ACM SIGMOD PROGRAMMING CONTEST 2020

DBGroup@UniMoRe • University of Modena and Reggio Emilia (Runner-Up)

Luca Zecchini

zecchini.luca@libero.it

Advisors: Giovanni Simonini and Sonia Bergamaschi, University of Modena and Reggio Emilia

dbgroup.unimore.it @DBGroupMO

UNIMORE UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

DBGROUP

ACM SIGMOD + PODS 2020 Portland, Oregon USA

Contest Overview

Task: To identify all the records that refer to the same real-world camera model, given a dataset X of camera specifications, extracted from different e-commerce websites.

Evaluation: F-score (harmonic mean of precision and recall) of the solution, submitted as a CSV file C containing all pairs of matching specifications, plus execution time to break ties.

Testing machine configuration: 4 x 3.0 GHz processor, 16 GB main memory, 128 GB storage, Linux operating system.

Dataset Description

Content:

- **Dataset X** of 29.787 specifications (JSON files) from 24 sources (folders).
- **Labelled datasets Y and W** generated from 306 and 908 specifications (Y included in W).

DATASET	COUPLES	MATCHES	NON-MATCHES
Dataset Y	46.665	3.582	43.083
Dataset W	297.651	44.039	253.612

Attributes:

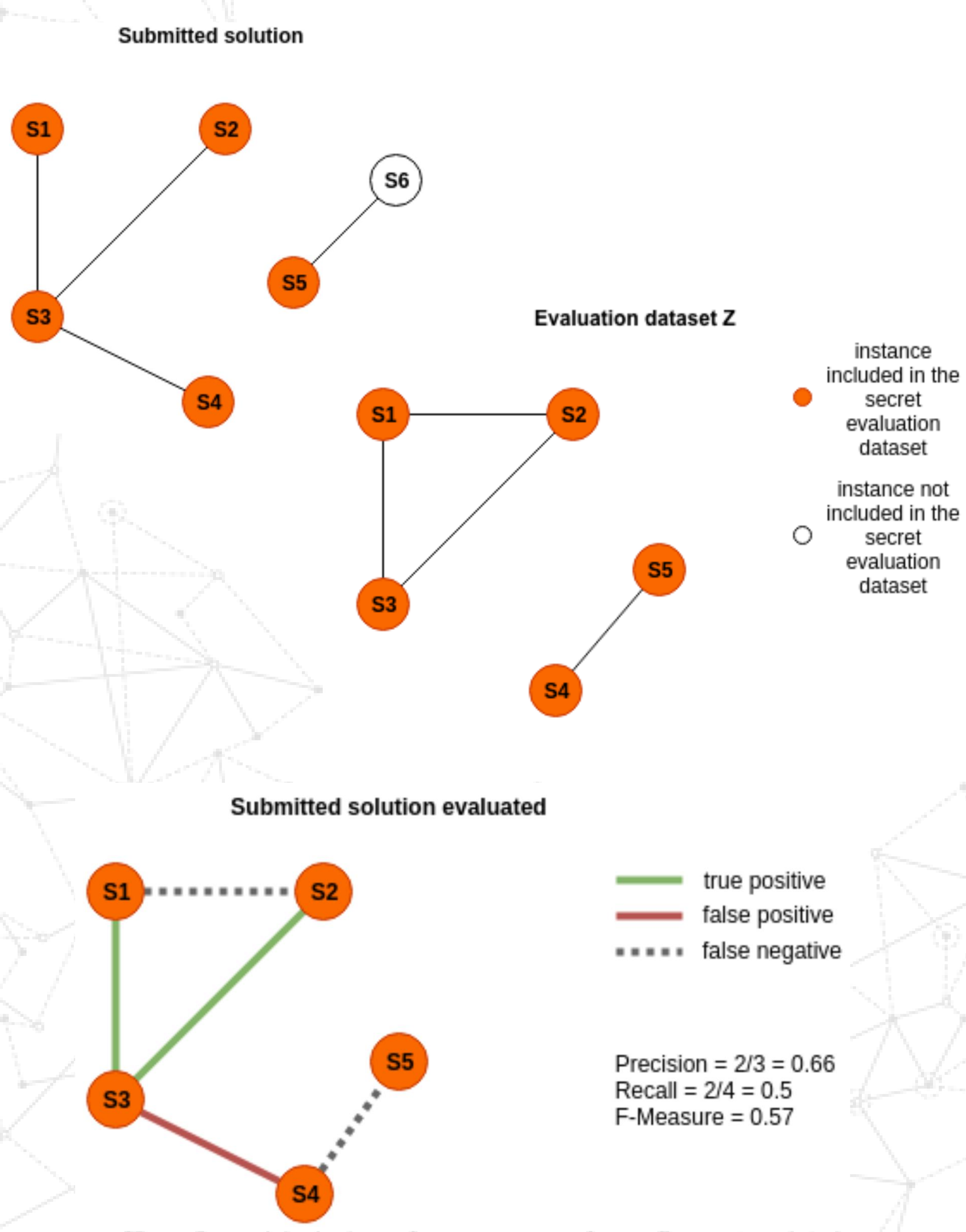
- Attribute *page_title* present in all specifications.
- Other 4.660 attributes with limited distribution and problems of homonymy and synonymy.

Notes:

- Matches can be found even inside the same source.
- **Transitive closure** on the matches.
- Color and possible accessories do not differentiate models.

Evaluation is performed on a secret **evaluation dataset Z**, disjointed from W (they can share nodes but not edges).

An edge is considered for the evaluation only if both nodes are present in Z.



Machine Learning Approach

To solve the problem by using **Magellan** (ML classifiers) and **DeepMatcher** (DL-based solutions), state-of-the-art Python libraries conceived for entity matching.

Matching = cameras with same brand and same model (all other information is superfluous), generally present in *page_title*. → Consider only the attribute *page_title*.

Training, validation, and test sets are generated from labelled dataset (3:1:1 ratio, same distribution of matches and non-matches).

ISSUE with Magellan: F = 0.4
Why? Because of the **variable length** of the attribute

Brand and model are generally located at the beginning of the string. → The attribute can be truncated after the **first *n* words**.

Chosen model is **RNN**, which considers the sequences of words, with *n* = 4.

CLASSIFIER	PRECISION	RECALL	F-SCORE
Decision Tree	92.45%	90.64%	91.54%
Random Forest	93.91%	88.27%	91.00%
RNN	**97%**	**95%**	**96%**
Attention	100%	73%	84%
Hybrid	98%	73%	84%
SIF	46%	31%	37%

Results of Magellan and DeepMatcher on Y with n = 4

CLASSIFIER	PRECISION	RECALL	F-SCORE
Random Forest	98.90%	95.03%	96.93%
SVM	98.29%	93.52%	95.85%
Logistic Regression	96.56%	89.10%	92.68%
Decision Tree	97.72%	87.10%	92.11%
Linear Regression	97.47%	79.27%	87.43%
Naïve Bayes	70.35%	94.13%	80.52%
RNN	**99.59%**	**96.96%**	**98.26%**

Results of Magellan and DeepMatcher on W with n = 4

Moving on dataset X:

- **Blocking** through **inverted index** (one of the 4 words in common) with P = 0.28 and R = 0.99.
- **Blacklisting** to remove most frequent useless words.
- Resolution of **aliases**.

ISSUE: F = 0.47 (P = 0.32, R = 0.85)
FAIL on FP
Why? Because matching is based on little brand-dependent details (e.g., the variation of a single letter or digit); so, the similarity patterns learned by Magellan and DeepMatcher on a few brands and models are not effective on the whole dataset (it is **impossible to generalize** them)

NB: For an erroneous interpretation of the task (disjointed nodes instead of edges), specifications present in the labelled dataset are not included in the candidate set (impact on recall).

Regular Expressions Approach

- Models are often represented with strings containing both letters and digits. → **Regular expressions** are a good candidate for finding them.
- Limited number of brands with a significant distribution. → Manageable through a **list of brands**.

Goal: to reduce the attribute to a **string composed of brand and model**.

An **inverted index** can be built directly on this new string, determining no more the blocks, but directly the **clusters of matching elements** (elements with same brand and same model), guaranteeing **high precision**.

Conceived Solution

Read the content of the JSON file as a dictionary

WORK ON EACH SPECIFICATION

Represent the tuple as a dictionary
- *id* : folder and file names
- *page_title* : no more truncated

Normalize *page_title*
- **Lowercase**
- Replace **punctuation** with an idle character and remove dashes
- Manage **aliases** (e.g., *fuji* → *fujifilm*)

Retrieve the **brand** by using a list of brands

Brand-based management of **prefixes and suffixes** (concatenation with model name)

Retrieve the **model** through a regular expression
- Brand-based management of only alphabetical or only numeric models
- Brand-based management of words which contain both letters and digits without being models
- Exclusion of the words representing measures

Further brand-based operations on the model
- Remove specific prefixes/suffixes (e.g., color suffixes)
- Manage equivalences among models
- Retrieve additional elements (e.g., *mark*)

Brand and model were detected?
YES → merge them in a new attribute *brand_n_model* and store in the list *solved*
NO → store in the list *unsolved*

FIND MATCHES

Generate **clusters of matching elements** through an **inverted index**
solved → on the whole attribute *brand_n_model*
unsolved → on the whole attribute *page_title*

Compute **Cartesian product** inside each cluster
Remove self products and reflexive pairs

Save matching pairs in the CSV file C

All the patterns and the characteristics typical of each brand were **manually extracted from the data**.

Results

FINAL F-MEASURE = 0.99
(P = 0.99, R = 0.98)

The final F-measure was the same as the other 4 finalist teams, and the tie was broken according to the execution time, certifying the proposed one as the **second-best performing solution**.

Conclusions

This experience was useful to show the limits that current state-of-the-art machine learning and deep learning systems for entity matching still present.

These systems are able to achieve good results when matches can be identified by means of **similarity-based features**, but in a lot of real-world scenarios matching is based on **little variations** which make the generalization of the learned patterns on the entire dataset impossible.

This task requires carefully designed rules, which these systems are not yet able to synthetize, so **a lot of human work is needed to wrangle the data**.

For the future, I plan to work on the **automation of deep learning data wrangling techniques**, trying to make them more sensitive to these variations and able to manage also critical situations like this one.