extraction on the Web [Pasca et al., ACL 2006]
- Coupled Pattern Learning (part of NELL) [Carlson et al., WSDM 2010]
- [Gupta & Manning, ACL 2014]
- INSTAREAD [Hoffman et al., CoRR abs. 2015]
- ...

Snowball [Agichtein & Gravano, DL 2000]

- 5-tuple: <*left, tag1, middle, tag2, right*>,
  - *tag1, tag2* are named-entity tags (from a NER component)
  - *left*, *middle*, and *right* are vectors of weighed terms.



Snowball Pattern Generation

Occurrences of seed tuples converted to Pattern Representation.

The weight of each term is a function of the frequency of the term in the corresponding context.

Patterns clustered using a similarity metric

Patterns are formed as *centroids* of the clusters.

Cluster 1

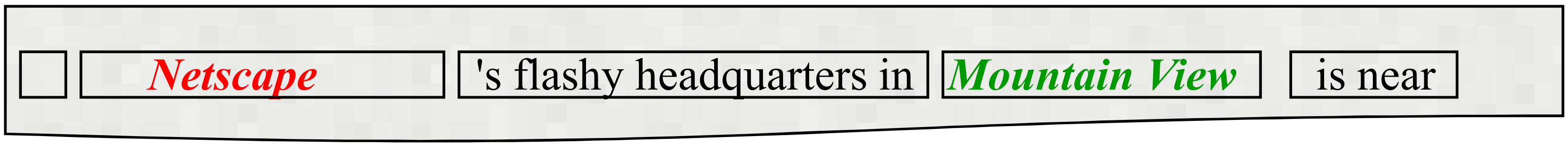
{<servers 0.75> <at 0.75>}	*ORGANIZATION*	{<'s 0.5> <central 0.5> <headquarters 0.5> <in 0.5>}	*LOCATION*	
{<operate 0.75> <from 0.75>}	*ORGANIZATION*	{<'s 0.7> <headquarters 0.7> <in 0.7>}	*LOCATION*	

Cluster 2

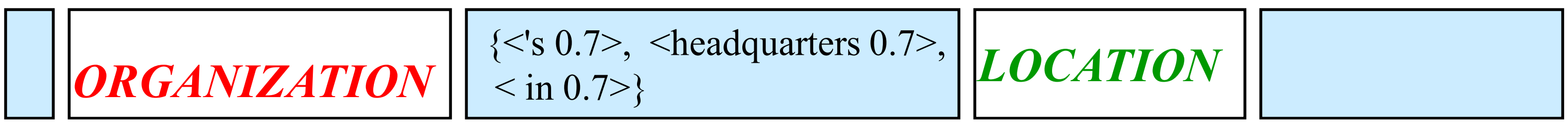
{<shares 0.75> <of 0.75>}	*LOCATION*	{<- 0.75> <based 0.75> }	*ORGANIZATION*	{<fell 1>}
{<the 1>}	*LOCATION*	{<- 0.75> <based 0.75> }	*ORGANIZATION*	{<introduced 0.75> <a 0.75>}

Snowball Tuple Extraction

- Represent each new text segment in the collection as a 5-tuple:

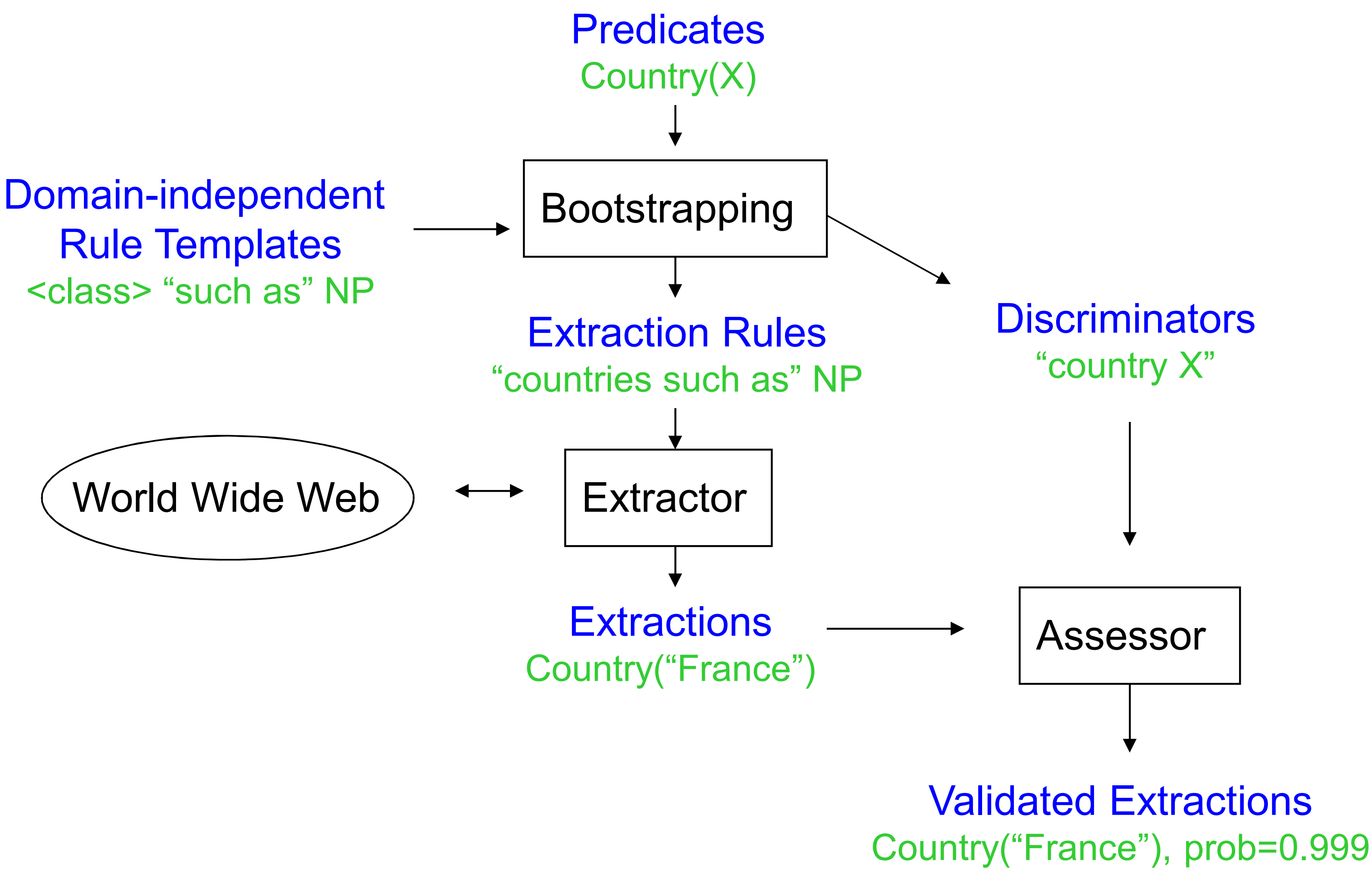


- Find most similar pattern (if any)



- Estimate correctness of extracted tuple:
  - A tuple has high confidence if generated by multiple high-confidence patterns
  - Conf (Pattern) = #positive /(#positive + # negative)
    - #positive: extracted tuples that agree on both Org and Loc attributes with a seed tuple from a previous iteration
    - #negative: extracted tuples with the same Org value with a seed tuple, but different Loc value (assumes Org is a key for the relation)

KnowItAll [Etzioni et al., J. AI 2005]



Slide from Dan Weld

KnowItAll Rules

Rule Template (domain-independent):

Predicate:	predName(Class1)
Pattern:	NP1 “such as” NPList2
Contraints:	head(NP1) = plural(label(Class1)
	properNoun(head(each(NPList2)))
Bindings:	instanceOf(Class1, head(each(NPList2)))

Extraction Rule (substituting “instanceOf” and “Country”)

Predicate:	instanceOf(Country)
Pattern:	NP1 “such as” NPList2
Contraints:	head(NP1) = “nations”
	properNoun(head(each(NPList2)))
Bindings:	instanceOf(Country, head(each(NPList2)))
Keywords:	“nations such as”

Sentence: *Other nations such as France, India and Pakistan, have conducted recent tests.*

Extractions:

instanceOf(Country, France), instanceOf(Country, India), instanceOf(Country, Pakistan)

KnowItAll Pattern Learning

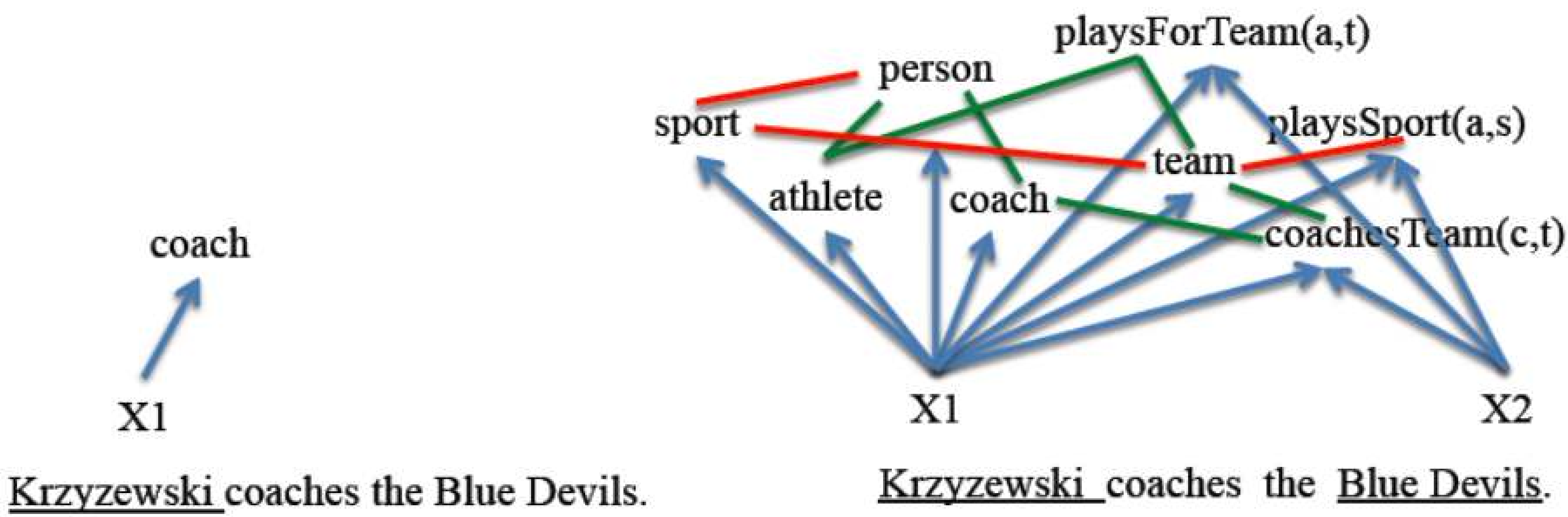
- **Goal:** supplement domain-independent patterns with domain-specific patterns

  "Headquarted in <city>"

- To increase recall (by learning extractors) and precision (by learning discriminators)

- Bootstrapping algorithm:
  - Start with seed instances generated by domain-independent extractors
  - For each seed, issue a Web search query and return the documents
  - For each occurrence in each document, form a context string by taking the *w* words to its left and right
  - Output the best patterns according to some metric. A pattern is any substring of the context string that includes the occurrence and at least one other word

Coupled Pattern Learning [Carlson et al., 2010]



hard (under constrained)
semi-supervised learning problem

easier (more constrained)
semi-supervised learning problem

Basic Idea: coupled training via multiple functions to avoid semantic drift
→ use the output of one classification function
to compare to another and vice versa

Coupled Pattern Learning [Carlson et al., 2010]

- Input: Ontology of entity and relation types; seed tuples
- Model of Representation: Sequence Patterns + Ranking Function
- Types of Coupled Constraints
  - Mutual exclusion
    - Mutually exclusive predicates cannot both be satisfied by the same input
  - Argument type-checking
    - E.g., arguments of CompanyIsInEconomicSector relation have to be of type Company and EconomicSector
- Coupled Pattern Learning:
  1. Generate patterns (for both entity and relation)
  2. Extract candidate tuples
  3. (New) Filter tuples based on constraints
  4. Rank patterns and tuples; decide which to promote
  5. Repeat
- Part of the NELL system [Mitchell et al., AAAI 2015]

Iterative Feedback in NELL [Mitchell et al., AAAI 2015]

serena_williams (female)
literal strings: Serena Williams, serena williams, serena-williams

Help NELL Learn!

NELL wants to know if these beliefs are correct.
If they are or ever were, click thumbs-up. Otherwise, click thumbs-down.

- serena_williams is a female
- serena_williams is a Canadian person
- serena_williams is an athlete
- serena_williams is an athlete who beat venus_williams (athlete)
- serena_williams is an athlete who wins open (awardtrophytournament)
- serena_williams is an athlete who wins australian_open (awardtrophytournament)
- serena_williams is an athlete who wins french_open (awardtrophytournament)

User feedback incorporated in next iterations of learning

categories

- **female**(100.0%)
  - Seed
  - CPL @824 (65.5%) on 20-mar-2014 ["_ have clinched at" "finals loss to _" "several women including _" "tennis stars like _" "she was runner-up to _" "_ is the only American woman" "_ 's Strokes" "_ become Olympic champions" "_ is top seed" "_ becomes the first African-American woman" "_ played doubles" "Venus Williams beat _" "_ wins the women" "_ defeated Daniela Hantuchova" "_ beat Venus Williams" "_ made a fashion statement" "Venus Williams defeated _" "_ defeated pair" "Dementieva beats _" "_ ignored pain" "female athletes like _" "_ defeated Jelena Jankovic" "match point against _" "_ getting broody" "_ ' tennis coach" "_ Looks Hot" "_ took the Gold Medal" "_ won a Grand Slam"] using serena_williams
  - SEAL @165 (100.0%) on 14-nov-2010 [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23] using serena_williams

Model-level Provenance

Transparent ML in Bootstrapping Systems

- Transparency in Model of Representation
  - Sequence Patterns + Ranking function
  - Partial Model-level Provenance: Extracted objects explained by the supporting patterns
    - Snowball: Term weights make patterns more difficult to comprehend, loosing some transparency
    - Cannot typically explain why the extracted object is above the ranking threshold

- Transparency in Learning Algorithm
  - Algorithm-level Provenance in KnowItAll and CPL
    - **Learning of each pattern** can be explained by the **supporting tuples**
    - **Extraction of each tuple** can be explained by the **supporting patterns**
  - Snowball → more diffused provenance because patters are centroids of clusters, hence explainable by support tuples of all patterns in the cluster
  - KnowItAll: some transparency in influencing the model based on initial keywords
  - SPIED-Viz [Gupta & Manning 2014] → Visually explain patterns/tuples (see Part 4)

- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline (Snowball, KnowItAll) or Interactive (CPL)
  - Possible to incorporate DK at deployment (by reviewing the patterns)
    - CPL→ crowdsourced review of tuples for continuous learning

INSTAREAD [Hoffmann et al., CoRR abs. 2015]

- **Model of Representation:** Prolog-like predicate-based rules

killNoun('murder');

killOfVictim(c, b) ⇐ prep-of(c, b) ∧ token(c, d) ∧ killNoun(d);

killed(a, b) ⇐ person(a) ∧ person(b) ∧ nsubjpass(c, a)

∧ token(c, 'sentenced') ∧ prep-for(c, d) ∧ killOfVictim(d, b);

Mr. Williams was sentenced for the murder of Wright.

killOfVictim(murder, Wright), killed(Williams, Wright)

- Support for disjunction (∨), negation (−), existential (∃) and universal (∀) quantification
- Rich set of predicates:
  - Built-in: tokenBefore, isCapitalized, ...
  - Output of other NLP systems: Phrase structure, Typed dependencies parser, Co-reference resolution, Named entities

INSTAREAD [Hoffmann et al., CoRR abs. 2015]

Semi-automatic rule generation with user in the loop

1. **Core Linguistic Rules:** Prepopulate the system with syntactic lexical patterns
   – Given subject X, object Y and verb 'kill', generate rules to capture 'X killed Y', 'Y was killed by X',…

2. **Bootstrapped Rule induction:** Use results of existing rules to generate seed tuples to automatically generate ranked list of new rules
   – Two ranking criteria: PMI and number of extractions
   – Allow the user to manually inspect the rules and select the rules

3. **Word-level distributional similarity:** Given seed keyword, automatically suggest similar keywords
   – Generate new rules based on user keyword selection

Transparent ML in INSTAREAD

- Transparency in Model of Representation
  - Predicate-based rules, declarative
  - Model-level Provenance
- Transparency in Learning Algorithm
  - Transparency in terms of user influencing the model by selecting rules
  - User-friendly visual interface (see Part 4)
- Transparency in Incorporation of Domain Knowledge (DK)
  - Interactive: User can modify/remove a generated rule, or define a new rule, e.g., based on suggested keywords
  - Easy to incorporate DK at deployment (by further modifying the rules)

Fact Extraction: Semi-supervised and Unsupervised

- Bootstrapping → initial set of seeds grown iteratively, over multiple iterations
- Distant supervision → a single iteration
- Unsupervised → no seeds

Fact Extraction: Distant Supervision

- General Framework
  1. Construct training set of seed tuples
  2. Distant supervision: generalize training set into extraction patterns
  3. Execute patterns
  4. Score extracted tuples

- Example systems:
  - OLLIE [Mausam et al., EMNLP 2012]
  - RENOUN [Yahya et al. EMNLP 2014]

OLLIE [Mausam et al., EMNLP 2012]

- Input: Seed triplets <arg1, {rel}, arg2>

- Model of Representation: Path Patterns + Classifier
  - Patterns centered around verbs, nouns, adjectives, etc.

- Pattern Learning: Generalize from sentences that are "paraphrases" of seed tuples

- Classifier (factual vs. non-factual):
  - Context analysis (dependency-based): to discard invalid facts, e.g., conditional, or attributed to someone else
  - Logistic regression classifier to identify other likely non-factual tuples
    - Trained on manually labeled triples extracted from 1000 sentences

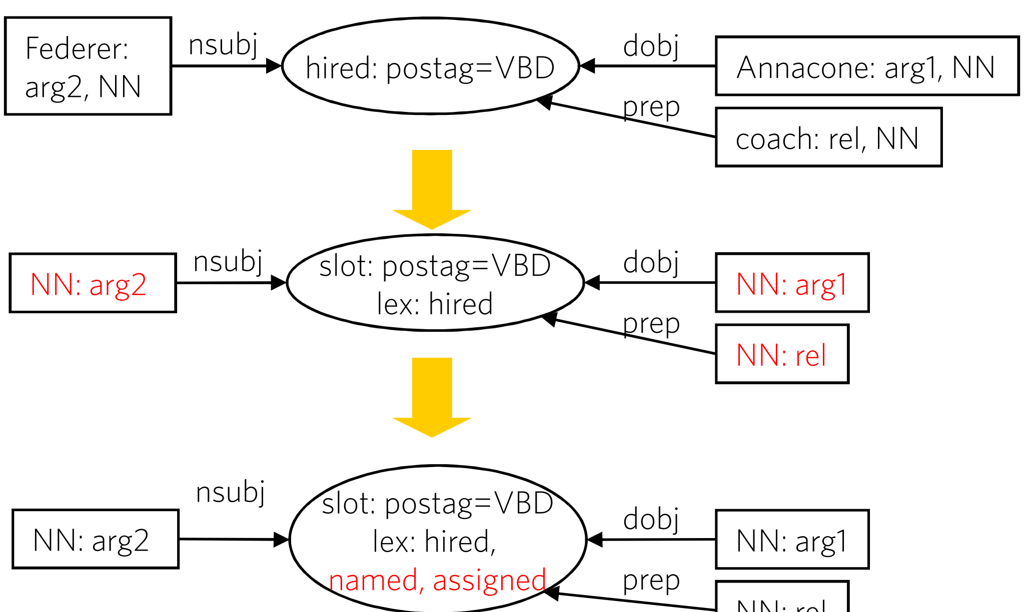
OLLIE Pattern Learning

(Annacone; is the coach of; Federer)

Seed tuple

Federer hired Annacone as coach

“Paraphrase” of seed tuple→ sentence contains content words linked by a linear dependency path



Dependency Parse

Delexicalize relation nodes

Retain lexical constraints on slot nodes, and generalize based on seed sentences where the fully delexicalized pattern was seen

RENOUN [Yahya et al., EMNLP 2014]

- Focus on facts centered around noun phrases:

  'The CEO of Google, Larry Page'   Google → CEO (Attribute) → Larry Page

- Model of Representation: Path Patterns + Ranking function
- Input: Ontology of nominal attributes (e.g., Biperdia)

  8 manually crafted high-precision patterns to find seed tuples in corpus

- Pattern Learning: Generalize from seed tuples
- Fact Scoring: Score(t) = **Σ** frequency(p_i) x coherence(p_i), for all patterns p_i that support t
  - A pattern has high coherence if it applies to attributes that are similar as per their word vectors
  - Rank facts by the score, and consider top-*K*, where *K* is set by the user

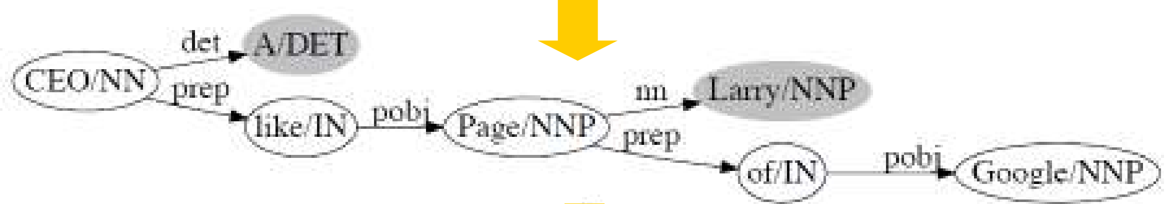
RENOUN Pattern Learning

Google → CEO (Attribute) → Larry Page

Seed tuple (Biperdia + 8 patterns)

A CEO, like Larry Page of Google is…

“Paraphrase” of seed tuple → contains Attribute of the seed, with Subject and Object as in seed



Dependency Parse

Minimal subgraph containing head tokens of S, A, O

Delexicalize the S, A, O nodes; lift noun POS tags to N; Discard patterns supported by less than 10 seed tuples

Transparent ML in Distant Supervision Systems

- Transparency in Model of Representation
  - Path Patterns + Classifier/Ranking function
  - Model-level provenance (partial)
    - Extracted objects explained by the supporting patterns
    - Ranking function (RENOUN) typically easier to understand than a logistic regression classifier (OLLIE)
    - OLLIE → dependency-based context analysis portion of the classifier is transparent

- Transparency in Learning Algorithm
  - Algorithm-level Provenance: **Learning of each pattern** can be explained by the **supporting tuples**
  - RENOUN → some additional transparency in terms of user influencing the model via the threshold K

- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline
  - Possible to incorporate DK at deployment (by reviewing the patterns)

Fact Extraction: Semi-supervised and Unsupervised

- Bootstrapping → initial set of seeds grown iteratively, over multiple iterations
- Distant supervision → a single iteration
- Unsupervised → no seeds

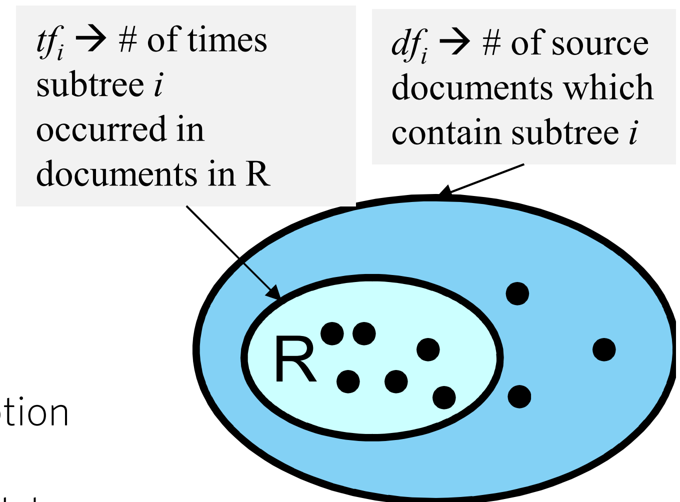
Fact Extraction: Unsupervised

- Traditional IE: [Sudo et al., ACL 2003]

- Open IE: REVERB [Fader et al., EMNLP 2011]

An Improved Extraction Pattern Representation Model for Automatic IE Pattern Acquisition [Sudo et al., ACL 2003]

- **Scope:** Traditional IE, w/ extraction task specified by TREC-like narrative description
- **Preprocessing:** Dependency Analysis, NE-tagging
- **Model:** Path patterns



- Learning Algorithm
  1. Retrieve relevant documents R
     - Issue search query using sentences from narrative description
  2. Count all possible subtrees in R
     - Make a Pattern List of those that conform the pattern model
  3. Rank each subtree (inspired by TF/IDF):

$$score_i = tf_i \cdot \left(\log \left(\frac{N}{df_i} \right) \right)^{\beta}$$

     - β trained to prioritize among overlapping patterns, preferring more specific patterns

REVERB [Fader et al., EMNLP 2011]

- **Scope:** Open IE of relations centered around verbs
- **Preprocessing:** POS tagging, NP chunking
- **Model:** Fixed syntactic pattern + classifier
- **Pattern:** <NP1> … < VP> … <NP2>
  - <VP> satisfies:
    - Syntactic constraint: V|VP|VW*P → to allow light-verb constructions (e.g., "give a talk at')
    - Lexical constraint → to avoid over-specified relations
      - Based on large dictionary of generic relation phrases, automatically discovered from 500M Web pages
    - Adjacent/overlapping VPs are merged into a single VP
  - <NP1> and <NP2> are the noun phrases closest to <VP> to the left/right
    - Exclude relative pronoun, who-adverb and existential "there"
- **Learning Algorithm:**
  - Find all matches for the syntactic pattern
  - Use logistic regression to assign a confidence to each extracted triple
    - Classifier trained manually labeled extracted triples from 1000 sentences
  - Trade precision for recall using a confidence threshold

Transparent ML in Unsupervised Fact Extraction

- Transparency in Model of Representation
  - Sequence/Path Patterns + Classifier/Ranking function
  - Model-level Provenance (partial)
    - Extracted objects explained by the supporting patterns
    - Ranking function ([Sudo 2013]) typically easier to understand compared to a logistic regression classifier (REVERB)

- Transparency in Learning Algorithm
  - No transparency

- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline
  - Can incorporate DK at deployment, by reviewing the patterns (not for REVERB)

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

RIPPER [Cohen, ICML 1995]

- Classic propositional rule learner algorithm that:
  - Performs efficiently on large noisy data
  - Extends naturally to first order logic representations
  - Competitive in generalization performance
- Input: positive and negative examples
- Algorithm (sketch)
  1. Building stage: Repeat until <stopping condition>
     1. Split examples into two sets: Grow and Prune
     2. Grow one rule by greedily adding conditions until the rule is 100% precise on Grow set
     3. Incrementally prune each rule based on Prune set → to avoid overfitting
  2. Optimization stage: Simplify ruleset by deleting rules in order to reduce total description length
- Useful for learning Predicate-based rules for IE, e.g. rule induction [Nagesh et al., 2012]
- Extensions: e.g., SLIPPER [Cohen & Singer 1999]

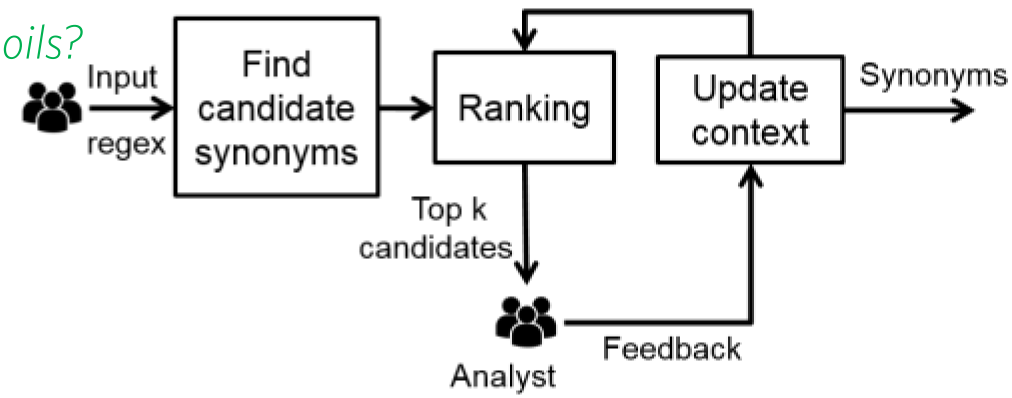
CHIMERA [Suganthan et al., SIGMOD 2015]

Rule generation for product classification: *(motor | engine) oils?* → *motor oil*

1. Tool to increase the recall of a single classification rule

Automotive
Truck
ATV
...

(motor | \syn) oils?



- *Rank candidate synonyms based on context similarity with known synonyms*
- *User feedback on some candidates → re-rank remaining candidates*

CHIMERA [Suganthan et al., SIGMOD 2015]

Rule generation for product classification: *(motor | engine) oils?* → *motor oil*

2. Tool to generate classification rules from examples
   - Sequence mining to generate candidate rules from labeled product titles
   - Greedy algorithm to select a subset of rules that provide good coverage and high precision

Transparent ML in Learning of Classification Rules

- Transparency in Model of Representation
  - Classification rules
  - Model-level Provenance

- Transparency in Learning Algorithm
  - RIPPER → No transparency
  - CHIMERA → transparency in terms of the user influencing the learning via (1) the initial rule and (2) selection of candidate synonyms

- Transparency in Incorporation of Domain Knowledge (DK)
  - Offline (RIPPER), or interactive (CHIMERA)
  - Possible to incorporate DK at deployment (by modifying the rules)

Transparent ML Techniques

	Unsupervised	Semi-supervised	Supervised
Dictionary			
Regex			
Rules			
Rules + Classifier			
Classification Rules			

Recap

- Transparency in Model
  - Model-level provenance available in most surveyed systems, with some exceptions: imperative language (FlashExtract), complex rules w/ weights (Snowball), using a CRF (AutoSlog-SE)

- Transparency in Learning Algorithm
  - Algorithm-level provenance available in a few systems, to various extents
  - User ability to influence the model → a variety of ways

- Transparency in Incorporation of Domain Knowledge
  - Interactive → few systems: WHISK, INSTAREAD, CHIMERA
  - Deployment → mostly depends on model-level provenance

Transparent ML: Building an End-to-end Transparent IE System

Outline

- Building a Transparent IE System
- Transparent Machine Learning
- Building Developer Tools around Transparent IE
- Case Study and Demo

Background: The SystemT Project

- Early 2000's: NLP group starts at IBM Research – Almaden
- Initial focus: Collection-level machine learning problems
- Observation: Most time spent on feature extraction
  - Technology used: Cascading finite state automata

Problems with Cascading Automata

- Scalability
- Expressivity
- Ease of comprehension
- Ease of debugging
- Ease of enhancement

Transparency

Lack of Transparency in Cascading Automata

IBM

Rule priority used to prefer First over Caps

Rule priority used to prefer First over Caps.

First preferred over Last since it was declared earlier

Lorem ipsum dolor sit amet, consect d g n tis facilisis, volutpat dapibus, ultrices sit amet, sem , volutpat dapibus, ultrices sit amet, sem **Tomorrow, we will meet Mark Scott, Howard Smith and** amet lt arcu tincidunt orci. Pellentesque justo tellus , scelerisque quis, facilisis nunc volutpat enim, quis viverra lacus nulla sit lect a sus tincidunt orci. Pellentesque justo tellus , scelerisque quis, facilisis quis, interdum non, ante. Suspendisse

Level 1

⟨Gazetteer⟩[type = FirstGaz] → ⟨First⟩

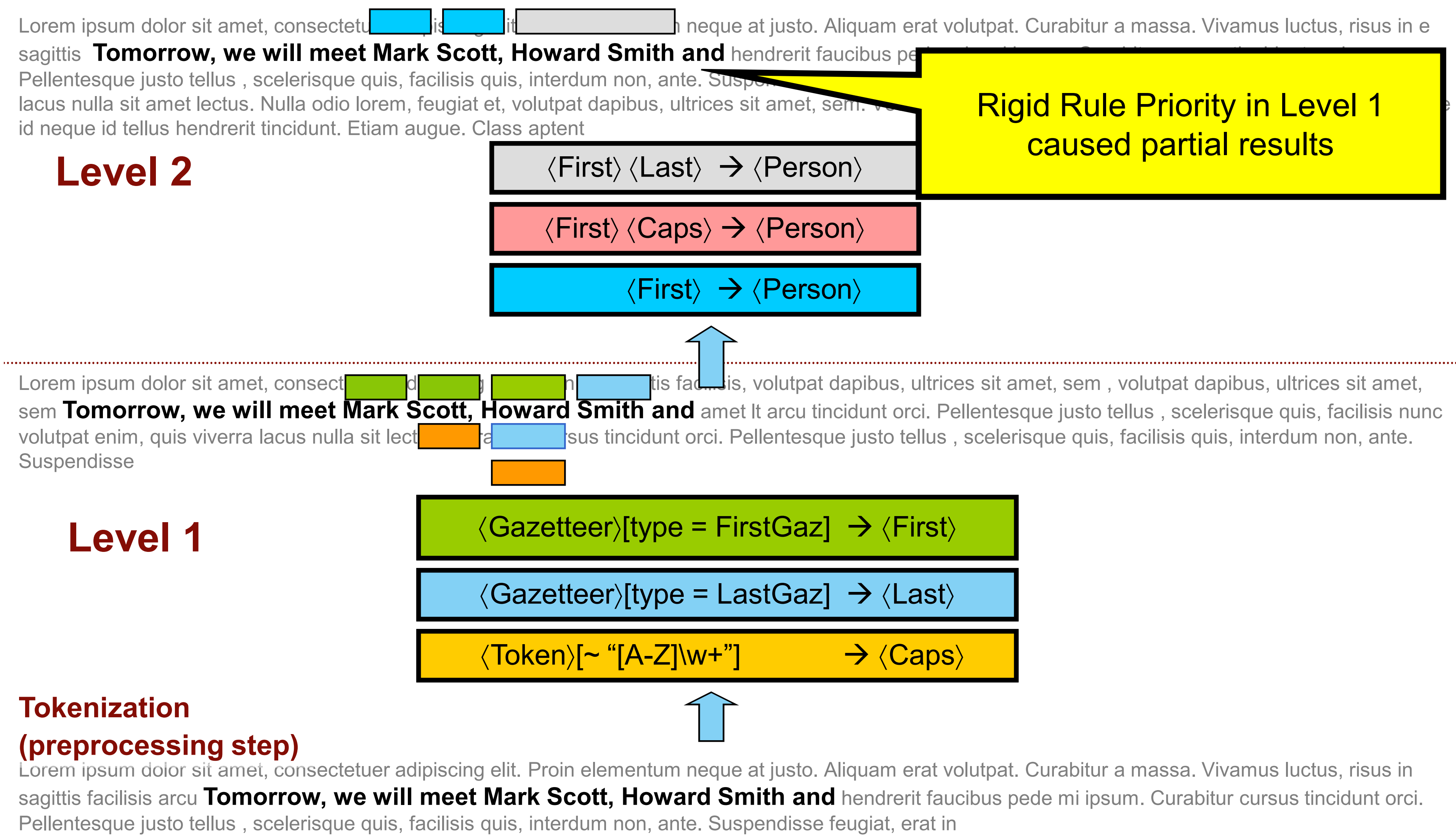
⟨Gazetteer⟩[type = LastGaz] → ⟨Last⟩

⟨Token⟩[~ "[A-Z]\w+"] → ⟨Caps⟩

Tokenization (preprocessing step)

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Proin elementum neque at justo. Aliquam erat volutpat. Curabitur a massa. Vivamus luctus, risus in sagittis facilisis arcu **Tomorrow, we will meet Mark Scott, Howard Smith and** hendrerit faucibus pede mi ipsum. Curabitur cursus tincidunt orci. Pellentesque justo tellus , scelerisque quis, facilisis quis, interdum non, ante. Suspendisse feugiat, erat in

Lack of Transparency in Cascading Automata



Problems with Cascading Automata

- Scalability: Redundant passes over document
- Expressivity: Frequent use of custom code
- Ease of comprehension
- Ease of debugging
- Ease of enhancement

Operational semantics
+ custom code
= no provenance

Outline

- Building a Transparent IE System
- Transparent Machine Learning
- Building Developer Tools around Transparent IE
- Case Study and Demo

Bringing Transparency to Feature Extraction

- Our approach: Use a declarative language
  - Decouple meaning of extraction rules from execution plan
- Our language: AQL (Annotator Query Language)
  - Semantics based on relational calculus
  - Syntax based on SQL

AQL Data Model (Simplified)

Document
text: String

Person		
first: Span	*last*: Span	*fullname*: Span

- Relational data model: data is organized in *tuples*; tuples have a *schema*
- Special data types necessary for text processing:
  - Document consists of a single ***text*** attribute
  - Annotations are represented by a type called Span, which consists of ***begin***, ***end*** and ***document*** attribute

AQL By Example


```
create view FirstCaps as
select CombineSpans(F.name, C.name) as name
from First F, Caps C
where FollowsTok(F.name, C.name, 0, 0);
```

- Declarative: Specify logical conditions that input tuples should satisfy in order to generate an output tuple
- Choice of SQL-like syntax for AQL motivated by wider adoption of SQL
- Compiles into SystemT algebra

Revisiting the Person Example

<First><Caps>

<First><Last>

<First>

```
create view Person as
select S.name as name
from (
     ( select CombineSpans(F.name, C.name) as name
       from First F, Caps C
       where FollowsTok(F.name, C.name, 0, 0))
   union all
     ( select CombineSpans(F.name, L.name) as name
       from First F, Last L
       where FollowsTok(F.name, L.name, 0, 0))
   union all
     ( select *
       from First F )
) S
consolidate on name;
```

Revisiting the Person Example

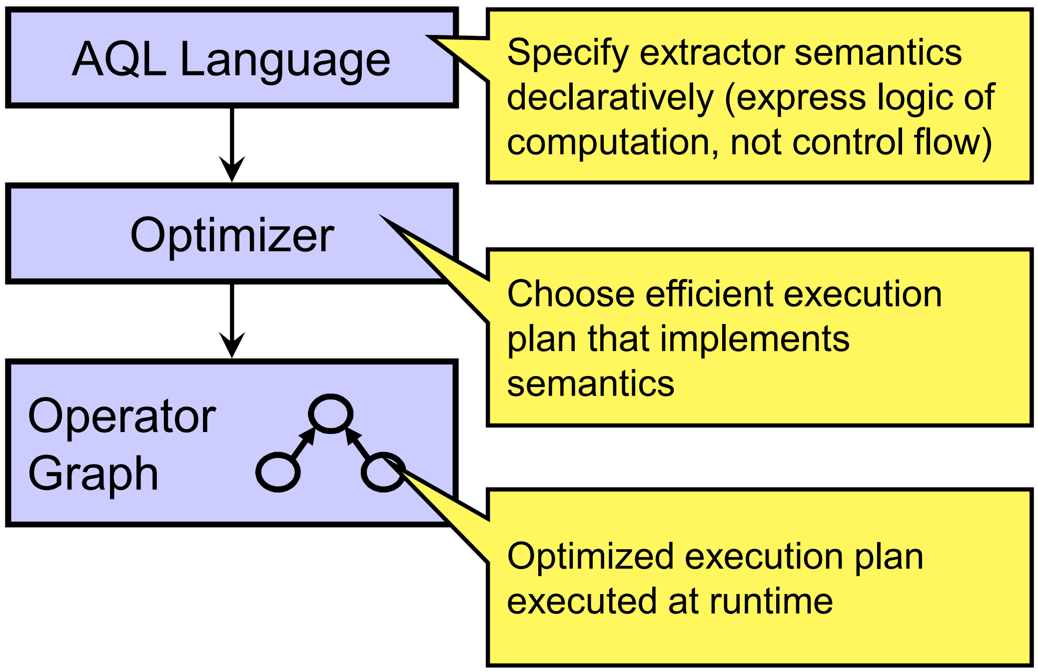
Input may contain overlapping annotations

(No Lossy Sequencing problem)

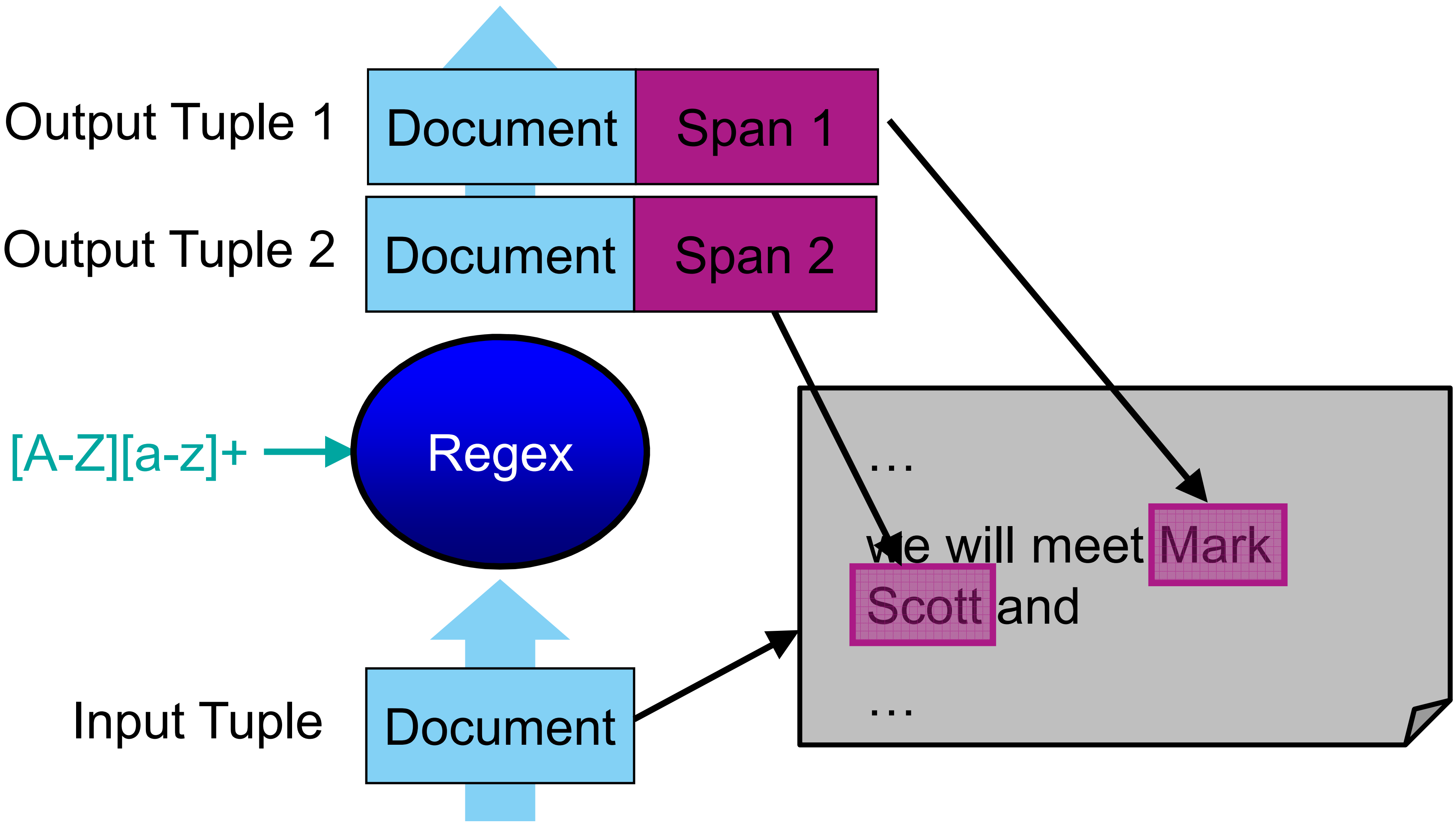
```
create view Person as
select S.name as name
from (
       ( select CombineSpans(F.name, C.name) as name
         from First F, Caps C
         where FollowsTok(F.name, C.name, 0, 0))
    union all
       ( select CombineSpans(F.name, L.name) as name
         from First F, Last L
         where FollowsTok(F.name, L.name, 0, 0))
    union all
       ( select *
         from First F )
) S
consolidate on name;
```

Explicit clause for resolving ambiguity

Compiling and Executing AQL



Regular Expression Extraction Operator



How AQL Solved our Problems

- Scalability: *Cost-based query optimization*
- Expressivity: *Complex tasks, no custom code*
- **Ease of comprehension**
- **Ease of debugging**
- **Ease of enhancement**

Clear and Simple Provenance

Computing Model-level Provenance

- (Model-level) Provenance: Explains output data in terms of the input data, the intermediate data, and the transformation (e.g., SQL query, ETL, workflow)
  - Surveys: [Davidson & Freire, SIGMOD 2008] [Cheney et al., Found. Databases 2009]
- For predicate-based rule languages (e.g., SQL), can be computed automatically!

PersonPhone rule:

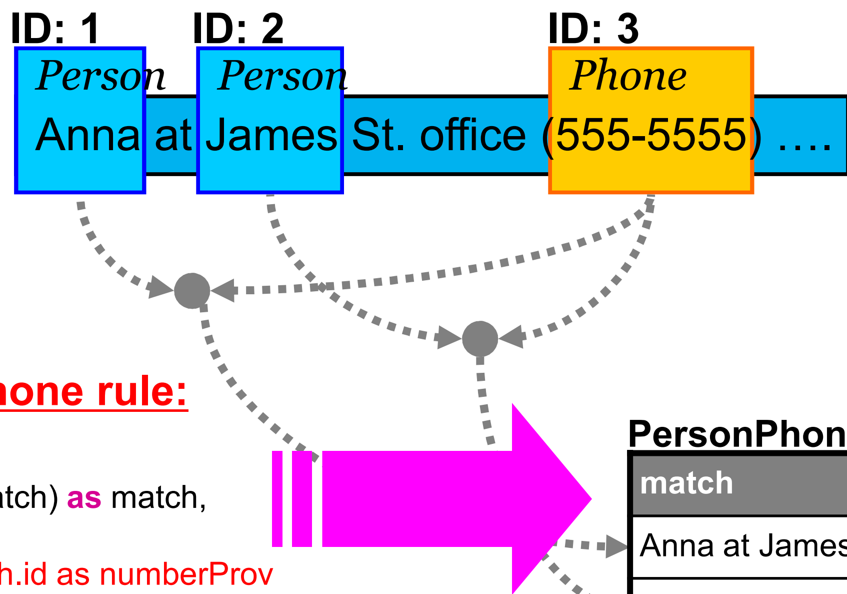
```
insert into PersonPhone
select Merge(F.match, P.match) as match
from    Person F, Phone P
where Follows(F.match, P.match, 0, 60);
```

PersonPhone

match
Anna at James St. office (555-5555
James St. office (555-5555

Computing Model-level Provenance

- (Model-level) Provenance: Explains output data in terms of the input data, the intermediate data, and the transformation (e.g., SQL query, ETL, workflow)
  - Surveys: [Davidson & Freire, SIGMOD 2008] [Cheney et al., Found. Databases 2009]
- For predicate-based rule languages (e.g., SQL), can be computed automatically!



Rewritten PersonPhone rule:

```
insert into PersonPhone
select Merge(F.match, P.match) as match,
         GenerateID() as ID,
         P.id as nameProv, Ph.id as numberProv
         'AND' as how
from     Person F, Phone P
where Follows(F.match, P.match, 0, 60);
```

PersonPhone

match	Provenance
Anna at James St. office (555-5555	1 AND 3
James St. office (555-5555	2 AND 3

AQL: Going beyond feature extraction

Extraction Task: Named-entity extraction
Systems compared: SystemT (customized) vs. [Florian et al.'03] [Minkov et al.'05]

Dataset	Entity Type	System	Precision	Recall	F-measure
CoNLL 2003	Location	SystemT	93.11	91.61	92.35
		Florian	90.59	91.73	91.15
	Organization	SystemT	92.25	85.31	88.65
		Florian	85.93	83.44	84.67
	Person	SystemT	96.32	92.39	94.32
		Florian	92.49	95.24	93.85
Enron	Person	SystemT	87.27	81.82	84.46
		Minkov	81.1	74.9	77.9

Transparency without machine learning outperforms machine learning without transparency.

[Chiticariu et al., EMNLP'10]

Outline

- Building a Transparent IE System
- Transparent Machine Learning
- Building Developer Tools around Transparent IE
- Case Study and Demo

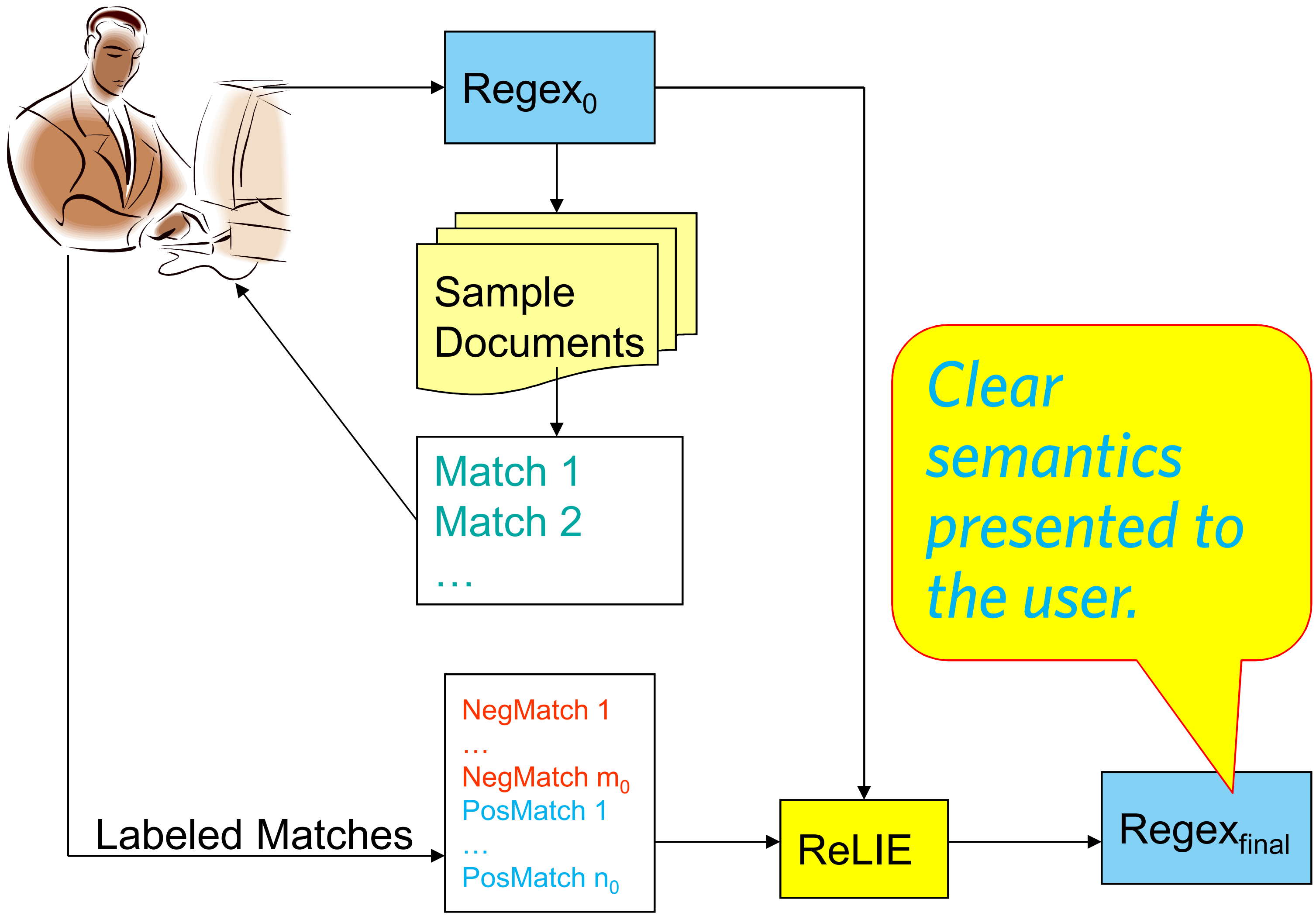
Machine Learning in SystemT

- AQL provides a foundation of transparency
- Next step: Add machine learning without losing transparency
- Major machine learning efforts:
  - Low-level features
  - Rule refinement
  - Rule induction
  - Normalization
  - Embedded Models

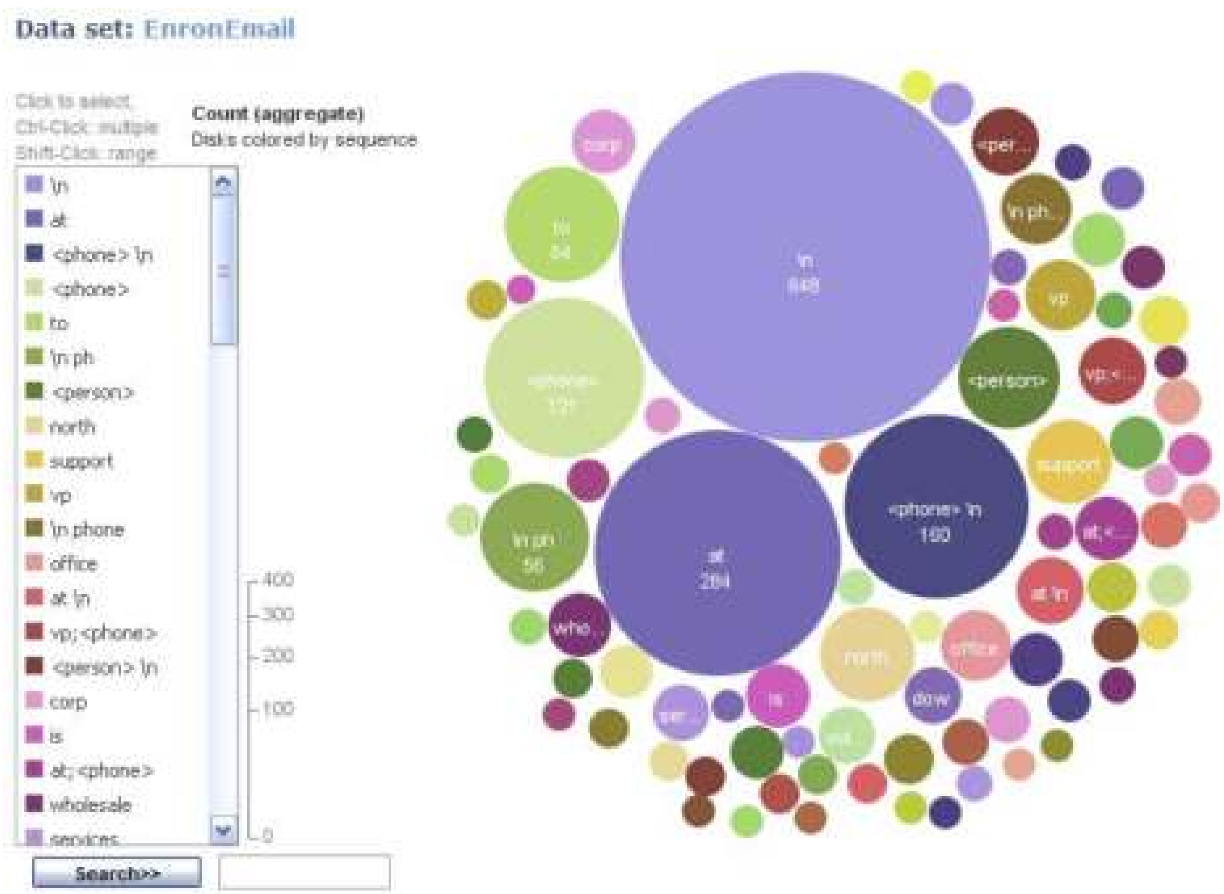
Machine Learning in SystemT

- Low-level features
- Rule refinement
- Rule induction
- Normalization
- Embedded Models

Recap from Part 3: Regular Expression learning with ReLIE [Li et al., EMNLP 2008]



Recap from Part 3: Pattern discovery for dictionaries [Li et al., CIKM 2011]



Machine Learning in SystemT

- Low-level features
- Rule refinement
- Rule induction
- Normalization
- Embedded Models

Recap: Rule Refinement [Liu et al. VLDB 2010]

R1: **create view** Phone **as**
  Regex(‘d{3}-\d{4}’ , Document, text);

R2: **create view** Person **as**
  Dictionary(‘first_names.dict’ , Document, text);

  Dictionary file *first_names.dict*:
  anna, james, john, peter…

R3: **create table** PersonPhone(match *span*);

  insert into PersonPhone
  select Merge(F.match, P.match) **as** match
  from Person F, Phone P
  where Follows(F.match, P.match, 0, 60);

- Rules expressed in SQL
  - Select, Project, Join, Union all, Except all
  - Text-specific extensions
    - Regex, Dictionary table functions
    - New selection/join predicates
  - Can express core functionality of IE rule languages
    - AQL, CPSL, XLog
- Relational data model
  - Tuples and views
  - New data type *span*: region of text in a document

Document:

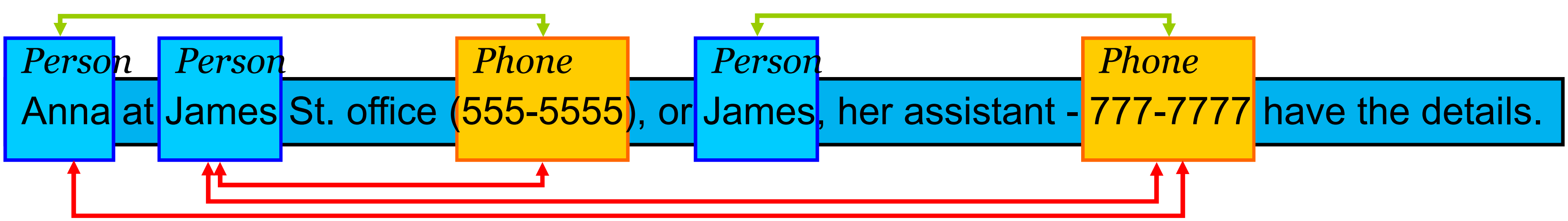
text
Anna at James St. office (555-5555), or James, her assistant – 777-7777 have the details.

Phone:

match
555-5555
777-7777

Person:

match
Anna
James
James



Method Overview [Liu et al. VLDB 2010]

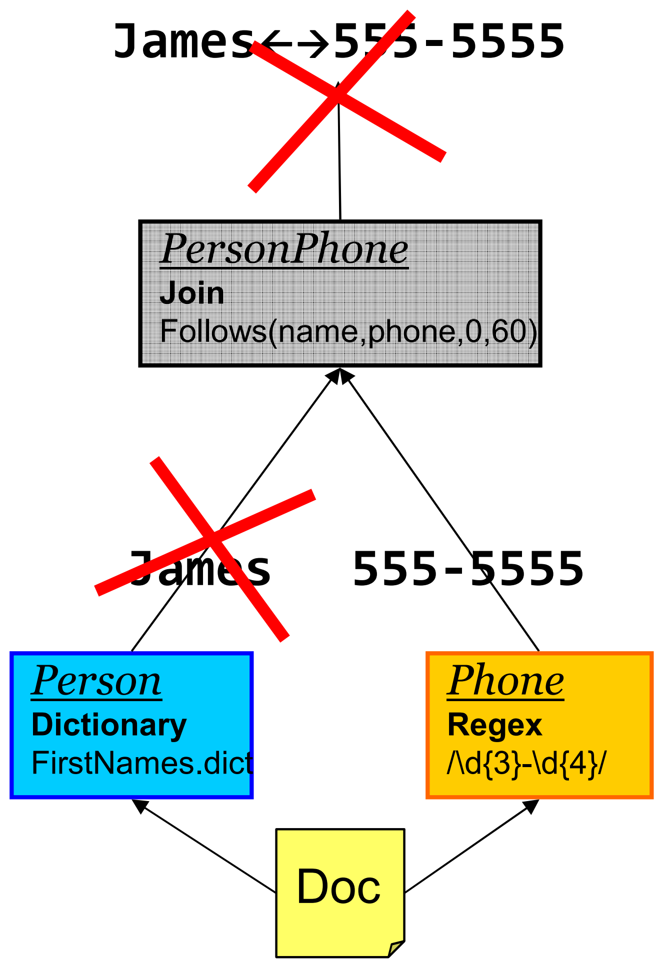
- Framework for systematic exploration of multiple refinements geared towards improving precision
- Input: Extractor P

  Labeled results in the output of P
- Goal: Generate refinements of P that remove false positives, while not affecting true positives
- Basic Idea:

  Cut any provenance link → wrong output disappears

Provenance (transparency) enables automatic rule refinement.

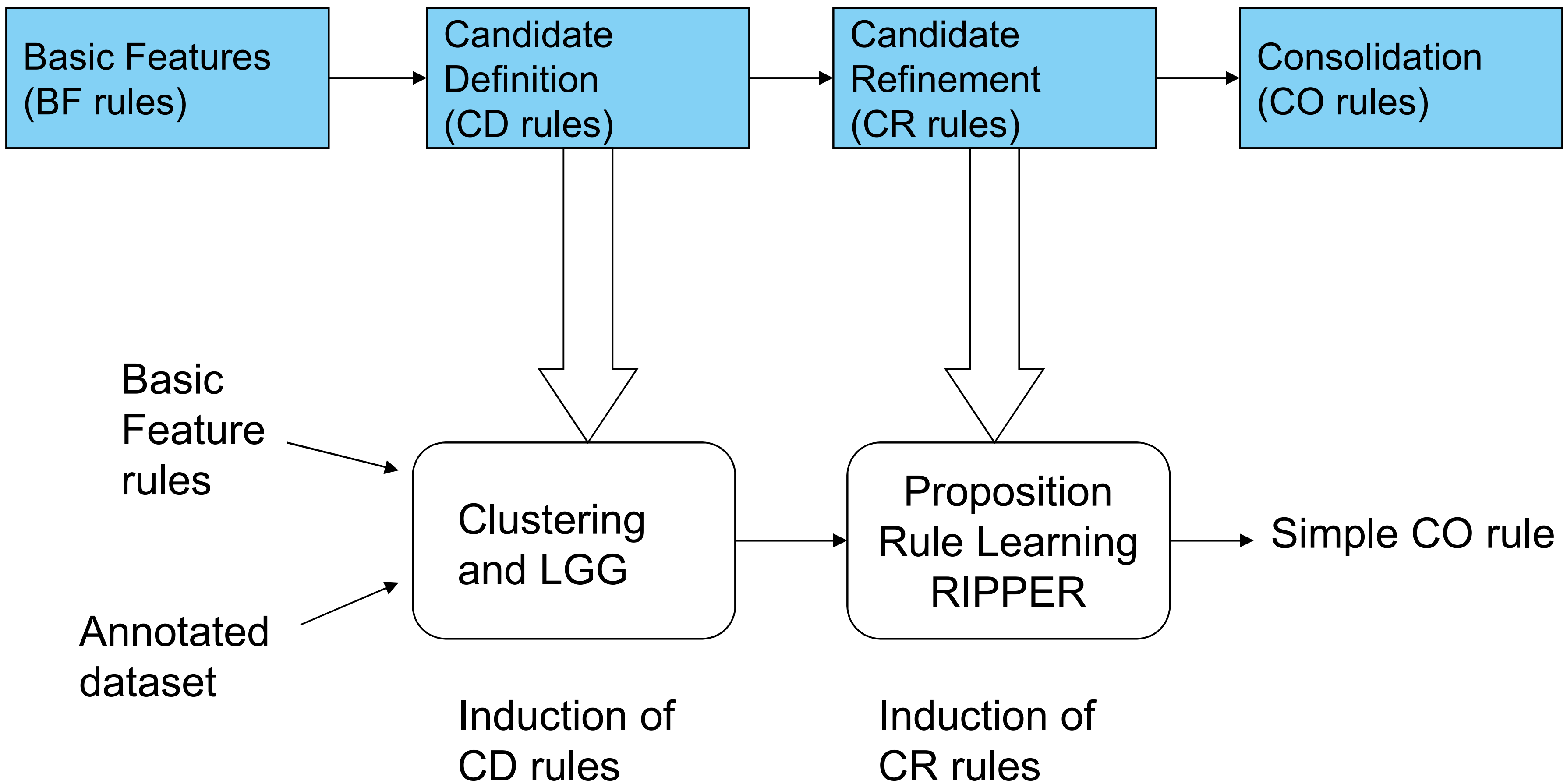
(Simplified) provenance of a wrong output



Machine Learning in SystemT

- Low-level features
- Rule refinement
- Rule induction
- Normalization
- Embedded Models

Recap from Part 3: Rule Induction [Nagesh et al., EMNLP 2012]



Recap: Least general generalisation (LGG) of annotations

PER: **john Smith** `person(X,D1) :- startsWith(X, X1), FirstNameDict(X1), endsWith(X, X2), immBefore(X1,X2), Caps(X2).`

PER: **John Doe** `person(Y,D2) :- startsWith(Y, Y1), FirstNameDict(Y1), Caps(Y1), endsWith(Y, Y2), immBefore(Y1,Y2), Caps(Y2).`

Prolog representation of declarative AQL

LGG of the above

```
person(Z,D) :- startsWith(Z, Z1), FirstNameDict(Z1),
               endsWith(Z, Z2), immBefore(Z1,Z2), Caps(Z2)
```

Machine Learning in SystemT

- Low-level features
- Rule refinement
- Rule induction
- Normalization
- Embedded Models

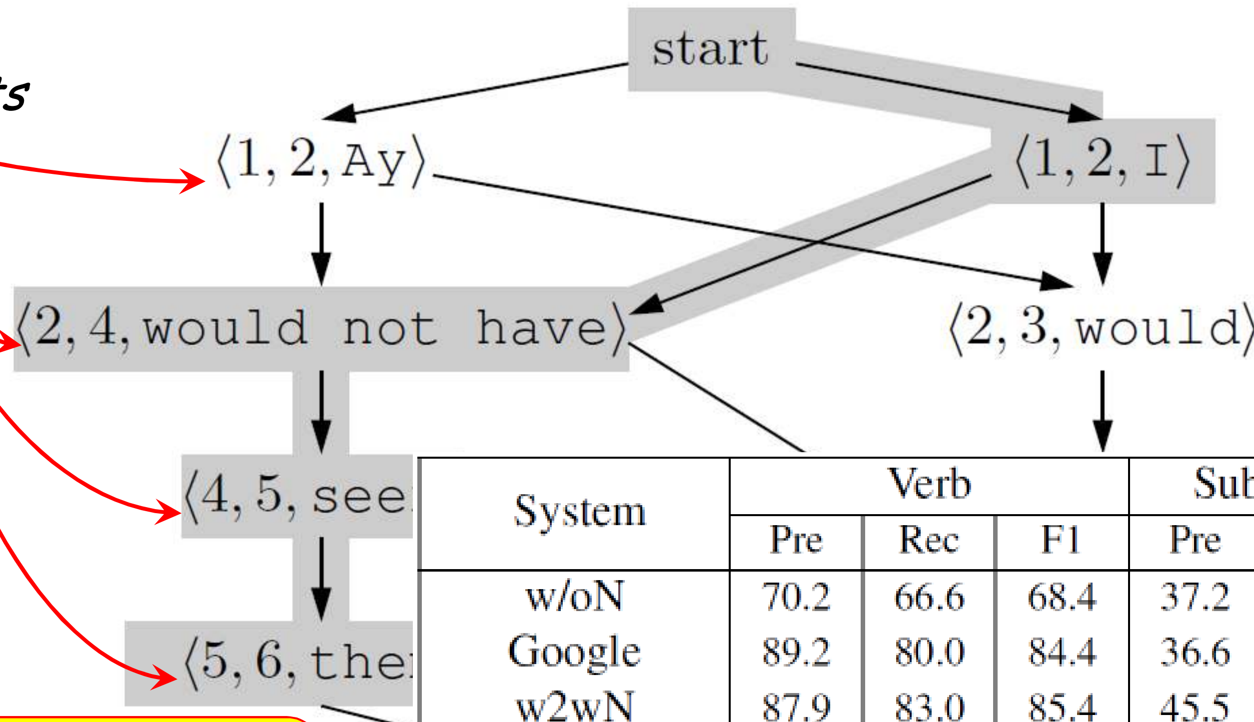
Normalization

- To deep-parse social media (tweets), we need to normalize the text into a more grammatical form
- Designed a normalizer based on a graph model
  - Zhang, Baldwin, Ho, Kimelfeld, Li: Adaptive Parser-Centric Text Normalization, ACL 2013
- Parameters tuned by supervised machine learning
- Customizable by mapping dictionaries
  - Contractions, abbreviations, etc.
  - Example: kinda → kind of, rep → the representative

Normalization Example

Ay woudent of see em.

Generated replacements



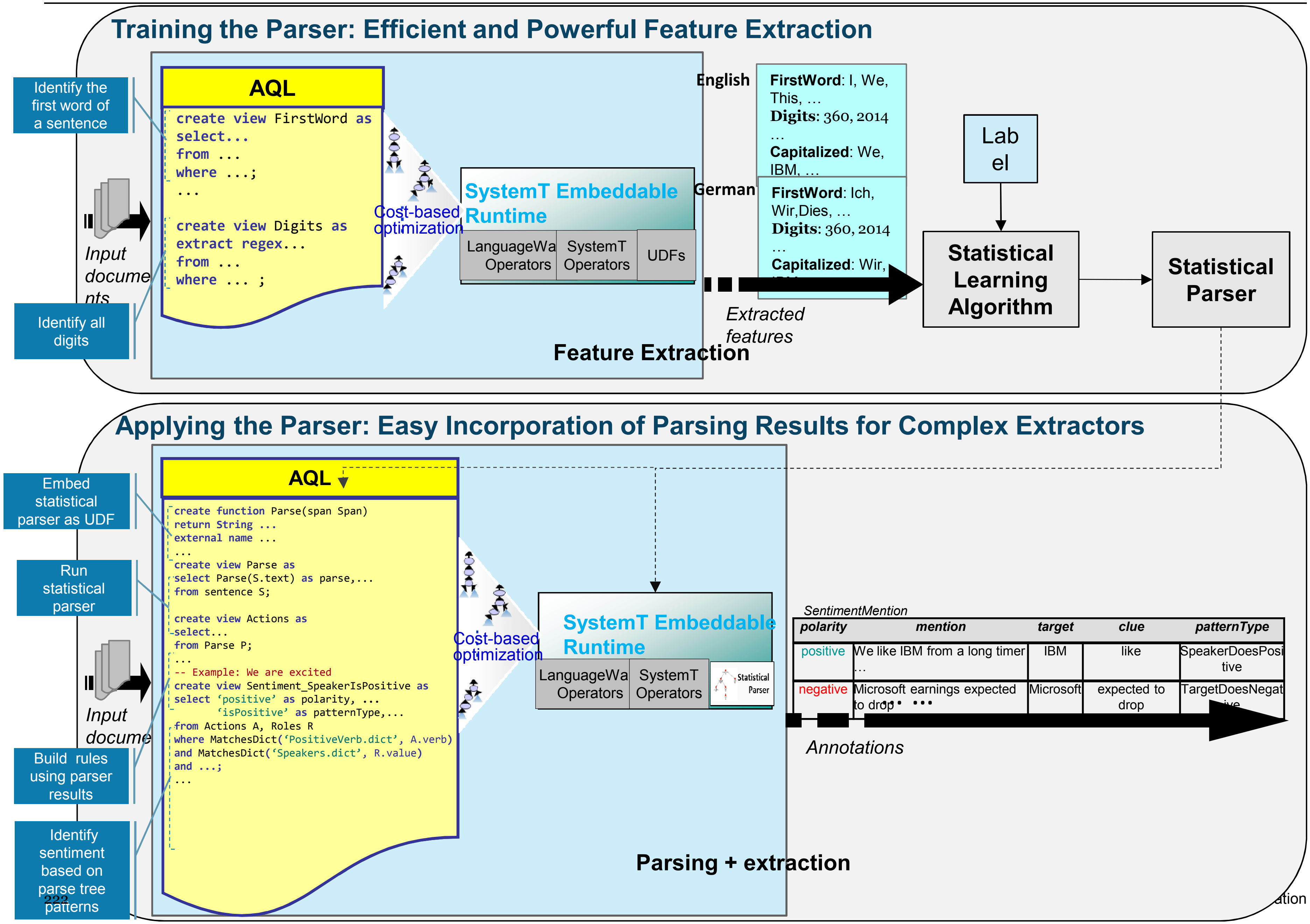
System	Verb			Subject-Object		
	Pre	Rec	F1	Pre	Rec	F1
w/oN	70.2	66.6	68.4	37.2	38.9	38.1
Google	89.2	80.0	84.4	36.6	46.9	41.1
w2wN	87.9	83.0	85.4	45.5	60.2	51.8
Gw2w	90.3	85.2	87.7	47.8	61.9	53.9
generic	92.2	90.4	91.3	55.1	72.1	62.5
domain specific	95.9	90.7	**93.2**	75.3	79.3	**73.4**

Targeted use of machine learning

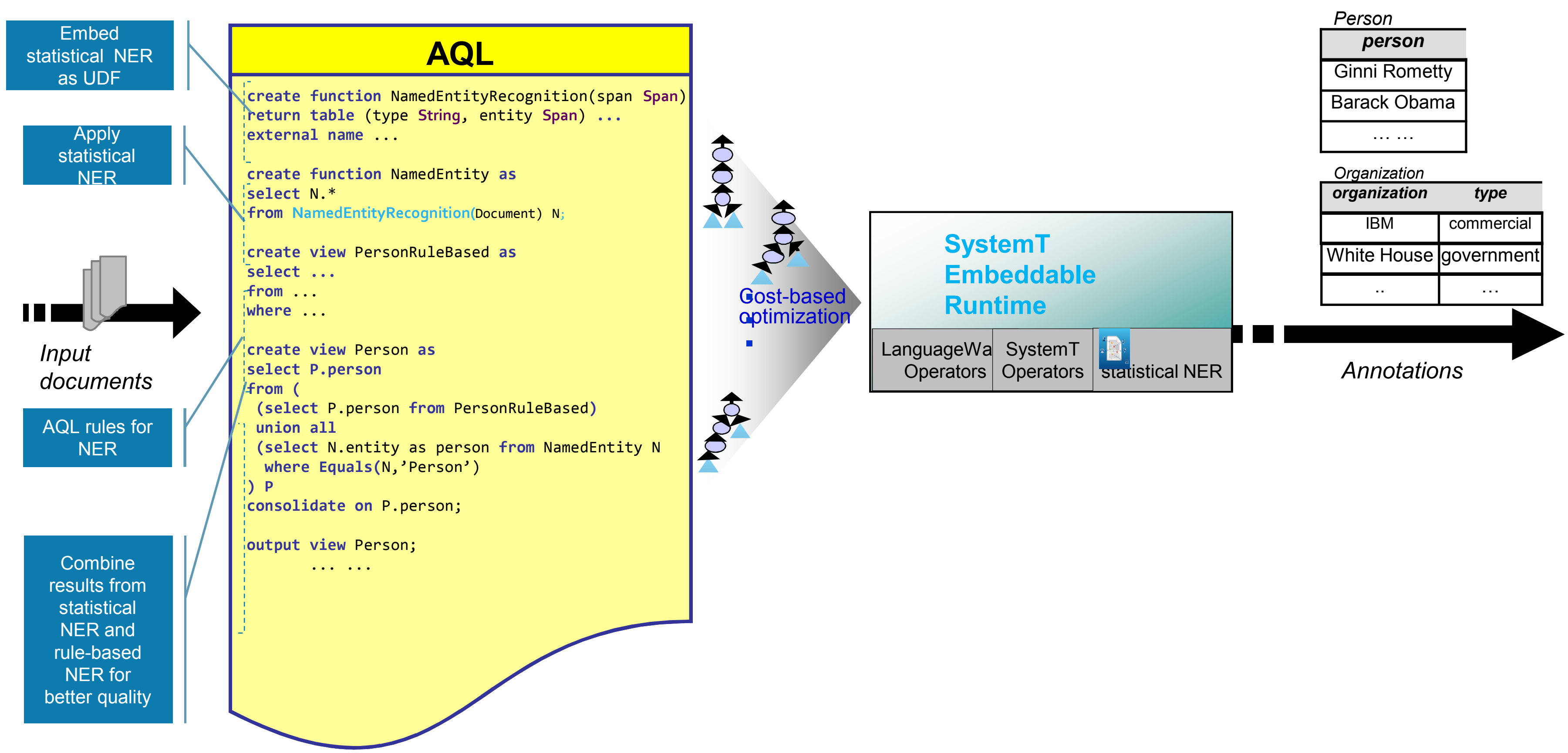
Machine Learning in SystemT

- Low-level features
- Rule refinement
- Rule induction
- Normalization
- Embedded Models

Simplify Training and Applying Statistical Parsers



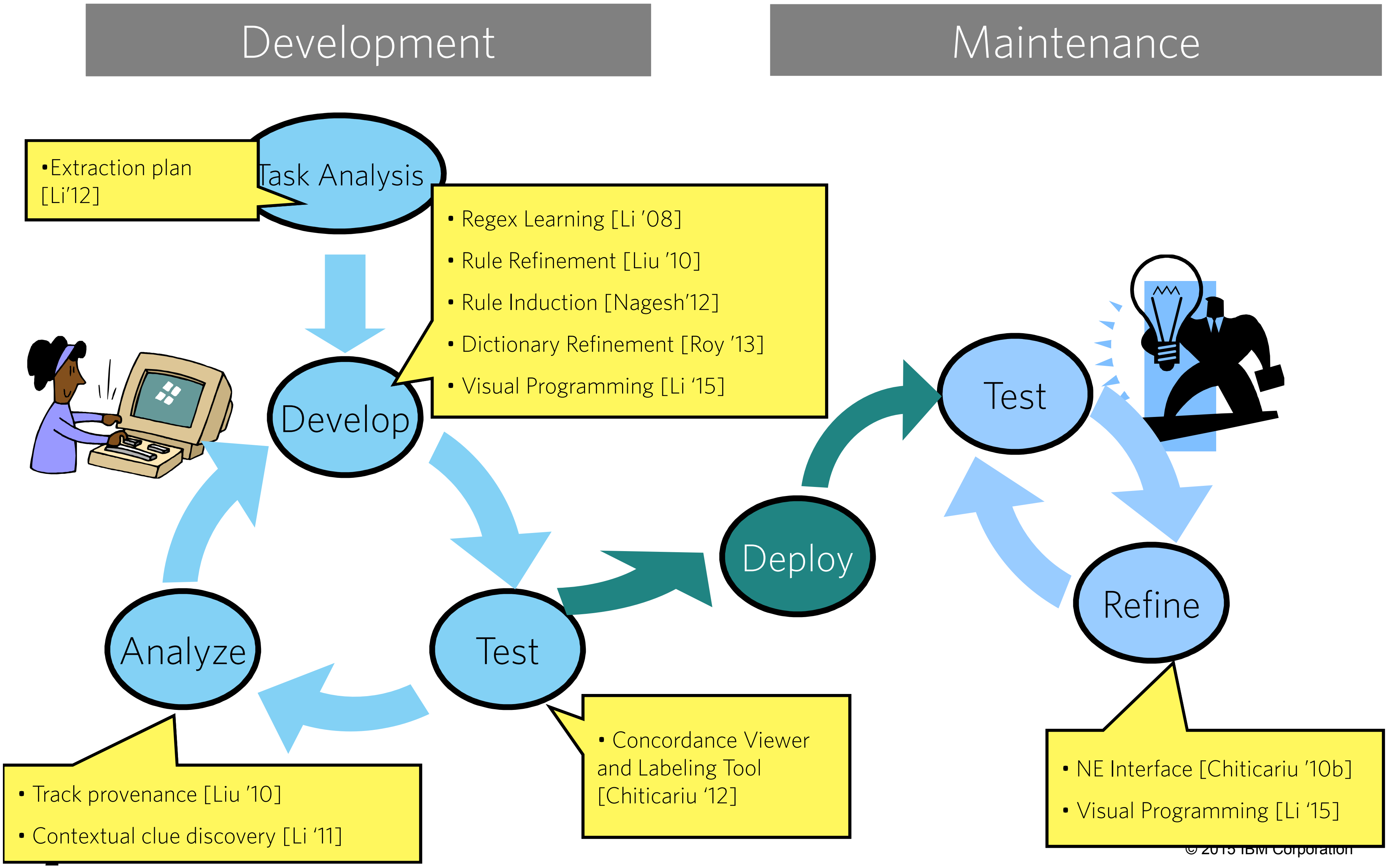
Combine Statistical and Rule-based NER for Better Quality



Outline

- Building a Transparent IE System
- Transparent Machine Learning
- Building Developer Tools around Transparent IE
- Case Study and Demo

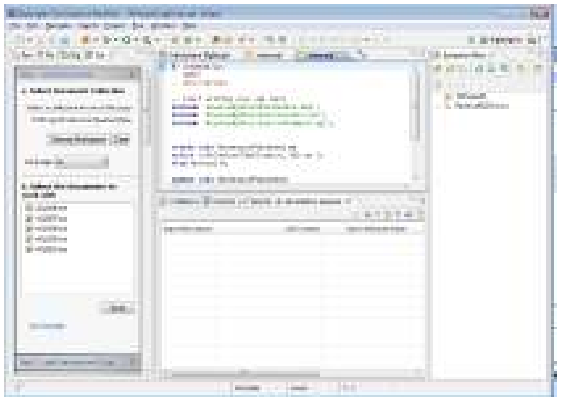
Transparent ML at different stages in Extractor Development



Different User Groups

1. Group 1

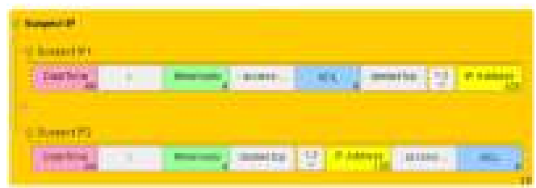
- data scientist
- programmer



- Full power AQL Eclipse-based tooling

2. Group 2

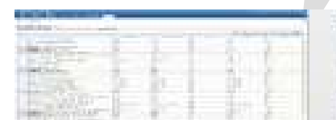
- data modeler
- data scientist
- technical sales
- enablement team

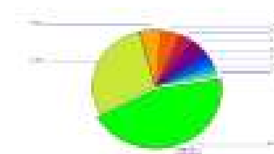


- Connect Concepts to build the extractor
- High-level visual abstraction
- Leverage smart parsing and pre-built concepts

3. Group 3

- business analyst
- business end user
- CTO





- Consumable visualization of extraction results

Powerful

Less Powerful

Difficult to learn/use

Simple

Eclipse Tools Overview

Ease of Programming

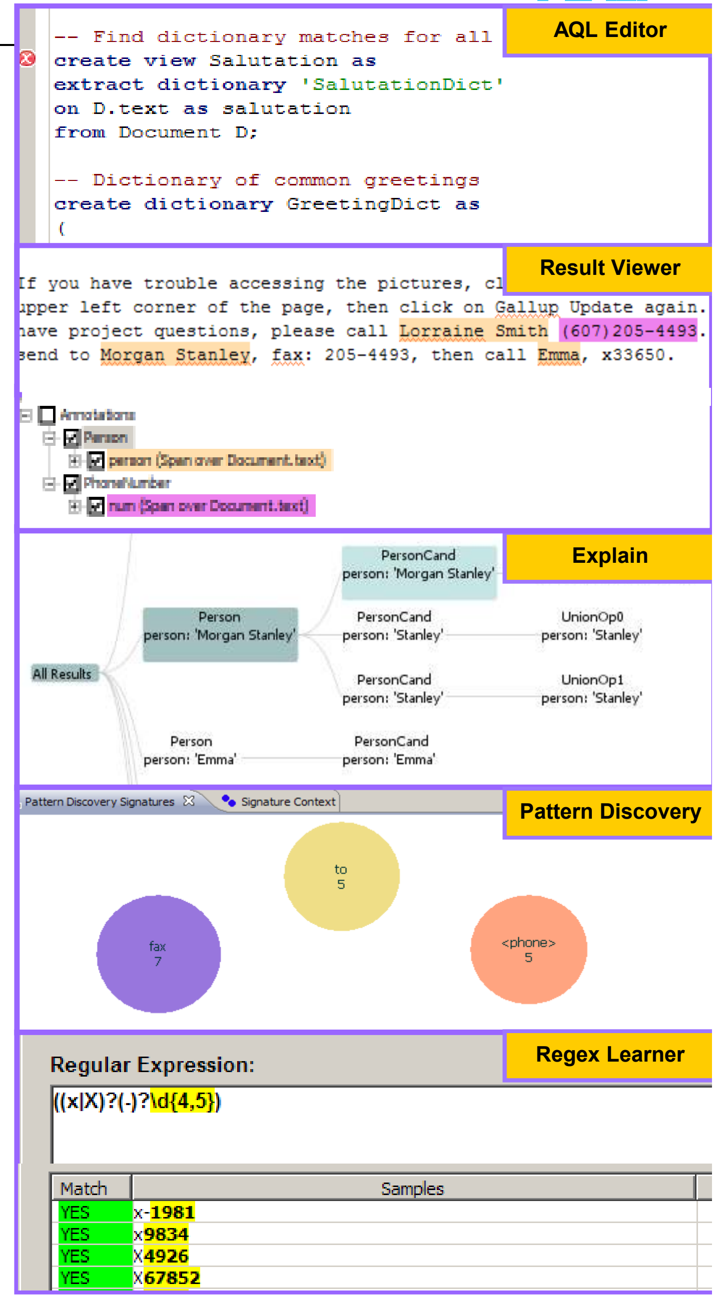
- **AQL Editor:** syntax highlighting, auto-complete, hyperlink navigation
- **Result Viewer:** visualize/compare/evaluate
- **Explain:** show how each result was generated
- **Workflow UI**: end-to-end development wizard

Automatic Discovery

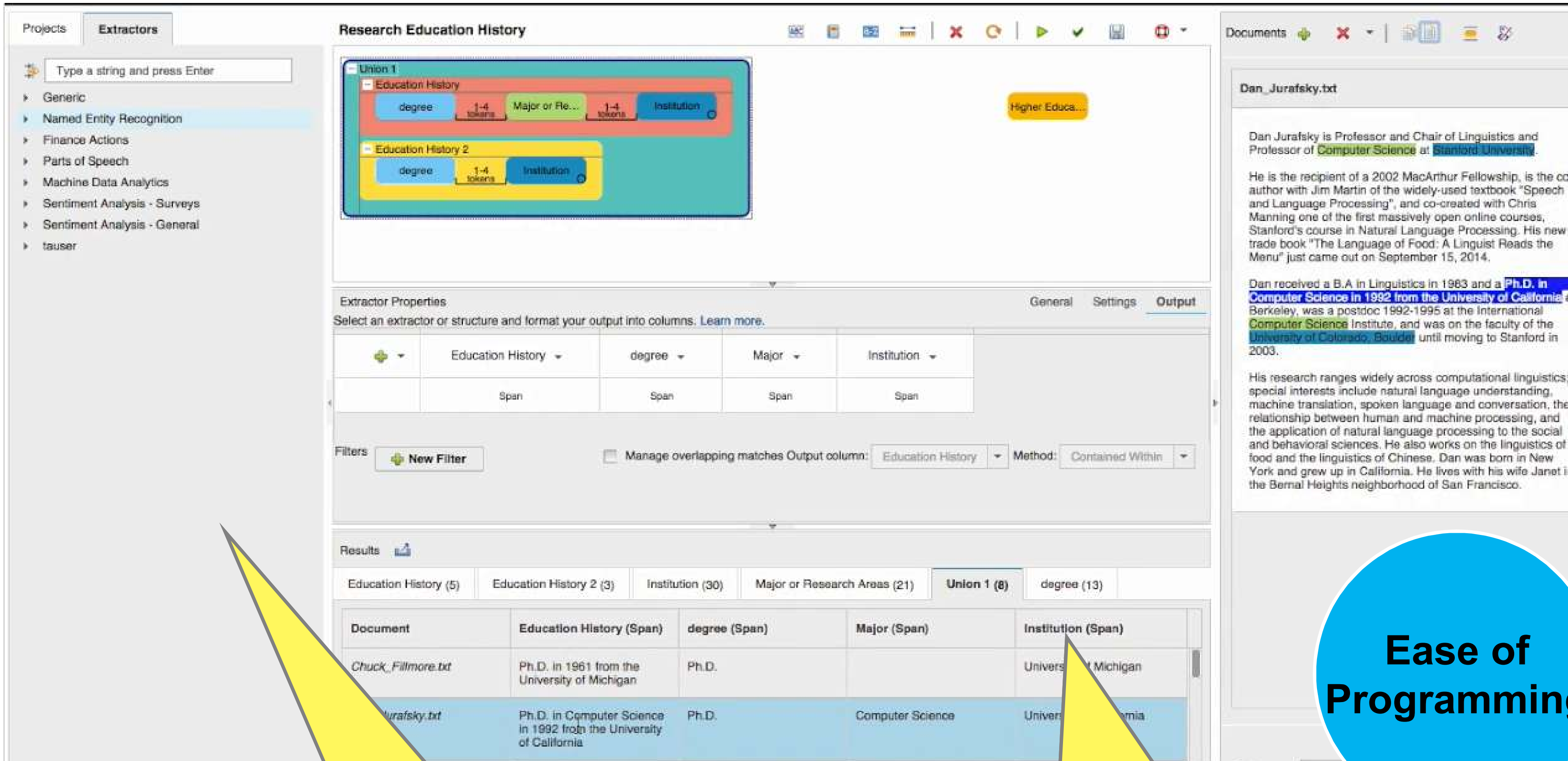
- **Regex Generator:** generate regular expressions from examples
- **Pattern Discovery**: identify patterns in the data

Performance Tuning

- **Profiler**: identify performance bottlenecks to be hand tuned



Web Tools Overview



Ease of Programming

Canvas: Visual construction of extractors, Customization of existing extractors

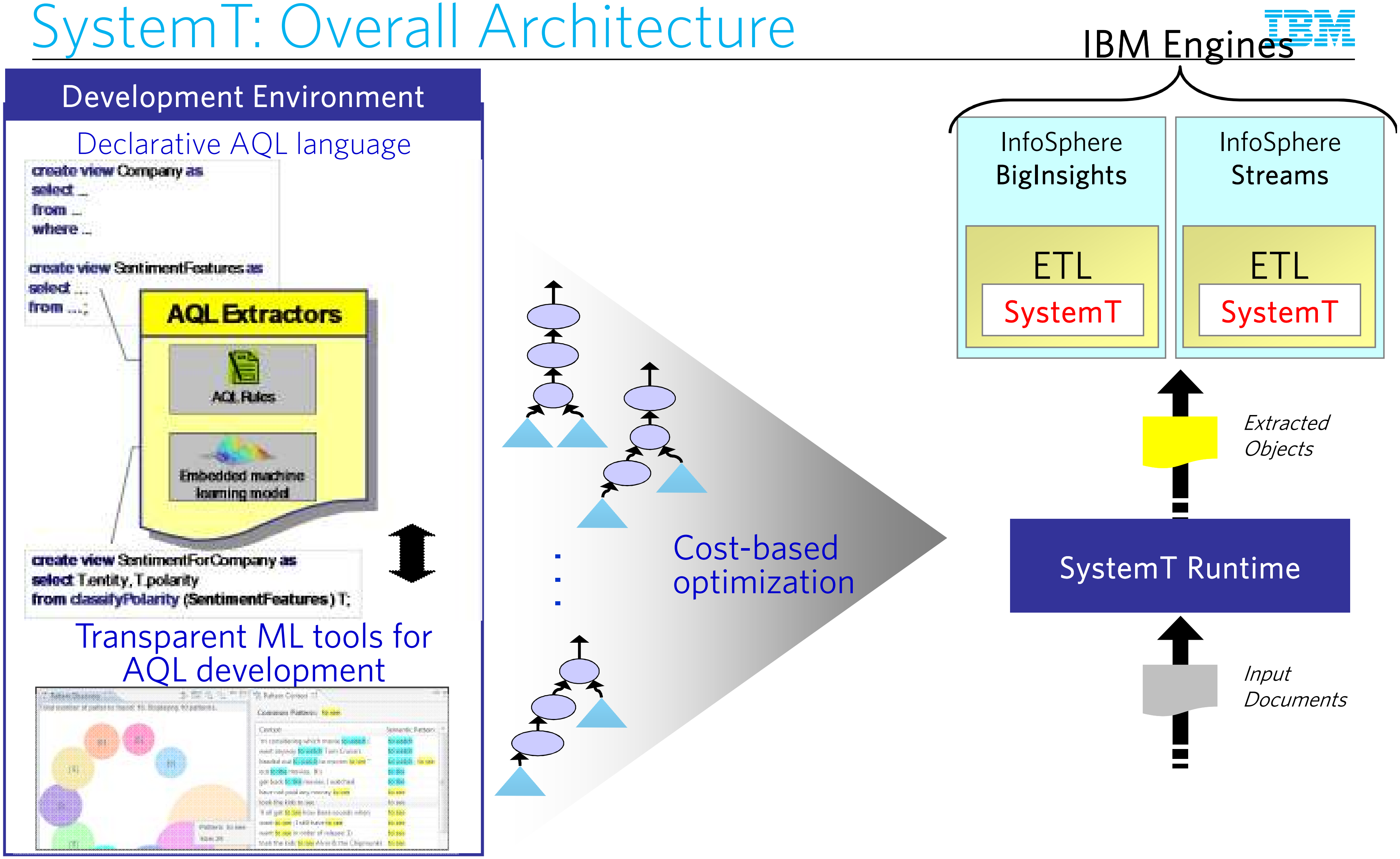
Result Viewer: visualize/compare/evaluate

Ease of Sharing

Concept catalog: share concepts

Project: share extractor development

SystemT: Overall Architecture



Rule language with familiar SQL-like syntax

Specify extractor semantics declaratively

Choose an efficient execution plan that implements the semantics

Embeddable Java runtime

Highly scalable, small memory footprint

Outline

- Building a Transparent IE System
- Transparent Machine Learning
- Building Developer Tools around Transparent IE
- Case Study and Demo

Case Study: Sentiment Analysis over Research Reports

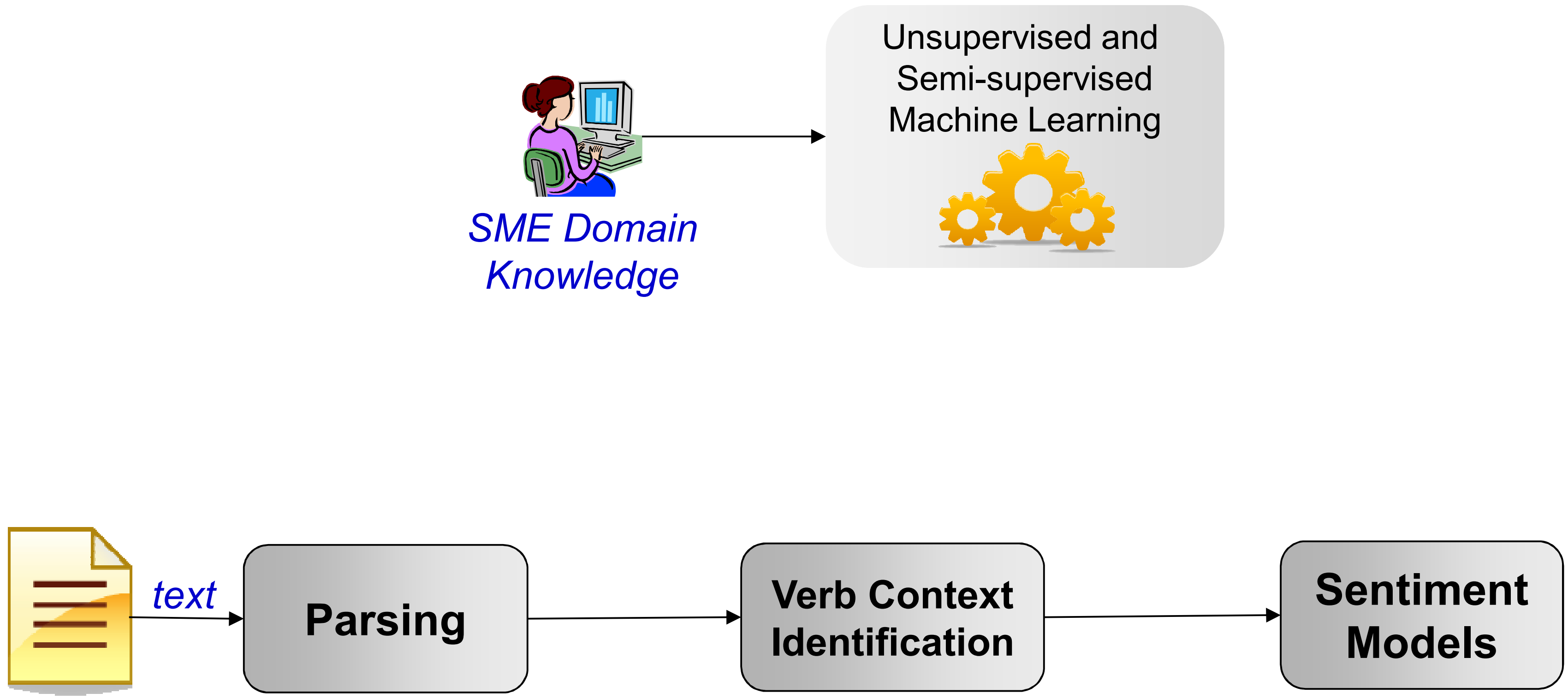
- Drawn from engagements with three major U.S. investment banks
- Basic problem: Automatically extract analysts' detailed opinions on securities and markets from analyst research reports
- Key challenges
  - Customizing for domain-specific expressions
  - Identifying the target of sentiment expressions
  - Aggregating sentiment by document

We are upgrading US equities back to Overweight on a 6-month.

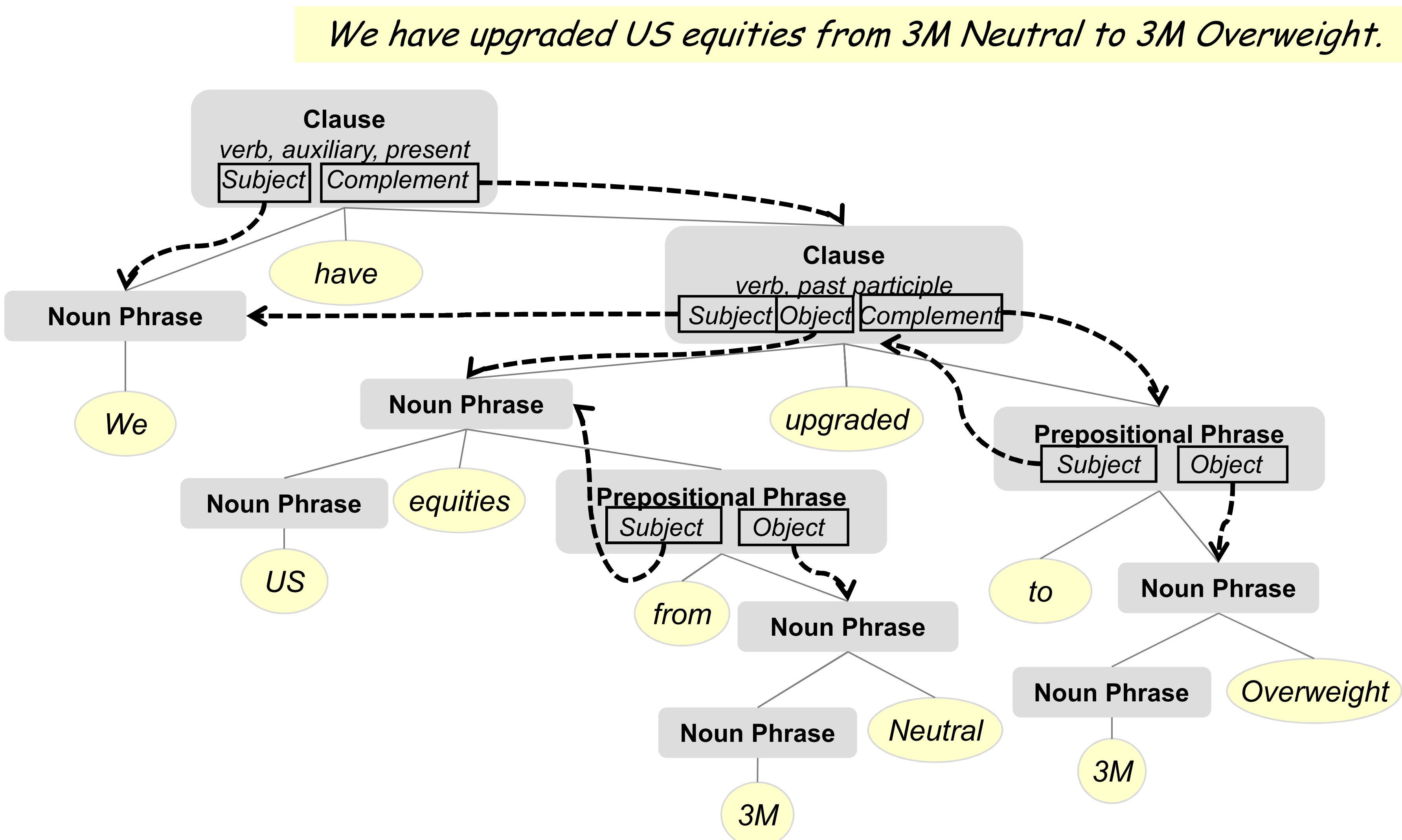
We have upgraded the Belgian market to Neutral from Underweight in the current quarter.

As a relative momentum call versus the weakness anticipated in ASEAN, we are upgrading Korea to Overweight, and upgrading Taiwan to Neutral in 1Q.

Sentiment Analysis over Research Reports



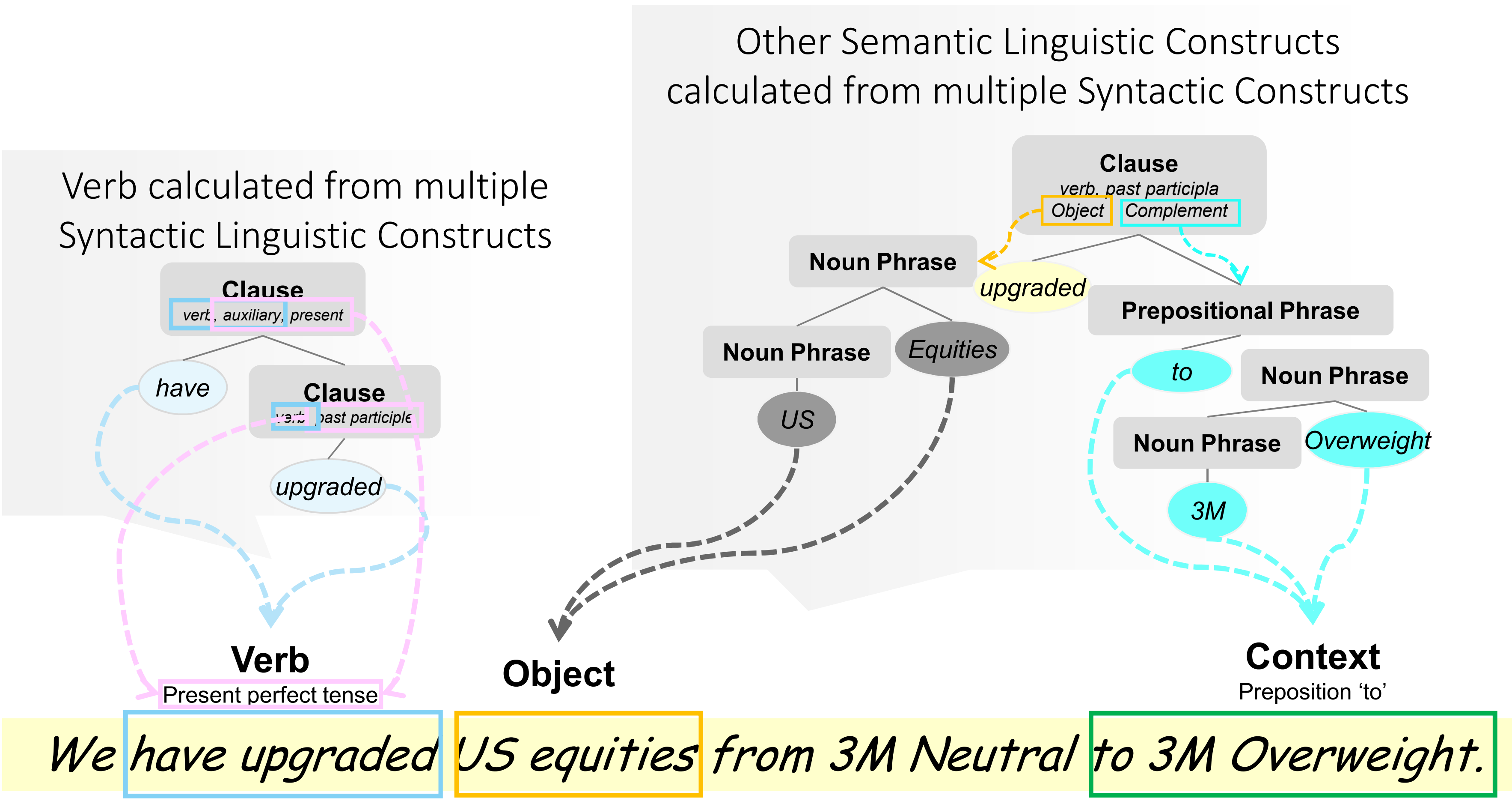
Phase 1: Parse



Transparent Machine Learning in the Parsing Phase

- Adaptive Text Normalization [Zhang et al., 2013]
  - Model targeted towards generating sentences that can be successfully parsed
  - Sequential rules + graph model
    - Explainable to a certain extent
  - Allows incorporation of domain knowledge at deployment
- The IBM English Slot Grammar Parser [McCord et al., 2012]
  - Candidate generation is rule-driven
  - Ranking is less transparent
  - Allows incorporation of domain knowledge at deployment
    - E.g., list of noun phrases, additional word senses

Phase 2: Identify Context



Transparent Machine Learning in the Context Identification Phase

- Dictionary Learning [Roy et al., 2013]
  - Refine dictionaries within an AQL rule set
  - Recall from Part 3

- Pattern Discovery [Li et al., 2011]
  - Unsupervised discovery of contextual patterns
    - E.g., financial metrics, asset class synonyms
  - Recall from Part 3

Phase 3: Assign Polarity

Outlook Model

Assign Positive polarity if:

- Verb is in present/future/infinitive tense
- Object contains entity
- Context w/ preposition 'to' contains positive recommendation (e.g. 'Overweight')

Domain knowledge

- Polarity determined by recommendation (Overweight) not Verb (upgrade)
- Statements in present perfect are relevant

Verb
Present perfect tense

Object

Context
Preposition 'to'

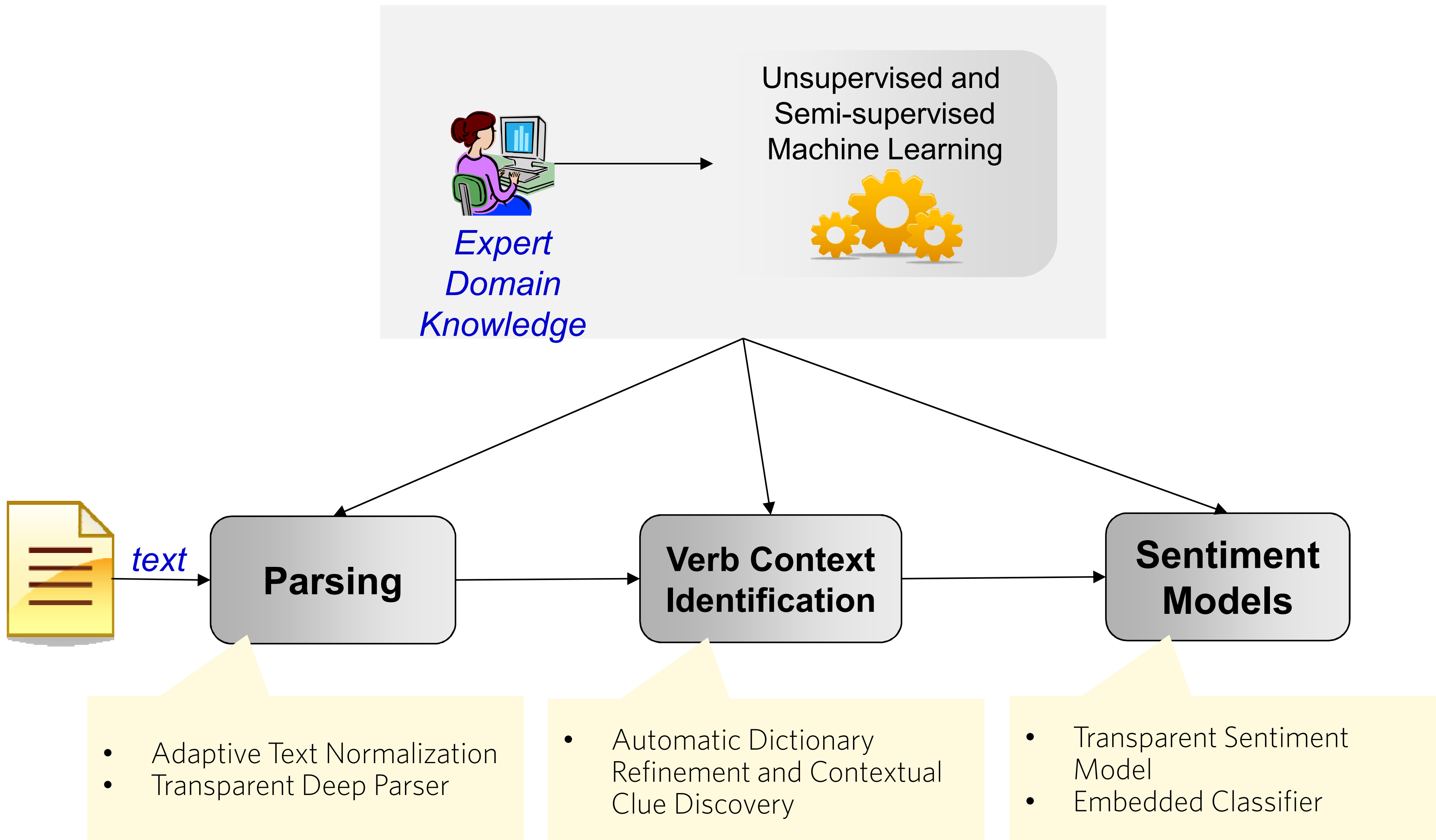
We have upgraded US equities from 3M Neutral to 3M Overweight.

Transparent Machine Learning in the Polarity Assignment Phase

- The sentiment model: AQL rules
  - Exposes customization points:
    - Dictionaries of sentiment clues
    - Disable or change the behavior of certain rules (e.g., discard past tense sentiments)
  - Generic model adapted for the domain, mostly manually
  - Automatic adaptation of dictionaries not possible due to absence of labeled data

- Sentiment Aggregation as a Classification Problem
  - Given individual sentiment instances for an entity from a document, classify the document-level polarity for the entity
  - SVM model trained based on (entity/polarity) pairs in 100 documents
  - Model embedded in AQL for scoring

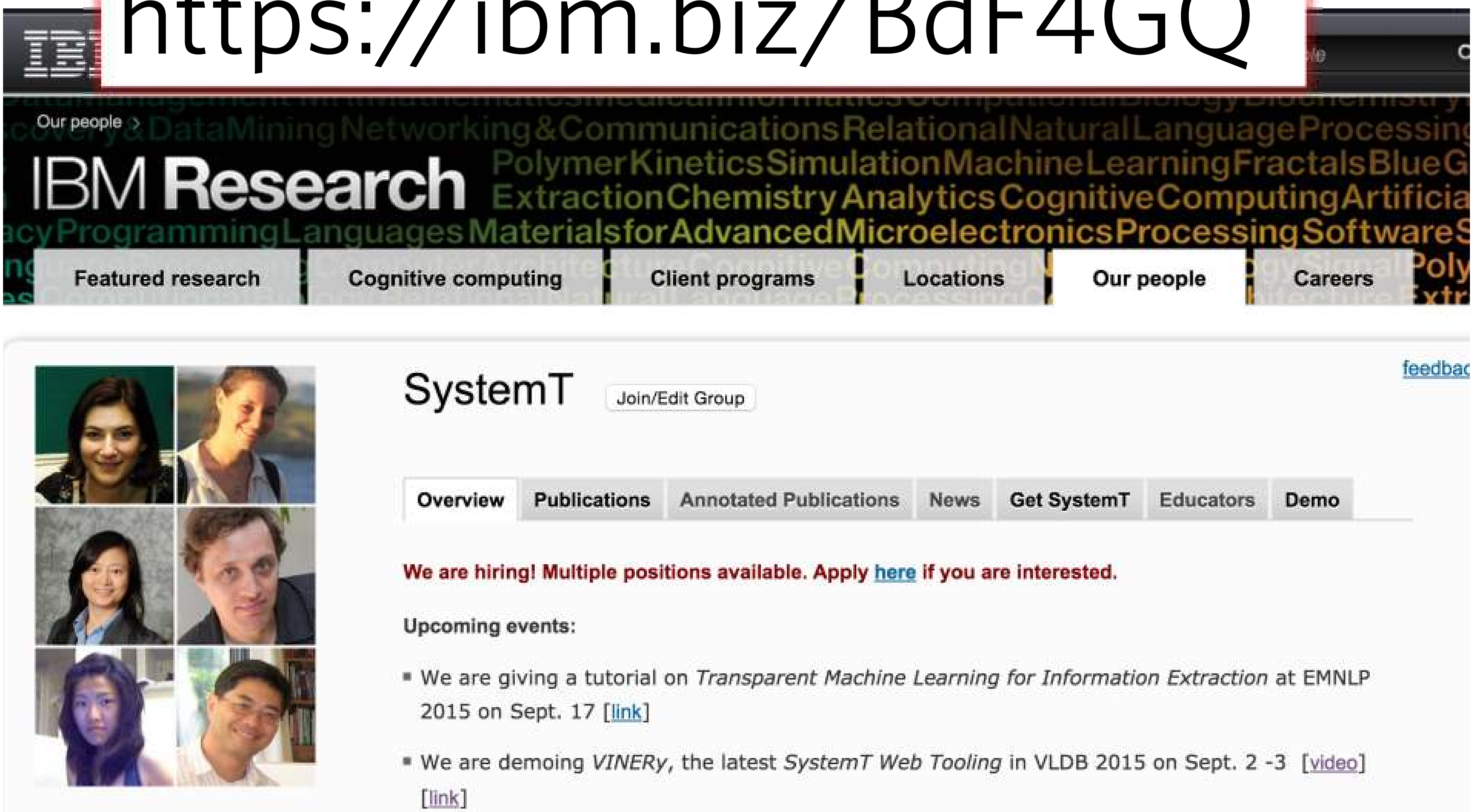
Sentiment Analysis over Research Reports: Transparent ML



Demo

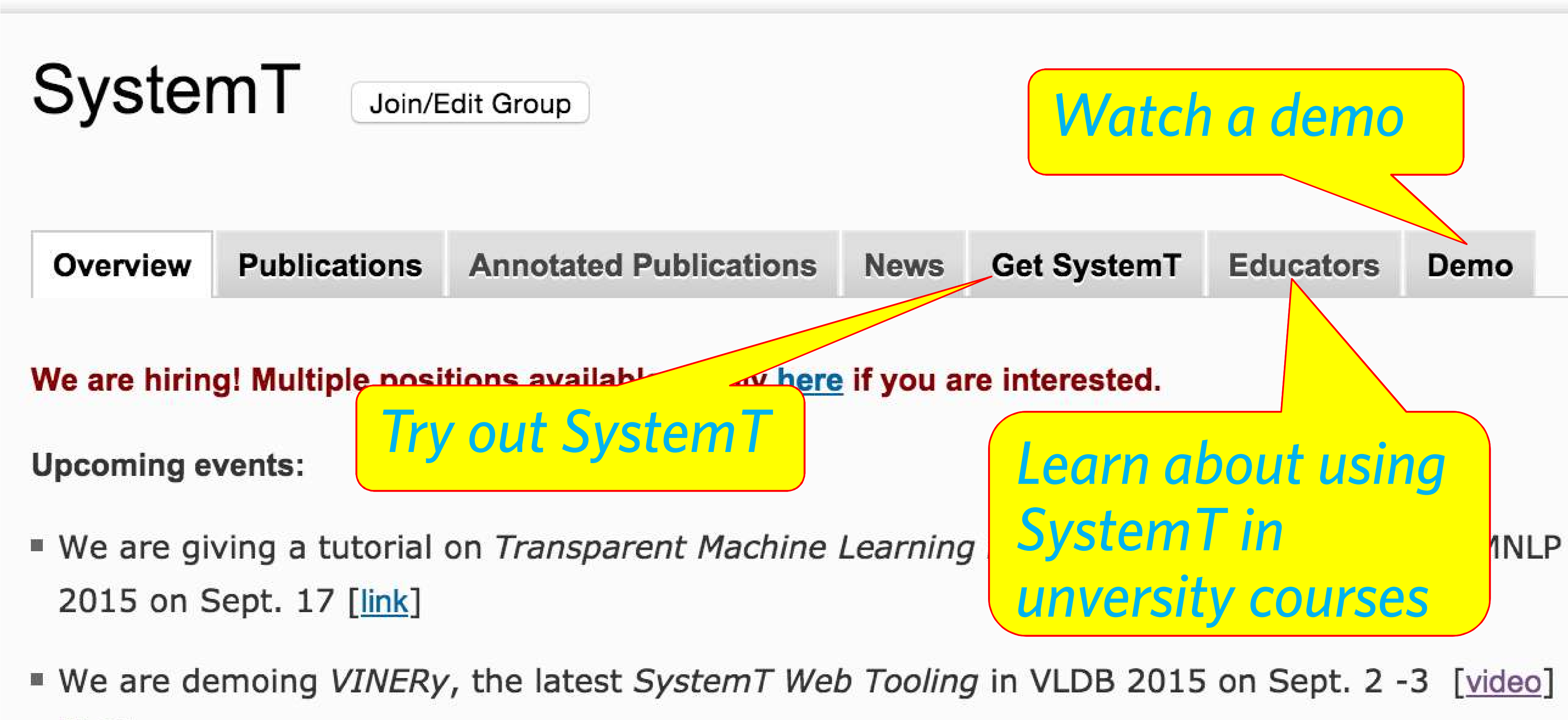
Find Out More about SystemT!

https://ibm.biz/BdF4GQ



Find Out More about SystemT!

https://ibm.biz/BdF4GQ



Other Systems

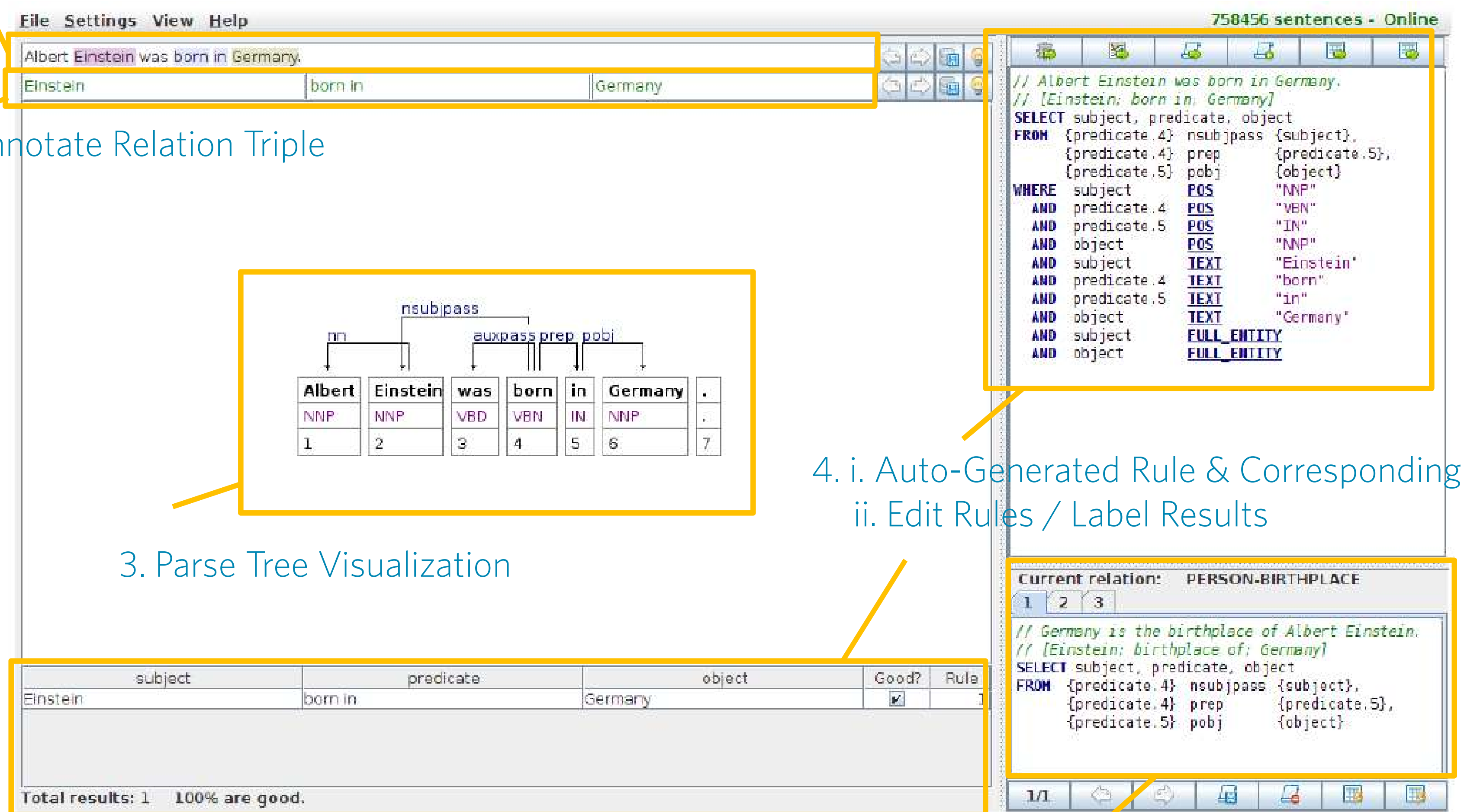
- PropMiner (TU Berlin) [Akbik et al., 2013]
- ICE (New York University) [He and Grishman, 2015]
- SPIED (Stanford) [Gupta and Manning, 2014]
- CHIMERA (WalmartLabs, U. Wisconsin-Madison) [Sun et al, 2014]
- BBN Technologies System [Freeman et al., 2011]
- INSTAREAD (U. Washington) [Hoffman et al., 2015]

PropMiner (TU Berlin)

[Akbik et al, 2013]

1. Construct Example Sentence

2. Annotate Relation Triple



3. Parse Tree Visualization

4. i. Auto-Generated Rule & Corresponding Results
ii. Edit Rules / Label Results

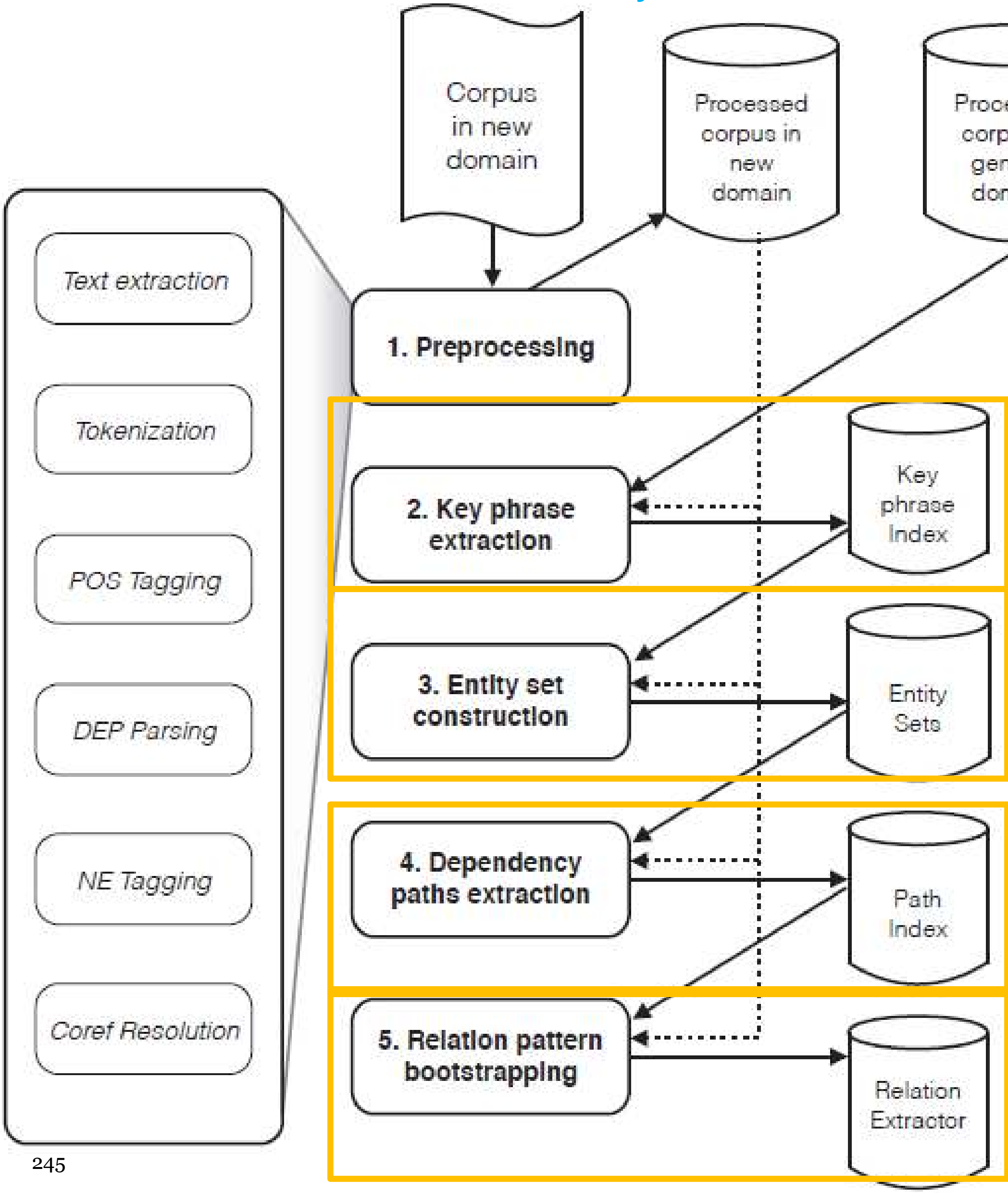
Additional features:

1. Sentence suggestion
2. Conflict resolution

5. Relevant Existing Rules

ICE (New York University)

[He and Grishman, 2015]



1. A ranked list of key phrases. Key phrases appear more often in the in-domain corpus than in general language will rank higher.

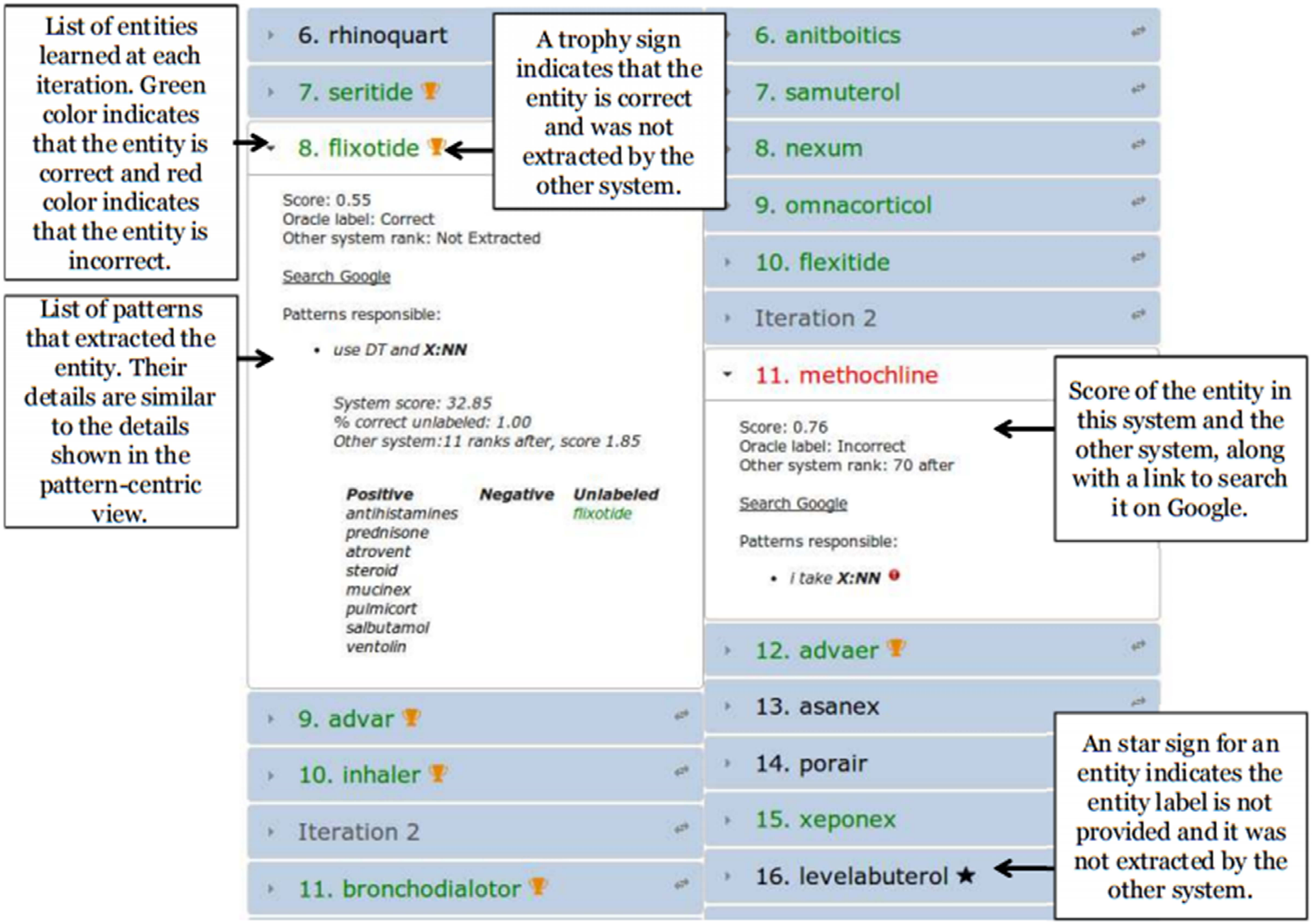
2. Given user-given or auto-constructed seeds, automatically construct a ranked list of similar terms in the corpus.

3. Linearize lexicalized dependency path for easier understanding.

4. Auto-construct exact and fuzzy dependency-path based relation extractors with bootstrapping user input

SPIED (Stanford)

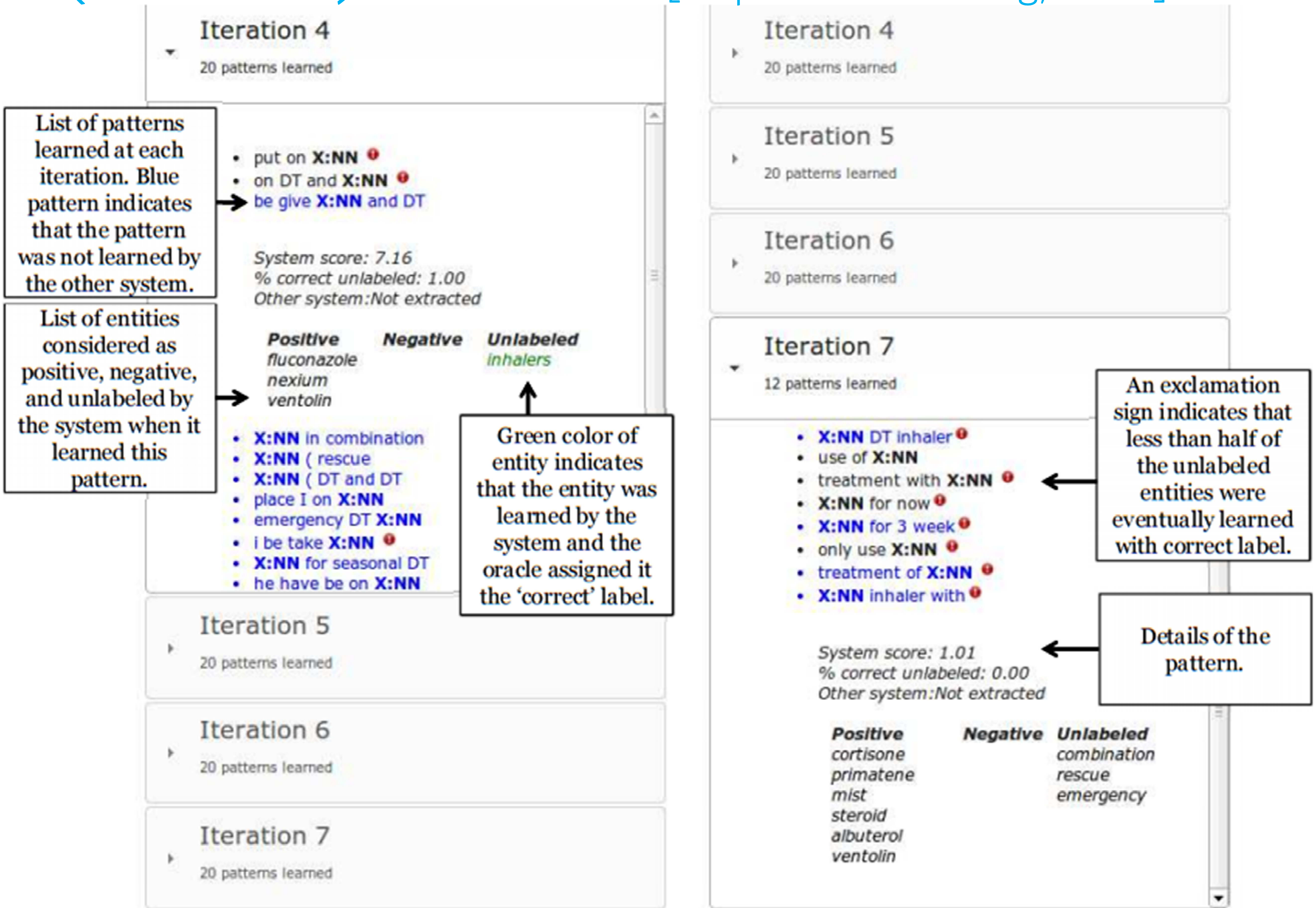
[Gupta and Manning, 2014]



Entity-centric view

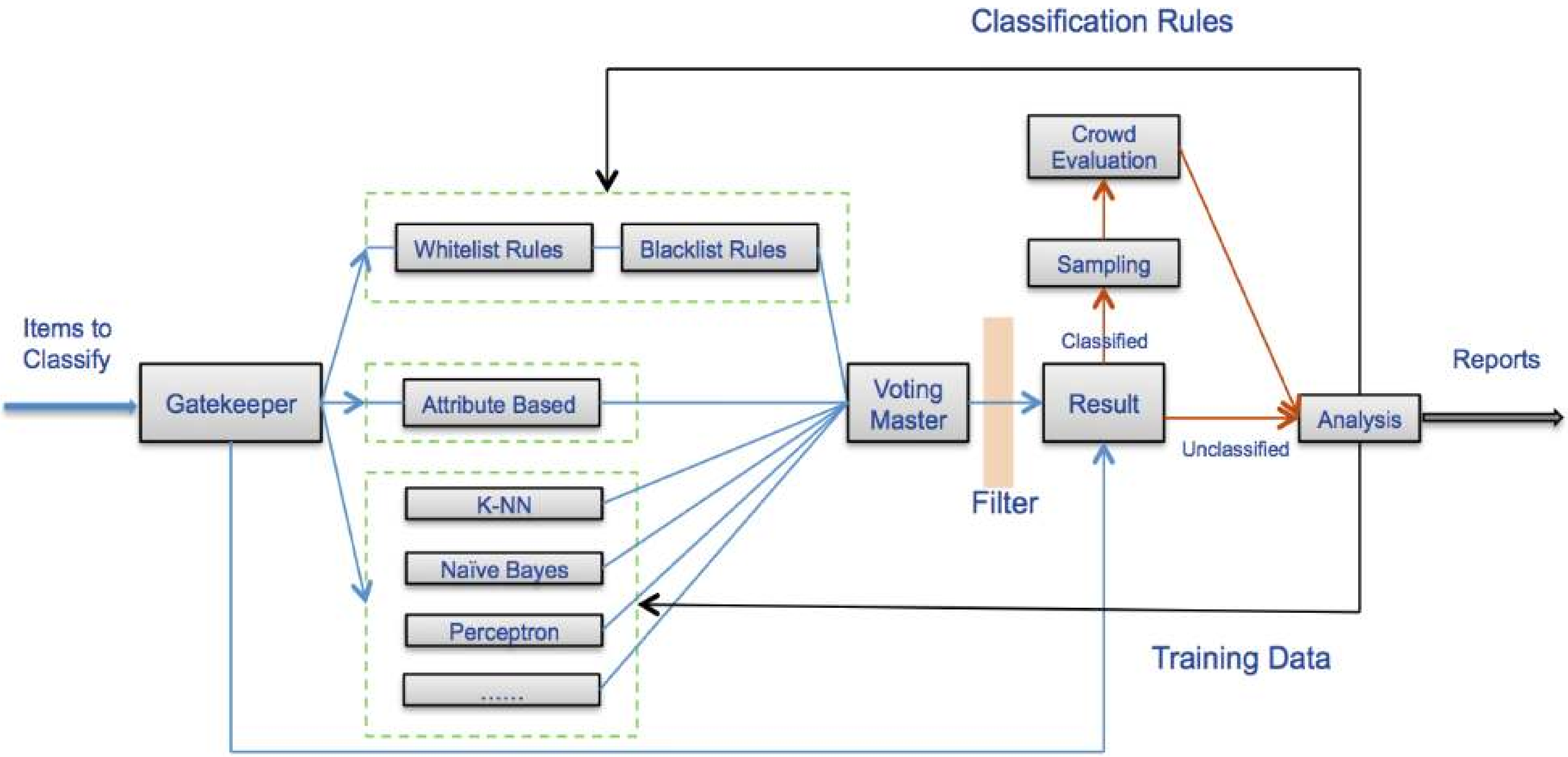
SPIED (Stanford)

[Gupta and Manning, 2014]



Pattern-centric view

CHIMERA (WalmartLabs, Univ. Wisconsin-Madison) [Sun et al, 2014]



Combine rule-based and machine learning based approaches to overcome

Challenges for ML-based approach:

1. Difficult to generate training data
2. Difficult to Generate Representative Sample
3. Difficult to Handle "Corner Cases"
4. Concept Drift & Changing Distribution

Challenges for rule-based approach:

1. Labor intensive
2. Time consuming
3. Cannot utilize existing labeled data

BBN Technologies System

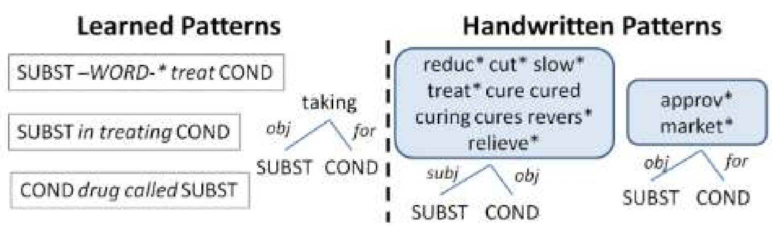
[Freeman et al, 2011]

Third-party Ontology and Resources
(guidelines/examples/sample documents)

Domain-Specialization
- Class detector based on unsupervised clustering
- Manually-added coreference heuristics
- Seed-based bootstrap relation learner
- Manually-developed rules in a pattern langu...

Existing ACE-specific Extractors

Opaque step



Sample patterns for *possibleTreatment*

INSTAREAD (University of Washington)

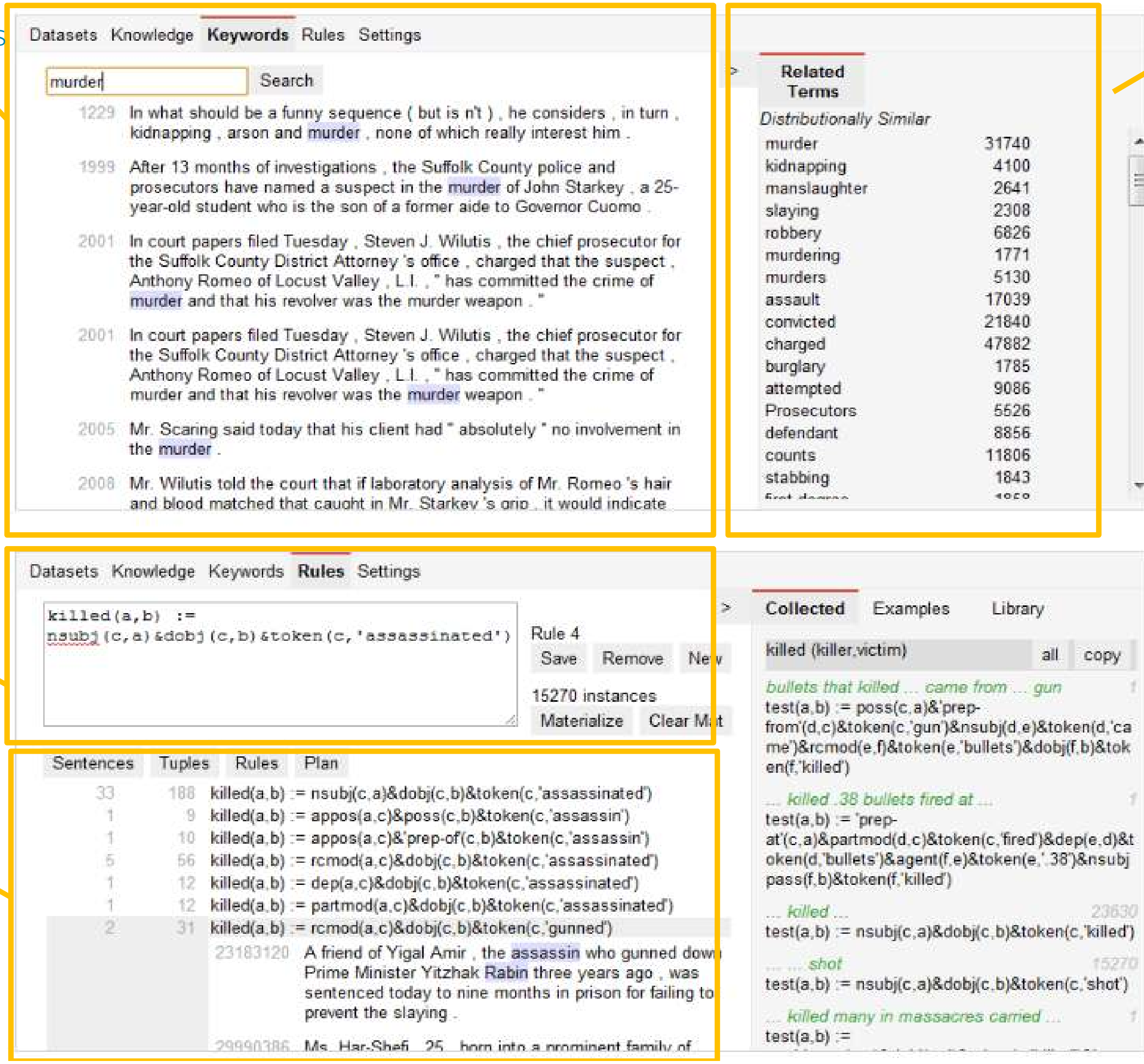
[Hoffman et al. 2015]

1. Identify examples by search.

2. Suggest related terms for more examples

3. User-created/refined rule

4. Auto-suggested rules via bootstrapping



Transparent ML for Information Extraction: Research Challenges and Future Directions

Research Challenges

- How to make transparent ML for IE more **principled**, **effective**, and **efficient?**

Future Directions - 1

- Define a standard IE language and data model
  - What is the right data model to capture text, annotations over text, and their properties?
  - Can we establish a standard declarative extensible language to solve most IE tasks encountered so far?
  - Desired characteristics:
    - Expressivity:

      Able to represent and combine different kinds of transparent models of representation
    - Extensibility:

      Allow new models to be added in the future
    - Declarativity:

      Enable optimization, scalability, explainability

Future Directions - 2

- Systems research based on a standard IE language
  - Data representation
  - Automatic performance optimization
  - Exploring modern hardware

Future Directions - 3

- ML research based on a standard IE language
  - How to learn basic primitives such as regular expressions and dictionaries?
  - How to automatically generate models that are comprehensible and debuggable ?
  - How to design learning algorithms that are more comprehensible and debuggable ?
  - How to enable easy incorporation of domain knowledge?

References

- [Agichtein & Gravano 2000] Eugene Agichtein, Luis Gravano. Snowball: extracting relations from large plain-text collections. ACM Conference on Digital Libraries, 2000
- [Aitken 2002] Learning information extraction rules: An inductive logic programming approach, European Conference on Artificial Intelligence, 2002
- [Baldwin et al, 2013] Tyler Baldwin, Yunyao Li, Bogdan Alexe, Ioana R Stanoi. Automatic Term Ambiguity Detection. ACL 2013
- [Banko et al., 2007] Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, Oren Etzioni: Open Information Extraction from the Web. IJCAI 2007: 2670-2676
- [Brauer et al., 2011] Falk Brauer, Robert Rieger, Adrian Mocan, Wojciech M. Barczynski:Enabling information extraction by inference of regular expressions from sample entities. CIKM 2011
- [Brin 1998] Sergey Brin. Extracting Patterns and Relations from the World Wide Web. WebDB, 1998
- [Cafarella et al., 2005] Michael J. Cafarella, Doug Downey, Stephen Soderland, Oren Etzioni . KnowItNow: fast, scalable information extraction from the web. HLT 2005
- [Califf & Mooney 1999] M. E. Calif and R. J. Mooney, Relational learning of pattern-match rules for information extraction, AAAI 1999
- [Califf & Mooney 2003] M. Calif and R. Mooney, Bottom-up Relational Learning of Pattern Matching Rules for Information Extraction, Journal of Machine Learning 2003.
- [Carlson et al. 2010] Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka, Jr., Tom M. Mitchell , Coupled semi-supervised learning for information extraction, WSDM 2010
- [Chang & Manning 2014] Angel X. Chang and Christopher D. Manning. 2014. TokensRegex: Defining cascaded regular expressions over tokens. Stanford University Technical Report, 2014.
- [Chiticariu et al., 2010] Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Frederick Reiss, Shivakumar Vaithyanathan. Domain adaptation of rule-based annotators for named-entity recognition tasks. EMNLP 2010
- [Cheney et al, 2009] James Cheney, Laura Chiticariu, Wang Chiew Tan: Provenance in Databases: Why, How, and Where. Foundations and Trends in Databases 1(4): 379-474 (2009)
- [Choi et al., 2005] Yejin Choi, Claire Cardie, Ellen Riloff, Siddharth Patwardhan: Identifying Sources of Opinions with Conditional Random Fields and Extraction Patterns. EMNLP 2005
- [Ciravegna 2001] F. Ciravegna, Adaptive information extraction from text by rule induction and generalization. IJCAI 2001
- [Coden 2014] Anni Coden, Daniel Gruhl, Neal Lewis, Pablo N. Mendes, Meena Nagarajan, Cartic Ramakrishnan, Steve Welch: Semantic Lexicon Expansion for Concept-Based Aspect-Aware Sentiment Analysis. SemWebEval@ESWC 2014: 34-40
- [Cohen 1995] W. Cohen. Fast effective Rule Induction. ICML 1995
- [Cohen & Singer 1999] W. Cohen, Y. Singer. A Simple, Fast and Effective Rule Learner. AAAI 1999
- [Davidson & Freire, 2008] Susan B. Davidson, Juliana Freire: Provenance and scientific workflows: challenges and opportunities. SIGMOD 2008
- [Del Corro & Gemulla 2013] Luciano Del Corro, Rainer Gemulla. ClausIE: clause-based open information extraction. WWW 2013
- [Downey et al., 2004] Doug Downey, Oren Etzioni, Stephen Soderland, and Daniel S. Weld. Learning Text Patterns for Web Information Extraction and Assessment. AAAI Workshop on Adaptive Text Extraction and Mining, 2004
- [Downey et al. 2007] Doug Downey, Matthew Broadhead, Oren Etzioni: Locating Complex Named Entities in Web Text. IJCAI 2007
- [Etzioni et al, 2005] Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, Alexander Yates. Unsupervised named-entity extraction from the web: an experimental study. J Artificial Intelligence 2005

References

- [Fader et al. 2011] Anthony Fader, Stephen Soderland, Oren Etzioni. Identifying relations for open information extraction. EMNLP 2011
- [Gerow 2014 A. Gerow. Extracting clusters of specialist terms from unstructured text. EMNLP 2014
- [Grishman &Min 2010] New York University KBP 2010 Slot-Filling System, Ralph Grishman and Bonan Min, TAC Workshop 2010
- [Gupta & Manning, 2014] Sonal Gupta and Christopher Manning. Improved Pattern Learning for Bootstrapped Entity Extraction. ACL 2014
- [Hoffman et al. 2015] R. Hoffmann, L. Zettlemoyer, D.S. Weld . Extreme Extraction: Only One Hour per Relation. June 2015.
- [Ji et al, 2010] H Ji, R Grishman, HT Dang, K Griffitt, J Ellis, Overview of the TAC 2010 Knowledge Base Population Track, TAC Workshop, 2010
- [Krishnamurthy et al., 2008] Rajasekar Krishnamurthy, Yunyao Li, Sriram Raghavan, Frederick Reiss, Shivakumar Vaithyanathan, Huaiyu Zhu: SystemT: a system for declarative information extraction. SIGMOD Record 37(4): 7-13 (2008)
- [Kudo & Matsumoto 2004] T. Kudo and Y. Matsumoto. 2004. A boosting algorithm for classification of semi-structured text. EMNLP 2004
- [Kobayashi et al. 2007] Nozomi Kobayashi , Kentaro Inui , Yuji Matsumoto .Extracting Aspect-Evaluation and Aspect-Of Relations in Opinion Mining , EMNLP 2007
- [Le & Gulwani 2014] Vu Le, Sumit Gulwani: FlashExtract: a framework for data extraction by examples. PLDI 2014
- [Li et al., 2008] Yunyao Li, Rajasekar Krishnamurthy, Sriram Raghavan, Shivakumar Vaithyanathan, H. V. Jagadish: Regular Expression Learning for Information Extraction. EMNLP 2008
- [Li et al, 2011] Y. Li, V. Chu, S. Blohm, H. Zhu, H. Ho. Facilitating pattern discovery for relation extraction with semantic-signature-based clustering. CIKM 2011
- [Liu et al, 2010] Automatic Rule Refinement for Information Extraction, Bin Liu, Laura Chiticariu, Vivian Chu, H. V. Jagadish, Frederick Reiss, PVLDB 3(1), 2010
- [Mausam et al. 2012] Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, Oren Etzioni. Open language learning for information extraction. EMNLP 2012
- [McIntosh & Curran 2009] Tara McIntosh, James R. Curran: Reducing Semantic Drift with Bagging and Distributional Similarity. ACL/IJCNLP 2009
- [McIntosh 2010] T. McIntosh: Unsupervised Discovery of Negative Categories in Lexicon Bootstrapping. EMNLP 2010

References

- [Mitchell et al. 2015] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, J. Welling. Never Ending Learning. AAAI 2015
- [Muslea 1999] Ion Muslea. Extraction patterns for information extraction tasks: A survey. In: AAAI Workshop on Machine Learning for Information Extraction, 1999
- [Nagesh et al, 2012] Ajay Nagesh, Ganesh Ramakrishnan, Laura Chiticariu, Rajasekar Krishnamurthy, Ankush Dharkar, Pushpak Bhattacharyya. Towards Efficient Named-Entity Rule Induction for Customizability, EMNLP-CoNLL, 2012
- [Pasca et al., 2006] Marius Paşca, Dekang Lin, Jeffrey Bigham, Andrei Lifchits, Alpa Jain. Names and similarities on the web: fact extraction in the fast lane. ACL 2006
- [Prasse et al., 2012] Paul Prasse, Christoph Sawade, Niels Landwehr, Tobias Scheffer: Learning to Identify Regular Expressions that Describe Email Campaigns. ICML 2012
- [Qadir 2012] Qadir, A. and Riloff, E. Ensemble-based Semantic Lexicon Induction for Semantic Tagging, SEM 2012
- [Riloff 1993], Ellen Riloff: Automatically Constructing a Dictionary for Information Extraction Tasks. AAAI 1993
- [Riloff 1996] Ellen Riloff:Automatically Generating Extraction Patterns from Untagged Text. AAAI 1996
- [Riloff & Jones 1999] Riloff, E. and Jones, R. Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping, AAAI 1999
- [Roy et al, 2013], Provenance-based dictionary refinement in information extraction, Sudeepa Roy, Laura Chiticariu, Vitaly Feldman, Frederick Reiss, Huaiyu Zhu, SIGMOD, 2013
- [Qadir & Riloff, 2012] A. Qudir, E. Riloff. Ensemble-based Semantic Lexicon Induction for Semantic Tagging, *SEM 2012
- [Qadir et al., 2015] A. Qudir, P. Mendes, D. Gruhl, N. Lewis. Semantic lexicon induction from Twitter with pattern relatedness and flexible term length. AAAI 2015
- [Qiu & Zhang 2014] Likun Qiu, Yue Zhang. ZORE: A Syntax-based System for Chinese Open Relation Extraction. EMNLP 2014
- [Sarawagi 2008] Sunita Sarawagi: Information Extraction. Foundations and Trends in Databases 1(3): 261-377 (2008)
- [Shen et al., 2007] W. Shen, A.Doan, J. F. Naughton, R. Ramakrishnan. Declarative Information Extraction Using Datalog with Embedded Extraction Predicates. VLDB 2007
- [Soderland 1999] S. Soderland, Learning information extraction rules for semi-structured and free text, Machine Learning, vol. 34, 1999.
- [Sudo et al., 2003] Kiyoshi Sudo, Satoshi Sekine, Ralph Grishman. An improved extraction pattern representation model for automatic IE pattern acquisition. ACL 2003
- [Surdeanu et al., 2003] Mihai Surdeanu, Sanda Harabagiu, John Williams, Paul Aarseth. Using predicate-argument structures for information extraction. ACL 2003
- [Surdeanu 2013] Mihai Surdeanu. Overview of the TAC2013 Knowledge Base Population Evaluation: English Slot Filling and Temporal Slot Filling, TAC Workshop, 2013
- [Yahya et al. 2014] Mohamed Yahya, Steven Whang, Rahul Gupta, Alon Y. Halevy. ReNoun: Fact Extraction for Nominal Attributes. EMNLP 2014

References

- [Akbik et al, 2013] Propminer: A Workflow for Interactive Information Extraction and. Exploration using Dependency Trees. ACL 2013
- [Atasu et al., 2013] Kubilay Atasu, Raphael Polig, Christoph Hagleitner, Frederick R. Reiss: Hardware-accelerated regular expression matching for high-throughput text analytics. FPL 2013
- [Chiticariu et al., 2011] Laura Chiticariu, Vivian Chu, Sajib Dasgupta, Thilo W. Goetz, Howard Ho, Rajasekar Krishnamurthy, Alexander Lang, Yunyao Li, Bin Liu, Sriram Raghavan, Frederick Reiss, Shivakumar Vaithyanathan, Huaiyu Zhu:The SystemT IDE: an integrated development environment for information extraction rules. SIGMOD Demo 2011
- [Chiticariu et al., 2010b] Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Frederick Reiss, Shivakumar Vaithyanathan: Domain Adaptation of Rule-Based Annotators for Named-Entity Recognition Tasks. EMNLP 2010
- [Cunningham et al., 2002] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan: A framework and graphical development environment for robust NLP tools and applications. ACL 2002
- [Fagin et al., 2013] Ronald Fagin, Benny Kimelfeld, Frederick Reiss, Stijn Vansummeren: Spanners: a formal framework for information extraction. PODS 2013
- [Fagin et al., 2014] Ronald Fagin, Benny Kimelfeld, Frederick Reiss, Stijn Vansummeren: Cleaning inconsistencies in information extraction via prioritized repairs. PODS 2014
- [Freeman et al, 2011] Extreme Extraction --Machine Reading in a Week, EMNLP 2011
- [Gupta and Manning, 2014] SPIED: Stanford Pattern-based Information Extraction and Diagnostics, CoNLL 2014
- [He and Grishman, 2015] ICE: Rapid Information Extraction Customization for NLP Novices, NAACL 2015
- [Hoffman et al. 2015] R. Hoffmann, L. Zettlemoyer, D.S. Weld . Extreme Extraction: Only One Hour per Relation. June 2015
- [Li et al., 2012] Yunyao Li, Laura Chiticariu, Huahai Yang, Frederick Reiss, Arnaldo Carreno-Fuentes: WizIE: A Best Practices Guided Development Environment for Information Extraction. ACL (System Demonstrations) 2012
- [Li et al., 2015] Yunyao Li, Elmer Kim; Marc Touchette; Ramiya Venkatachalam; Hao Wang. VINERy: A Visual IDE for Information Extraction. VLDB Demo, 2015
- [McCord et al., 2012] Michael C. McCord, J. William Murdock, Branimir Boguraev: Deep parsing in Watson. IBM Journal of Research and Development 56(3): 3 (2012)
- [Polig et al., 2014a] Raphael Polig, Kubilay Atasu, Laura Chiticariu, Christoph Hagleitner, H. Peter Hofstee, Frederick R. Reiss, Huaiyu Zhu, Eva Sitaridi: Giving Text Analytics a Boost. IEEE Micro 34(4): 6-14 (2014)
- [Polig et al., 2014b] Raphael Polig, Kubilay Atasu, Heiner Giefers, Laura Chiticariu: Compiling text analytics queries to FPGAs. FPL 2014
- [Sun et al, 2014] Chimera: Large-Scale Classification using Machine Learning, Rules, and Crowdsourcing, VLDB 2014
- [Zhang et el., 2013] Congle Zhang, Tyler Baldwin, Howard Ho, Benny Kimelfeld, Yunyao Li: Adaptive Parser-Centric Text Normalization. ACL (1) 2013