

Data Integration and Large Scale Analysis

05 Entity Linking and Deduplication

Shafaq Siddiqi

Graz University of Technology, Austria



Agenda

- **Motivation and Terminology**
- **Entity Resolution Concepts**
- **Entity Resolution Tools**
- **Example Applications**

Motivation and Terminology

Recap: Corrupted/Inconsistent Data

#1 Heterogeneity of Data Sources

- Update anomalies on denormalized data / eventual consistency
- Changes of app/prep over time (US vs us) → inconsistencies

No Global Keys

#2 Human Error

- Errors in semi-manual data collection, laziness (see default values), bias
- Errors in data labeling (especially if large-scale: crowd workers / users)

#3 Measurement/Processing Errors

- Unreliable HW/SW and measurement equipment (e.g., batteries)
- Harsh environments (temperature, movement) → aging

Uniqueness & duplicates

Contradictions & wrong values

Missing Values

Ref. Integrity

[Credit: Felix Naumann]

| ID | Name | BDay | Age | Sex | Phone | Zip |
|----|-------------|------------|-----|-----|----------|-------|
| 3 | Smith, Jane | 05/06/1975 | 44 | F | 999-9999 | 98120 |
| 3 | John Smith | 38/12/1963 | 55 | M | 867-4511 | 11111 |
| 7 | Jane Smith | 05/06/1975 | 24 | F | 567-3211 | 98120 |

| Zip | City |
|-------|--------------|
| 98120 | San Jose |
| 90001 | Lost Angeles |

Typos

Terminology

[Douglas Burdick, Ronald Fagin, Phokion G. Kolaitis, Lucian Popa, Wang-Chiew Tan: Expressive power of entity-linking frameworks. *J. Comput. Syst. Sci.* 2019]

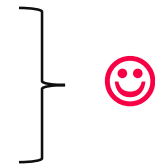


Entity Linking

- “**Entity linking** is the problem of creating links among records representing real-world entities that are related in certain ways.”
- “As an important special case, it includes **entity resolution**, which is the problem of **identifying or linking duplicate entities**”

Other Terminology

- Entity Linking → Entity Linkage, Record Linkage
- Entity Resolution → Data Deduplication, Entity Matching



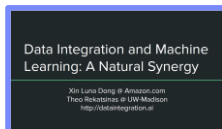
Applications

- Named entity recognition and disambiguation
- Archiving, knowledge bases and graphs
- Recommenders / social networks
- Financial institutions (persons and legal entities)
- Travel agencies, transportation, health care

Barack Obama
 Barack Hussein Obama II
 The **US president (2016)**

Barack and Michelle
are married

Entity Resolution Concepts



[Xin Luna Dong, Theodoros Rekatsinas: Data Integration and Machine Learning: A Natural Synergy. Tutorials, **SIGMOD 2018**, **PVLDB 2018**, **KDD 2019**]



[Sairam Gurajada, Lucian Popa, Kun Qian, Prithviraj Sen: Learning-Based Methods with Human in the Loop for Entity Resolution, Tutorial, **CIKM 2019**]



[Felix Naumann, Ahmad Samiei, John Koumarelas: Master project seminar for Distributed Duplicate Detection. Seminar, **HPI WS 2016**]

Problem Formulation

Entity Resolution

- “Recognizing those records in two files which represent identical persons, objects, or events”
- Given two data sets A and B
- Decide for all pairs of records $a_i - b_j$ in $A \times B$ if match (**link**), no match (**non-link**), or not enough evidence (**possible-link**)

[Ivan Fellegi, Alan Sunter: A Theory for Record Linkage, J. American. Statistical Assoc., pp. 1183-1210, **1969**]



Naïve Deduplication

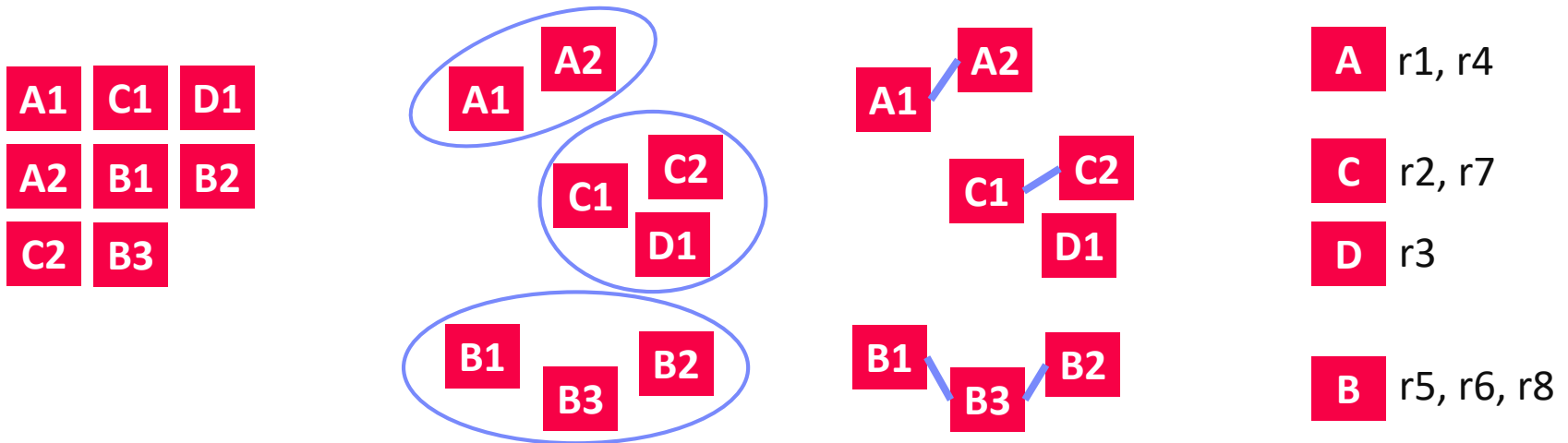
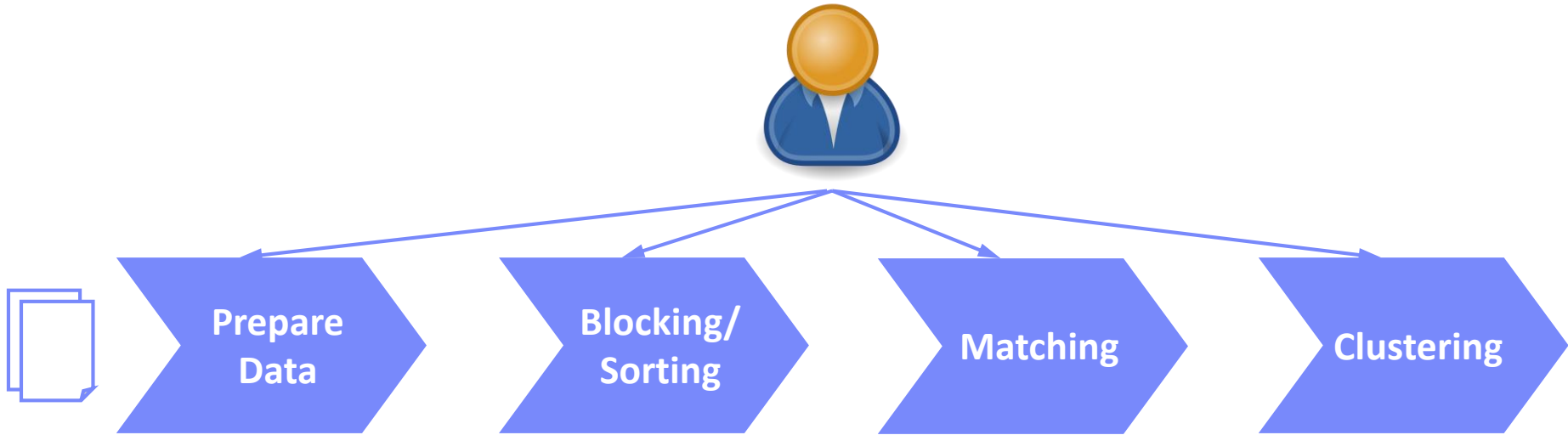
- UNION DISTINCT via hash group-by or sort group-by
- Problem:** only exact matches

| Name | Position | Affiliation | Department |
|-----------------|----------|-------------|------------|
| Shafaq Siddiqui | Lecturer | Sukkur IBA | CS |
| Shafaq Siddiqui | TA | TU Graz | CSBME |

→ Similarity Measures

- Token-based: e.g., Jaccard $J(A,B) = (A \cap B) / (A \cup B)$
- Edit-based: e.g., Levenshtein $lev(A,B) \rightarrow \min(\text{replace, insert, delete})$
- Phonetic similarity (e.g., soundex, metaphone), **Python lib Jellyfish**

Entity Resolution Pipeline



Entity Linking Approaches

[Xin Luna Dong, Theodoros Rekatsinas:
Data Integration and Machine Learning:
A Natural Synergy. **PVLDB 2018**]



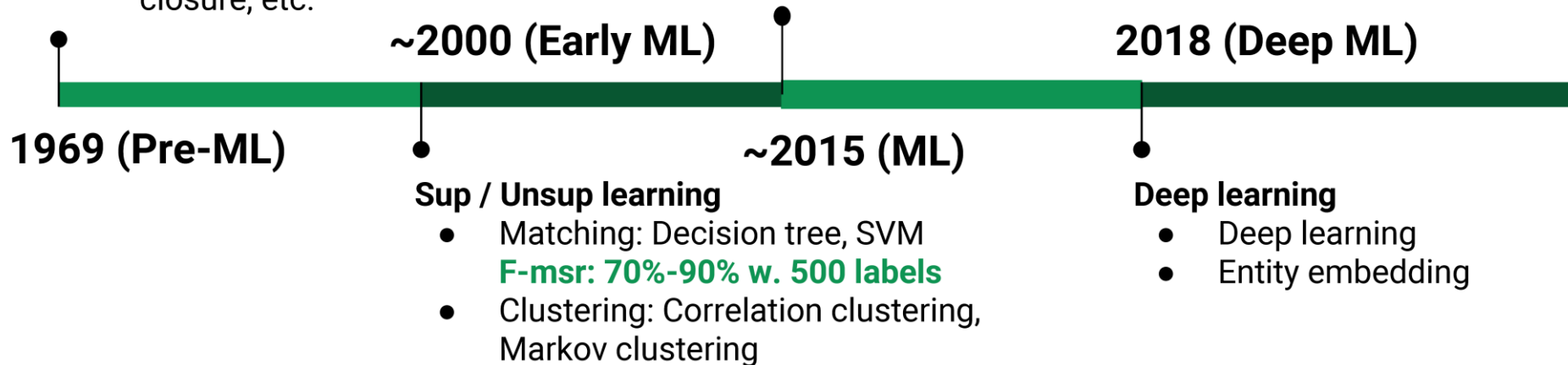
50 Years of Entity Linkage

Rule-based and stats-based

- Blocking: e.g., same name
- Matching: e.g., avg similarity of attribute values
- Clustering: e.g., transitive closure, etc.

Supervised learning

- Random forest for matching
F-msr: >95% w. ~1M labels
- Active learning for blocking & matching
F-msr: 80%-98% w. ~1000 labels



Sup / Unsup learning

- Matching: Decision tree, SVM
F-msr: 70%-90% w. 500 labels
- Clustering: Correlation clustering, Markov clustering

Deep learning

- Deep learning
- Entity embedding

Step 1: Data Preparation

■ #1 Schema Matching and Mapping

- See lecture [04 Schema Matching and Mapping](#)
- Create **homogeneous schema** for comparison
- Split composite attributes

Autonomous,
heterogeneous
systems

■ #2 Normalization

- Removal of special characters and white spaces
- [Stemming](#)
- [Capitalization](#) (to upper/lower)
- Remove redundant works, resolve abbreviations

likes/liked/likely/liking
→ **like**

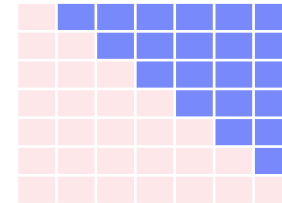
■ #3 Data Cleaning

- See lecture [06 Data Cleaning and Data Fusion](#)
- Correct data corruption and inconsistencies

Step 2: Blocking and Sorting

#1 Naïve All-Pairs

- Brute-force, naïve approach
 $\rightarrow n*(n-1)/2$ pairs \rightarrow **$O(n^2)$ complexity**

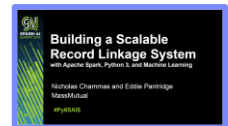


#2 Blocking / Partitioning

- Efficiently create small blocks of similar records for pair-wise matching
- Basic:** equivalent values on selected attributes (name)
- Predicates:** whole field, token field, common integer, same x char start, n-grams
- Hybrid:** disjunctions/conjunctions \rightarrow JR01111
- Blocking Keys:

| | | | | |
|--------------|------------|------------|----|-------|
| John Roberts | 20 Main St | Plainville | MA | 01111 |
| Julia Ray | 32 Main St | Plainville | MA | 01111 |

- Learned: Minimal rule set via greedy algorithms
- \rightarrow **Significant reduction:** 1M records \rightarrow 1T pairs
 \rightarrow 1K partitions w/ 1K records \rightarrow 1G pairs (**1000x**)



[Nicholas Chammas, Eddie Pantridge:
 Building a Scalable Record Linkage
 System, **Spark+AI Summit 2018**]

Step 2: Blocking, cont.

■ #3 Sorted Neighborhood

- Define **sorting keys** (similar to blocking keys)
- Sort records by sorting keys
- Define **sliding window of size m** (e.g., 100) and compute all-pair **matching within sliding window**

■ #4 Blocking via Word Embeddings and LSH/DL

- Compute word/attribute embeddings + tuple embeddings
- **Locality-Sensitive Hashing (LSH)** for blocking
- K hash functions $h(t) \rightarrow k\text{-dim hash-code}$
- L hash tables, each k hash functions

Distributed Tuple Representation

[Muhammad Ebraheem et al: Distributed Representations of Tuples for Entity Resolution. **PVLDB 2018**]



[Saravanan Thirumuruganathan et al. Deep Learning for Blocking in Entity Matching [...]. **PVLDB 2021**]



V %*% H

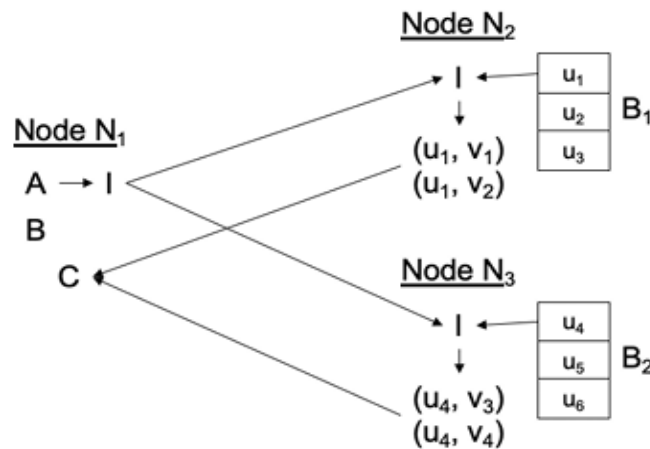
$$\begin{aligned}
 h1 &= [-1, 1, 1], & h2 &= [1, 1, 1], \\
 h3 &= [-1, -1, 1], & h4 &= [-1, 1, -1],
 \end{aligned}$$

$$\begin{aligned}
 v[t1] &= [0.45, 0.8, 0.85] & [1.2, 2.1, -0.4, -0.5] & \rightarrow [1, 1, -1, -1] & \rightarrow [12] \text{ Hash} \\
 v[t2] &= [0.4, 0.85, 0.75] & [1.2, 2.0, -0.5, -0.3] & \rightarrow [1, 1, -1, -1] & \rightarrow [12] \text{ bucket}
 \end{aligned}$$

Step 2: Blocking, cont.

- #5 TF/IDF based blocking
 - top-k blocking scheme

[Derek Paulsen et al: Sparkly: A Simple yet Surprisingly Strong TF/IDF Blocker for Entity Matching. **PVLDB 2023**]



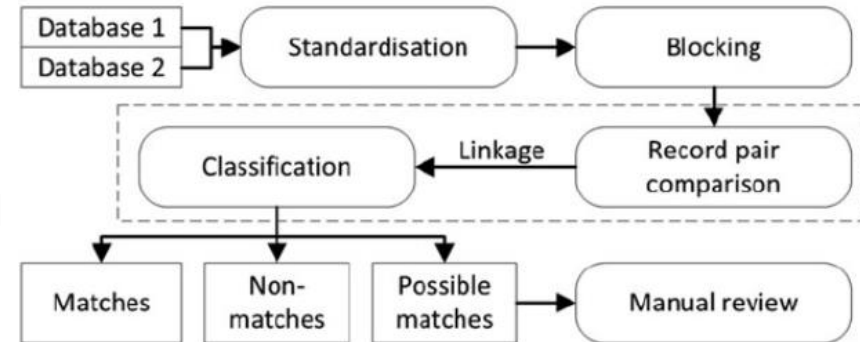
Step 3: Matching

#1 Basic Similarity Measures

- Pick similarity measure $\text{sim}(r, r')$ and thresholds: high θ_h (and low θ_l)
- Record similarity: avg attribute similarity
- **Match:** $\text{sim}(r, r') > \theta_h$ **Non-match:** $\text{sim}(r, r') < \theta_l$
possible match: $\theta_l < \text{sim}(r, r') < \theta_h$

#2 Learned Matchers (Traditional ML)

- **Phase 1:** Model Generation
- **Phase 2:** Model Application
- Selection of samples for labeling (sufficient, suitable, **balanced**)
- **SVM** and **decision trees**, **logistic regression**, **random forest**, XGBoost



[O'Hare, K.et.al. D. P., & A. Jurek-Loughrey,2019]

[Mikhail Bilenko, Raymond J. Mooney: Adaptive duplicate detection using learnable string similarity measures. **KDD 2003**]



[Hanna Köpcke, Andreas Thor, Erhard Rahm: Evaluation of entity resolution approaches on real-world match problems. **PVLDB 2010**]



[Xin Luna Dong: Building a Broad Knowledge Graph for Products. **ICDE 2019**]



Step 3: Matching, cont.

Deep Learning for ER

- Automatic **representation learning** from text (avoid feature engineering)
- Leverage pre-trained **word embeddings for semantics** (no syntactic limitations)

Example DeepER



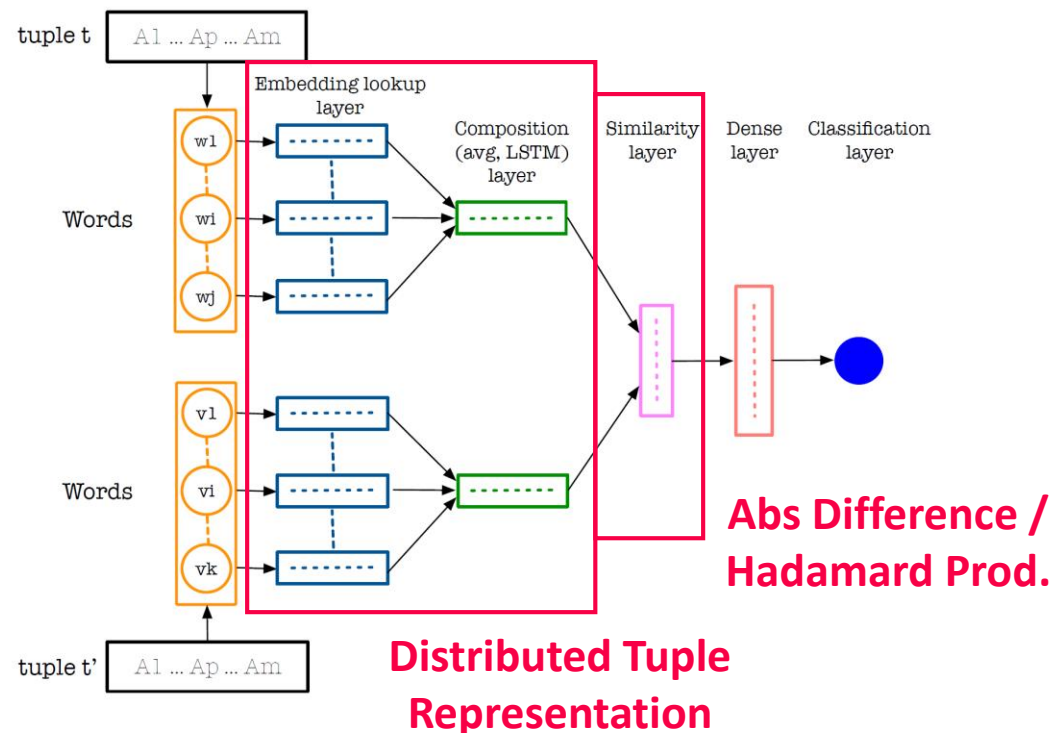
[Muhammad Ebraheem et al: Distributed Representations of Tuples for Entity Resolution. **PVLDB 2018**]

Example Magellan

- DL for text and dirty data



[Sidharth Mudgal et al: Deep Learning for Entity Matching: A Design Space Exploration. **SIGMOD 2018**]

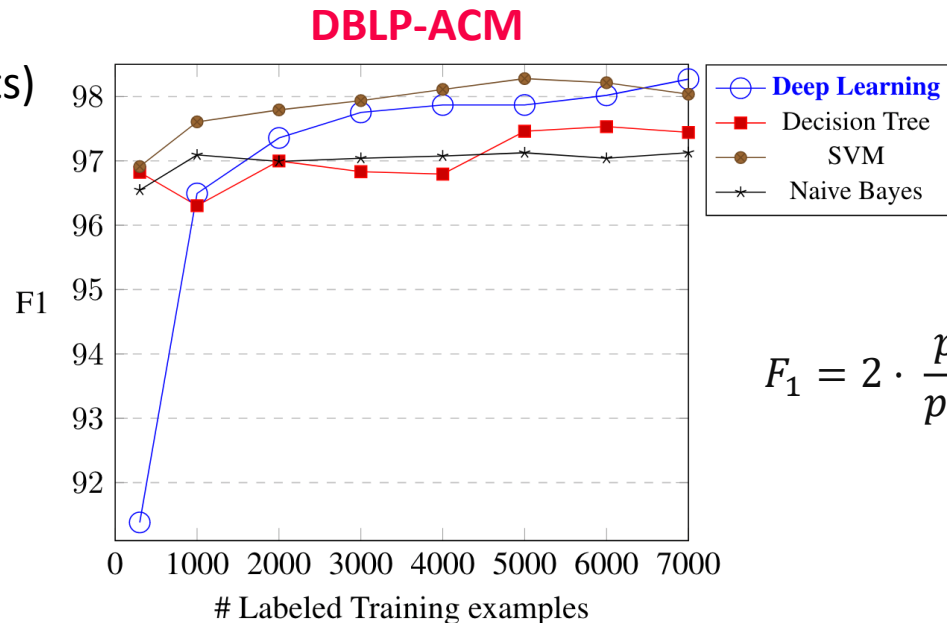


Step 3: Matching, cont.

[Sairam Gurajada, Lucian Popa, Kun Qian, Prithviraj Sen: Learning-Based Methods with Human in the Loop for Entity Resolution, Tutorial, **CIKM 2019**]

■ Labeled Data

- Scarce (experts)
- **Class skew**



$$F_1 = 2 \cdot \frac{\textit{precision} \cdot \textit{recall}}{\textit{precision} + \textit{recall}}$$

➔ Transfer Learning

- Learn model from high-resource ER scenario (w/ regularization)
- Fine-tune using low-resource examples

➔ Active Learning

- Select instances for tuning to min labeling

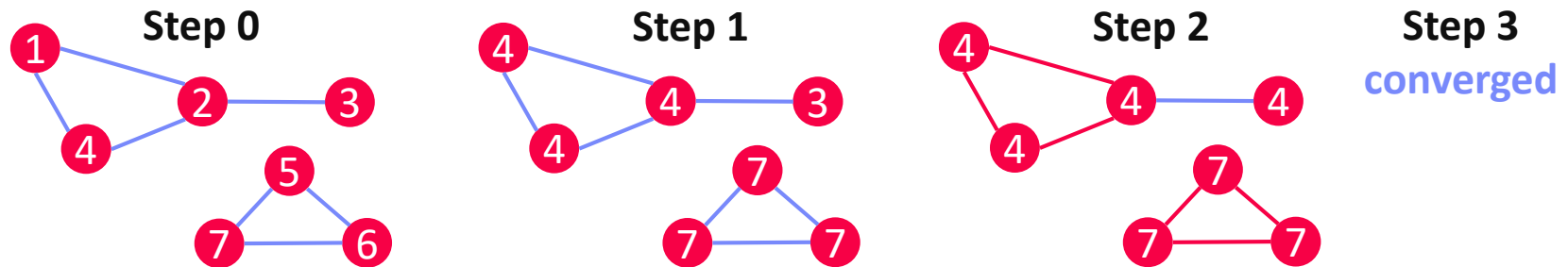
[Jungo Kasai et al: Low-resource Deep Entity Resolution with Transfer and Active Learning. **ACL 2019**]



Step 4: Clustering

Recap: Connected Components

- Determine connected components of a graph (subgraphs of connected nodes)
- Propagate $\max(\text{current}, \text{msgs})$ if \neq current to neighbors, terminate if no msgs



Clustering Approaches

- **Basic:** connected components (transitive closure) w/ edges $\text{sim} > \theta_h$
 → Issues: **big clusters** and **dissimilar records**
- **Correlation clustering:** +/- cuts based on sims → global opt NP-hard
- **Markov clustering:** stochastic flow simulation via random walks

[Oktie Hassanzadeh, Fei Chiang, Renée J. Miller, Hyun Chul Lee: Framework for Evaluating Clustering Algorithms in Duplicate Detection. **PVLDB 2009**]



Incremental Data Deduplication

■ Goals

- Incremental stream of updates
→ previously **computed results obsolete**
- Same or **similar results** AND **significantly faster** than batch computation

[Anja Gruenheid, Xin Luna Dong,
Divesh Srivastava: Incremental
Record Linkage. **PVLDB 2014**]



■ Approach

- End-to-end incremental record linkage for new and changing records
- Incremental maintenance of similarity graph and incremental graph clustering
- Initial graph created by **correlation clustering**
- Greedy update approach in polynomial time
 - Directly connect components from increment ΔG into Q
 - **Merge** of **pairs of clusters** to obtain better result?
 - **Split** of **cluster into two** to obtain better result?
 - **Move** nodes **between two clusters** to obtain better result?

Entity Resolution Tools

Python Dedupe

<https://docs.dedupe.io/en/latest/API-documentation.html>
https://dedupeio.github.io/dedupe-examples/docs/csv_example.html

■ Overview

- **Python library for data deduplication** (entity resolution)
- **By default:** logistic regression matching (and blocking)

■ Example

```

fields = [
    {'field': 'Site name', 'type': 'String'},
    {'field': 'Address', 'type': 'String'}]
deduper = dedupe.Dedupe(fields)
  
```

```
# sample data and active learning
```

```
deduper.sample(data, 15000)
dedupe.consoleLabel(deduper)
```

Do these records refer
 to the same thing?
 (y)es / (n)o /
 (u)nsure / (f)inished

```
# learn blocking rules and pairwise classifier
```

```
deduper.train()
```

```
# Obtain clusters as lists of (RIDs and confidence)
```

```
threshold = deduper.threshold(data, recall_weight=1)
clustered_dupes = deduper.match(data, threshold)
```

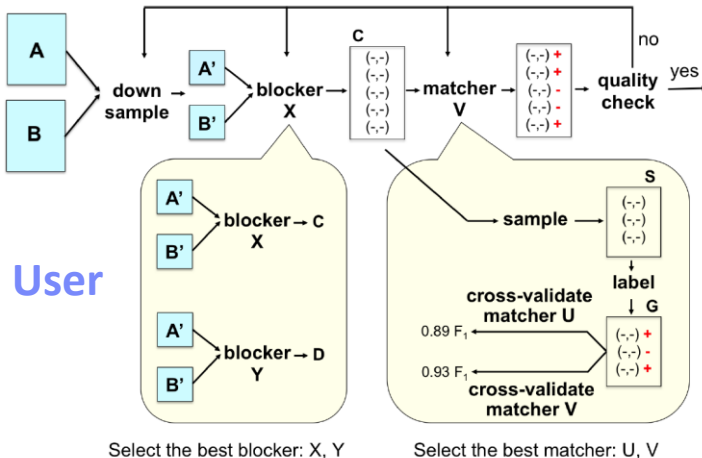
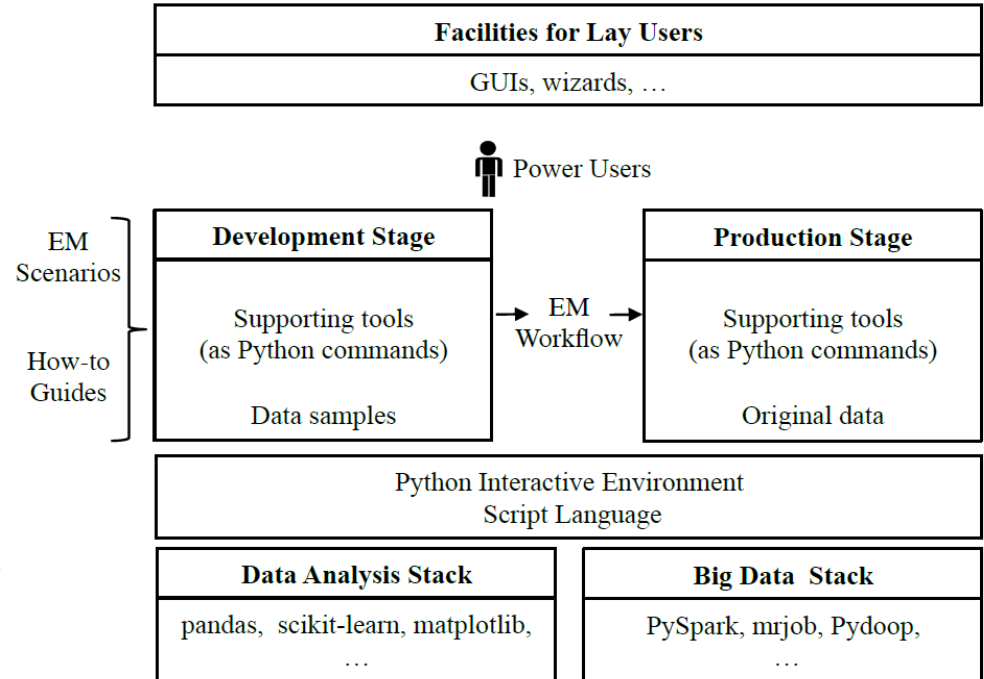
Magellan (UW-Madison)

[Pradap Konda et al.: Magellan: Toward Building Entity Matching Management Systems. **PVLDB 2016**]



System Architecture

- How-to guides for users
- Tools for individual steps of **entire ER pipeline**
- Build on top of existing Python/big data stack
- Scripting environment for power users

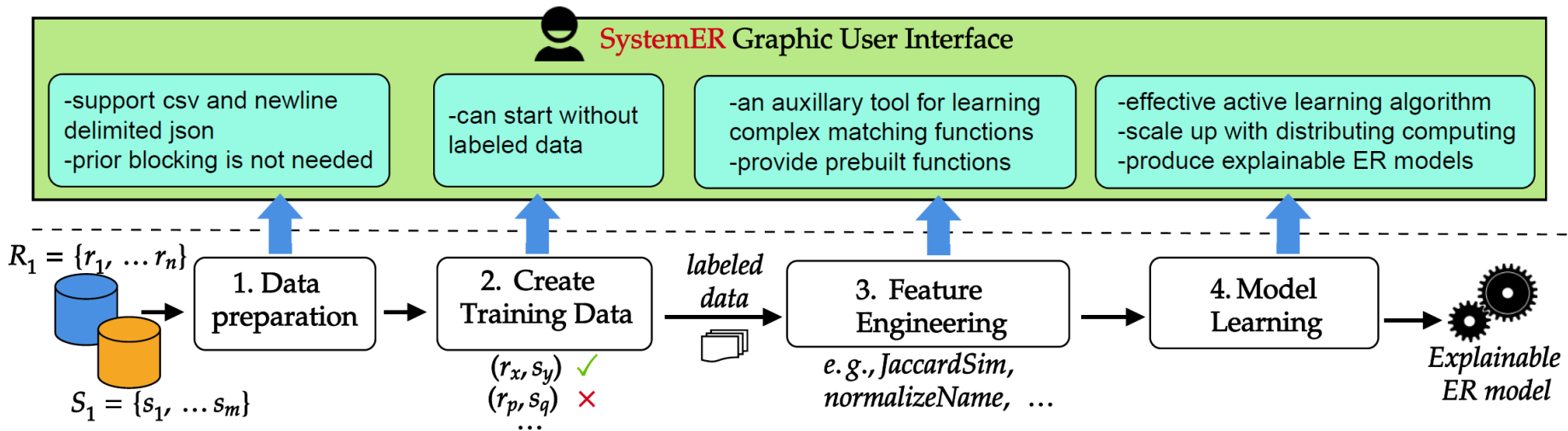


[Yash Govind et al: Entity Matching Meets Data Science: A Progress Report from the Magellan Project. **SIGMOD 2019**]



SystemER (IBM Research – Almaden)

[Kun Qian, Lucian Popa, Prithviraj Sen:
SystemER: A Human-in-the-loop System for
Explainable Entity Resolution. **PVLDB 2019**]



Learns explainable
ER rules (in HIL)

```

DBLP.title = ACM.title
AND DBLP.year = ACM.year
AND jaccardSim(DBLP.authors, ACM.authors) > 0.1
AND jaccardSim(DBLP.venue, ACM.venue) > 0.1
→ SamePaper(DBLP.id, ACM.id)
    
```

[Mauricio A. Hernández, Georgia Koutrika, Rajasekar Krishnamurthy, Lucian Popa, Ryan Wisnesky:
HIL: a high-level scripting language for entity integration. **EDBT 2013**]

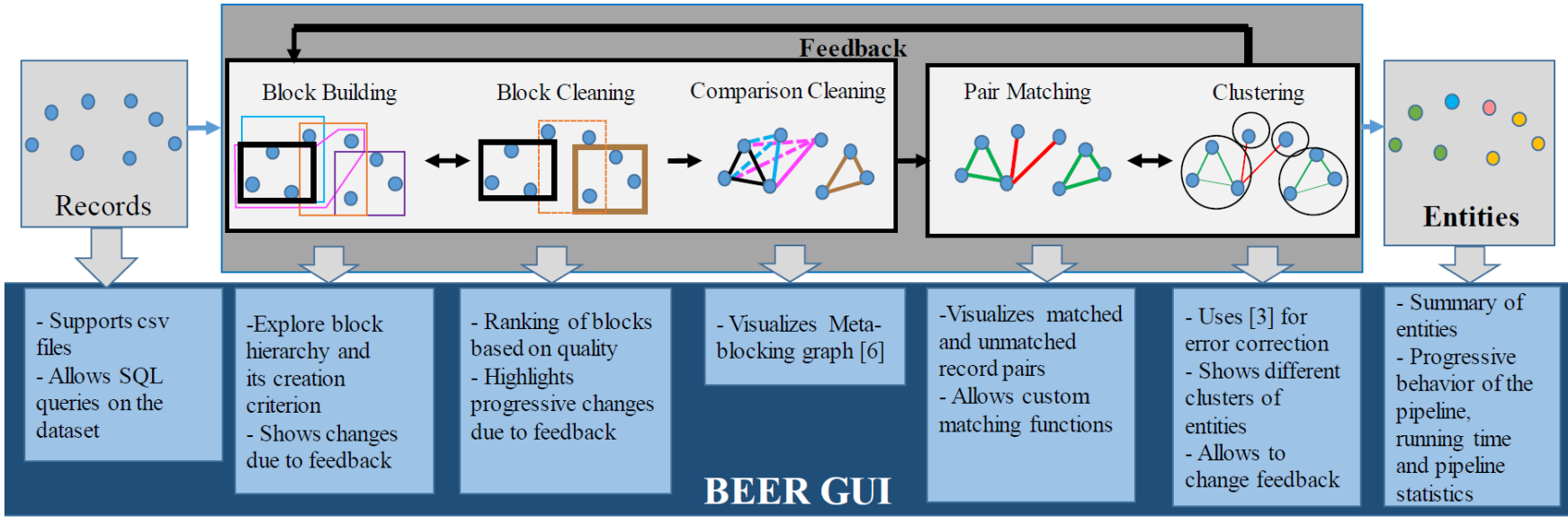


BEER (Blocking for Effective Entity Resolution)

[Sainyam Galhotra, Donatella Firmani, Barna Saha, and Divesh Srivastava: BEER: Blocking for Effective Entity Resolution, SIGMOD 2021]



Feedback after 1% sample



- Supports csv files
- Allows SQL queries on the dataset

- Explore block hierarchy and its creation criterion
- Shows changes due to feedback

- Ranking of blocks based on quality
- Highlights progressive changes due to feedback

- Visualizes Meta-blocking graph [6]

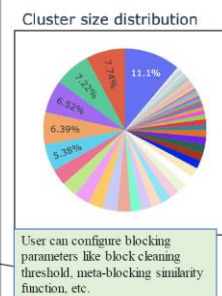
- Visualizes matched and unmatched record pairs
- Allows custom matching functions

- Uses [3] for error correction
- Shows different clusters of entities
- Allows to change feedback

- Summary of entities
- Progressive behavior of the pipeline, running time and pipeline statistics

BEER GUI

The screenshot shows the 'Choose a Dataset' interface. It includes a 'Pre-selected' dropdown set to 'cars', a 'Submit' button, and a text box for a 'SQL Query'. Below this is a table of input datasets with columns for selection, recordid, color, brand, and description. At the bottom, there are options for 'Blocking' (Token based blocking), 'ER' (Hybrid Ordering), and a 'Run ER' button.

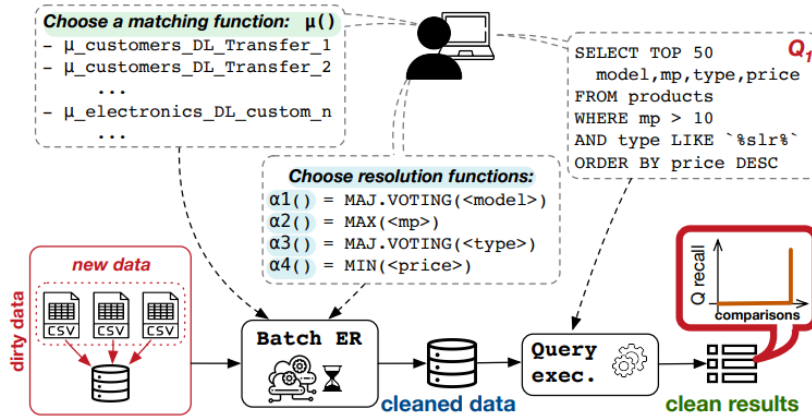


This screenshot shows the 'Configuration of two techniques to be compared' and 'Overall Performance statistics of two techniques' sections. It includes a 'Block Hierarchy' visualization, a 'Comparison of block hierarchies' chart, and a 'Progressive F-score vs Queries' graph. A table compares Technique 1 and Technique 2 on Quality, F-score, and Running Time.

| Quality | Technique 1 | Technique 2 |
|---------------------|-------------|-------------|
| F-score | 0.96 | 0.78 |
| Running Time (mins) | 195 | 130 |

BrewER (Entity Resolution On-Demand)

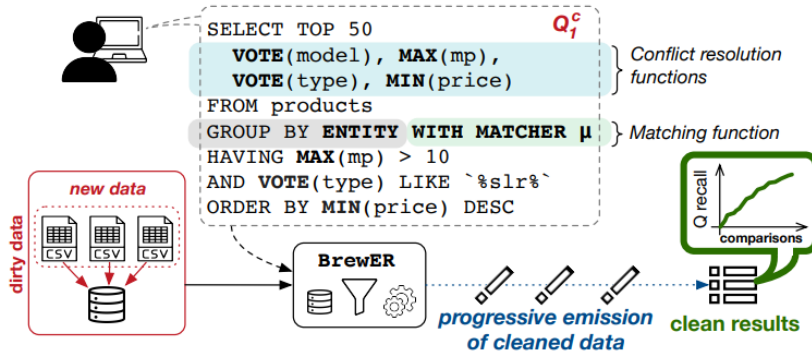
[Giovanni Simonini, Luca Zecchini, Sonia Bergamaschi, Felix Naumann: Entity Resolution On-Demand. **PVLDB 2022**]



(a) The traditional pipeline: the data scientist specifies how to clean the data with ER; once cleaned, she issues the query.

```
SELECT [TOP k]  $\langle \alpha_j(A_j) \rangle$ 
FROM  $\mathcal{D}$ 
[WHERE  $\varphi$ ]
GROUP BY ENTITY WITH MATCHER  $\mu$ 
[HAVING  $\langle \alpha_j(A_j) \rangle$  {LIKE|IN|<|≤|>|≥|=} const}]
[ORDER BY  $\alpha_j(A_j)$  [ASC|DESC]]
```

Figure 2: Query syntax in BrewER.



(b) The ER-on-demand pipeline: the data scientist specifies how to clean the data within the query.

Example Applications

Record Linkage

■ Task: Distributed Entity Resolution on Apache Spark

- Uni Leipzig Benchmarks

■ Example 1: DBLP, ACM, Google Scholar Publications

- (title, authors, venue, year)
- Basic preprocessing via title capitalization, etc
- How about leveraging the linked PDF papers?

https://dbs.uni-leipzig.de/research/projects/object_matching/benchmark_datasets_for_entity_resolution

■ Example 2: Amazon, Google Products

- (name, description, manufacturer, price)
- NLP for matching medium and long descriptions, e.g., word embeddings
- How about leveraging the product images (different angles)

In practice:

multi-modal data, and
feature engineering

■ SIGMOD Programming Contest 2022

- Design blocking scheme for Notebooks specifications dataset

<https://dbgroup.eng.unimore.it/sigmod22contest/task.shtml?content=description>

Data Management – Autograding

- **Plagiarism Detection via Entity Resolution**
 - <https://issues.apache.org/jira/browse/SYSTEMDS-3191>
 - **Data preparation:** file names/properties, runtime, correctness
 - **Blocking:** by programming language, results sets
 - **Matching**
 - Exact matches via basic diff + threshold
 - Code similarity via SotA embeddings
 - **Clustering**
 - Connected components within each block (min sim threshold)

[Fangke Ye et al: MISIM: An
End-to-End Neural Code
Similarity System. **CoRR 2020**
arxiv.org/pdf/2006.05265.pdf]



Summary and Q&A

- **Motivation and Terminology**
- **Entity Resolution Concepts**
- **Entity Resolution Tools**
- **Example Applications**

Fundamental Data
Integration Technique,
w/ lots of applications +
remaining challenges

- **Next Lectures (**Data Integration Architectures**)**
 - **06 Data Cleaning and Data Fusion** [Nov 10]