Record linkage of banks and municipalities through multiple criteria and neural networks

Antonio Maratea [Corresp., 1] , **Angelo Ciaramella** [1] , **Giuseppe Pio Cianci** [1]

[1] Department of Science and Technologies, University of Naples "Parthenope", Naples, Italy

Corresponding Author: Antonio Maratea
Email address: antonio.maratea@uniparthenope.it

Record linkage aims to identify records from multiple data sources that refer to the same entity of the real world. It is a well known data quality process studied since the second half of the last century, with an established pipeline and a rich literature of case studies mainly covering census, administrative or health domains. In this paper a method to recognize matching records from real municipalities and banks through multiple similarity criteria and a Neural Network classifier is proposed: starting from a labeled subset of the available data, first several similarity measures are combined and weighted to build a feature vector, then a Multi-Layer Perceptron (MLP) network is trained and tested to find matching pairs. For validation, 7 real datasets have been used (3 from banks and 4 from municipalities), purposely chosen in the same geographical area to increase the probability of matches. The training only involved 2 municipalities, while testing involved all sources (municipalities vs municipalities, banks vs banks and and municipalities vs banks). The proposed method scored remarkable results in terms of both precision and recall, clearly outperforming threshold-based competitors.

Record Linkage of Banks and Municipalities through Multiple Criteria and Neural Networks

Antonio Maratea[1], Angelo Ciaramella[1], and Giuseppe Pio Cianci[1]

[1]Department of Science and Technologies, University of Naples "Parthenope", 80143 Naples, Italy,

Corresponding author:

Antonio Maratea[1]

Email address: antonio.maratea@uniparthenope.it

ABSTRACT

Record Linkage aims to identify records from multiple data sources that refer to the same entity of the real world. It is a well known data quality process studied since the second half of the last century, with an established pipeline and a rich literature of case studies mainly covering census, administrative or health domains.

In this paper a method to recognize matching records from real municipalities and banks through multiple similarity criteria and a Neural Network classifier is proposed: starting from a labeled subset of the available data, first several similarity measures are combined and weighted to build a feature vector, then a Multi-Layer Perceptron (MLP) network is trained and tested to find matching pairs.

For validation, 7 real datasets have been used (3 from banks and 4 from municipalities), purposely chosen in the same geographical area to increase the probability of matches. The training only involved 2 municipalities, while testing involved all sources (municipalities vs municipalities, banks vs banks and and municipalities vs banks).

The proposed method scored remarkable results in terms of both precision and recall, clearly outperforming threshold-based competitors.

INTRODUCTION

Record Linkage (RL from now on, also called *entity resolution* or *entity matching*), is the process of identifying records coming from different sources that refer to the same real-world entity. Similarly, *Record Deduplication* (RD from now on) is the process of identifying duplicate records, where the same entity of the real word has been entered multiple times in the same data source. The main difference between RL and RD is that records coming from different sources may lack a common identifier and schema (please refer to Christen (2012); Zhang (2011) for a general discussion of related issues).

The pairing of records or the identification of duplicates is a statistically challenging and computationally demanding problem: scarce data quality control results in errors, noise, missing values, omissions, ambiguities and even lies in the data, that combined with the differences in database schemas and regional conventions when the number of nationalities grows, results in a daunting number of factors to be considered. The brute-force comparison All Versus All (AVA) of the records from different sources to discover occasional matches is unfeasible even for a modest volume of data. Notwithstanding these difficulties, RL is known and has been treated since the second half of the last century, with a rich literature in different domains (surveys in the context of data warehousing and for different phases can be found in Brizan and Tansel (2006); Christen (2011)) and an established implementation process (see section *Phases of RL*).

While data quality is related to and borrows techniques from RL and RD, it acts as an *a priori* filter, trying to prevent the insurgence of duplicate records with a proper control process *before* the data consolidation. RL and RD, on the contrary, act as *a posteriori* filters, trying to detect duplicate records and clean the data *after* consolidation. To the data analyst, they represent a required pre-processing step when it is legitimate to assume that the data to be analyzed are corrupted by consequence of a scarce

quality control during acquisition, or come from heterogeneous sources with respect to time, place or context (please refer to Batini and Scannapieco (2006) for a general discussion of related issues).

The traditional approach to RL and RD is probabilistic, or rule-based, and only relatively recently Machine Learning alternatives have emerged (see Zhao and Ram (2005)). The probabilistic approach is grounded on the estimation of probabilities of match and on thresholds for the similarity scores; the rule-based approach tries to explicitly model the knowledge of domain experts; the Machine Learning approach, on the contrary, relies only on data and can be cost-sensitive, supervised, unsupervised, or semi-supervised.

In the following a multiple-criteria feature vector and a supervised classifier are proposed in the *classification* phase of the RL process, outperforming classic crude threshold-based methods and producing remarkable results.

The paper is organized as follows: in Section *Background* the phases of a traditional RL process and the related literature on Neural Networks are sketched; in Section *Materials and methods* the proposed method is presented in detail; in Section *Experiments* the used data and the experiments are fully described; finally, in Section *Conclusions*, main conclusions are drawn.

BACKGROUND

First the currently established phases of a RL process will be outlined, then the recent literature on Neural Networks applied to RL will be summarized.

Phases of RL

The RL of two sources generally involves five independent phases, each exploiting different techniques, criteria, and algorithms Christen (2012):

1. ***Data pre-processing***: the two sources are cleaned and normalized to ensure that all the data have the same format and are as much as possible error free;

2. ***Indexing***: to save time, the record pairs that are evidently different are filtered out from the comparisons through clustering, sorting or blocking. Record Linkage is exponential in nature, as each record from the first source should be compared with all the records from the other sources, hence indexing is critical for performance;

3. ***Comparison***: only the record pairs within the same block (or cluster) are actually compared one by one, using multiple similarity criteria that are conveyed into a similarity vector;

4. ***Classification***: the similarity vector obtained from each pair of records within the same block (or cluster) is classified into one of three groups:

   - (M) – *matches*, that is pairs that do refer to the same entity of the real world;

   - (NM) – *non-matches*, that is pairs that do not refer to the same entity of the real world;

   - (PM) – *potential-matches* or ambiguous pairs, that is pairs that can't be classified with sufficient precision or reliability;

5. ***Evaluation***: the classified pairs are reviewed for final labeling. Specifically, the PM class is subject to a *clerical review*, where domain experts decide for each ambiguous pair if it is actually a match or not.

Related work

Being a possible consequence of both a poor data quality process than natural differences in the data evolving over time, RL has been found in countless domains and tackled using many techniques and approaches. The related literature can be broadly split into *census*, *administrative* or *health* related applications, with a majority of works exploiting probabilistic methods. Artificial Neural Networks (ANN from now on) as classifiers are less in number and relatively recent.

Some of the main issues of administrative data linkage are privacy and confidentiality requirements and to the absence of common identifiers (much like health data. Please refer to Harron et al. (2017)

92   for a discussion and to Vatsalan et al. (2013) for a taxonomy). If from one side epidemiological studies
93   linking census or administrative data to diseases are pretty common, from the other side the specific case
94   of linking census with financial data at the individual level is rare, if not absent, in the RL literature. The
95   reason is that medical data are mostly held by public agencies that have a public interest in sharing their
96   data and supporting medical research, while banks are private companies that keep their data safe and
97   secure on their servers, having little interest in sharing their analysis.

98   Recent literature on Machine Learning applied to RL includes Aiken et al. (2019), that compares
99   probabilistic, stochastic and machine learning approaches, showing that supervised methods outperform
100 unsupervised ones; Dong and Rekatsinas (2018), that surveys state-of-the-art data integration solutions
101 based on Machine Learning with a discussion of open research challenges;Kasai et al. (2019), that
102 leverages Deep Learning in a combination of Transfer and Active Learning aiming to save labeled data up
103 to an order of magnitude; Di Cicco et al. (2019), that presents an attempt of explainable Deep Learning
104 exploiting LIME, a popular tool for prediction explanations in classification; Hou et al. (2019), that
105 propose a paradigm called "gradual machine learning"where data are labeled automatically through
106 iterative factor graph inference, starting with the easiest instances and going up to the hardest ones.

107 A survey of the state of the art is beyond the scope of this paper, so this section will focus on ANN
108 classifiers applied to classification in RL, in line with the recent explosion of ANN related literature
109 (please see Maratea and Ferone (2018)).

110 One of the first attempts is apparently due to Zhao and Ram (2005), that proposed a set of ensemble
111 learning methods combining multiple base classifiers, including a backpropagation ANN. Records are
112 compared through various similarity measures and classifiers are merged through Bagging, Boosting,
113 Stacking, Cascading or cross-validated committees. Using $25,000$ records ($20,000$ non-matching exam-
114 ples and $5,000$ matching examples) from an application service provider (ASP) for the airline industry, the
115 authors tried to identify the same customer that reserved different flights with different airline companies,
116 reaching over 99% of accuracy. Authors warn the reader on the limited generalization of the experiments
117 due to the "somewhat balanced"data used.

118 More recently, Wilson (2011) showed on the base of genealogical databases that the results obtainable
119 from the probabilistic RL are easily improvable through one of the various available Machine Learning
120 or ANN techniques, and that even a simple single layer perceptron network tends to outclassify the
121 probabilistic approaches, reaching 95% of precision and 97.2% of recall compared to 72.5% precision
122 and 91% recall of the probabilistic methods.

123 A singular case is the work Siegert et al. (2016) in the linkage of epidemiological cancer registries data
124 previously pseudo-randomized through hashing and encrypted for privacy reasons. Features are extracted
125 from the obscured data and used as they were a new coding of the records, then the classification is
126 performed on these coded data. Similarly to Zhao and Ram (2005), multiple classifiers and ensembles are
127 tested, with many aggregation functions. Approximately 35,000 match pairs and 38,000 of not matches
128 for a total of 73,000 pairs of records were manually classified from the North Rhine-Westphalia cancer
129 registry in Germany and used as training-set for the supervised learning classifiers. The proposed ANN is
130 structured with a single hidden layer of 60 neurons and a sigmoidal activation function. Among the three
131 classifiers used, the one based on ANN provided better results in both precision and recall terms, reaching
132 a 95.2% precision and 94.1% recall. Even in this case, data are artificially balanced.

133 Subitha and Punitha (2014) propose the use of Elman's neural networks to pair the medical records
134 collected by hand from different hospitals and departments, achieving an accuracy of 85% and a recall
135 of 98% with respect to fuzzy decision trees (75% and 95%) and decision trees (79% and 96%). The
136 comparing phase was performed using only the normalized Levenshtein distance as the similarity criteria.
137 The Elman Neural Network (ENN) is a particular type of neural network in which a layer of neurons
138 called "context units" connected with a fixed weight of 1.0 both to the input and to the output of the
139 hidden layer is added. In the context units, a copy of the last output of the hidden layer is saved to be used
140 for subsequent inputs. In this way the network can maintain a sort of state, allowing tasks such as the
141 prediction of sequences that go beyond the power of a standard multilayer perceptron.

142 Mudgal et al. (2018) present a general framework for the application of Deep Neural Networks (DNN
143 from now on) to RL, stressing connections with Natural Language Processing: three type of problems
144 are highlighted: structured, textual and dirty RL. Their goal is to illustrate the benefits and limitations of
145 DL when applied to RL. An empirical evaluation based on 4 models and 11 datasets is presented to show
146 when DNN outperform traditional approaches, obtaining a relevant improvement in case of unstructured

data.

Kooli et al. (2018) report a DNN application for RL using Relevant Word Extraction and Word Embedding instead of the classic calculation of the similarity vector from record pairs. Three architectures are compared: Multi-Layer Perceptron (MLP), Long Short Term Memory networks (LSTM) and Convolutional Neural Networks (CNN) on poorly structured scientific publications databases by getting very good results and showing improvements compared to classical similarity-based approaches. The authors point out that their approach is fully automatic unlike a previous job of Gottapu et al. (2016) that also uses DNN but in a human/machine hybrid fashion, by facilitating the manual categorization in a crowd-sourcing methodology, by proposing to the operator a list of possible matches. Ebraheem et al. (2018) used unidirectional and bidirectional recurrent neural networks (RNNs) with long short term memory (LSTM) hidden units to convert each tuple to a distributed representation, which was the base for subsequent classification. Experiments on 7 Datasets from different domains showed promising results.

MATERIALS AND METHODS

A block representation of the proposed method is depicted in Fig. 1: it involves the *comparing* and *classifying* phases of RL (see section *Phases of RL*): it takes as input pairs of records and classifies them into "match" (M) or "non-match" (NM) through an ANN.

The features of each candidate record pair are extracted by comparing each corresponding attribute with multiple similarity functions, resulting in a similarity vector (see below for more details). It is this similarity vector that is used for the subsequent classification of the candidate pair.
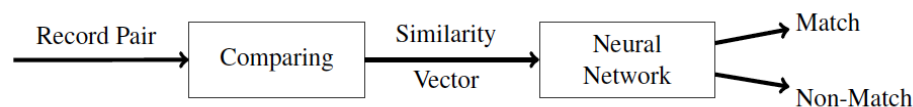
Figure 1. Block diagram of the proposed method.

In order to have comparable tests, all previous phases use the same methods and parameters.

Data sources

Real data from separate municipal and banking databases have been gathered, chosen in the same geographical area to increase the probability of match (see Table 1). For both municipal and banking databases, each person is identified through the fiscal code (alias social security number, alias insurance number). Being a natural key and a reliable field, this common identifier was leveraged to create, through joins, a gold standard for training and evaluation purposes: if in a pair, both records had the same value for the fiscal code, they are considered a true-match.

The fiscal code is derivable from first name, last name, date and place of birth plus some special codes. Surprisingly, clerical reviews showed that in rare cases even the fiscal code presents some errors, i.e. does not correspond to the value derivable from the other fields of the record. This can lead to rare cases in which a pair of records is correctly classified as a match by the classifier but results indeed a false positive for the evaluation metric. For this reason, all the figures of merit presented hereafter have to be considered conservatively estimated.

Table 1. used databases flanked by their size.

Database name	type	Alias	# record
ANAG_UNO	Municipal registry	A	42698
ANAG_DUE	Municipal registry	B	45100
ANAG_TRE	Municipal registry	C	42559
ANAG_QUATTRO	Municipal registry	D	57434
BCC_UNO	Bank details	X	1052737
BCC_DUE	Bank details	Y	101651
BCC_TRE	Bank details	Z	93179

180 **Relevant attributes**

181 As a first step, it is necessary to select a subset of relevant and shared attributes between the municipal
182 and banking databases to be used in the RL process. The selected attributes are shown in Table 2.

183   Special attention should be paid to attributes *address*, *zip code* and *province* because they have a
184 different meaning, representing the home address in municipalities databases and the residence address in
185 banking databases. These values are often the same but do not always coincide.

Table 2. Attributes in common between banking and municipal databases.

Attribute	Municipal Column	Bank Column
SURNAME	COGNOME	INTESTAZIONE_A
NAME	NOME	INTESTAZIONE_B
SEX	SESSO	SESSO
BIRTH STATE	STATO_NASCITA	NAZIONALITA
BIRTH DATE	DAT_NASCITA	DATA_NASCITA
BIRTH PLACE	COM_NAS	DESCR_COM_NASC
BIRTH PROVINCE	PROV_COM_NAS	PROVINCIA_NASC
ADDRESS	VIA_DOM NUMERO_CIVICO_DOM	VIA_RES
ZIP CODE	CAP_DOM	CAP_RES
PROVINCE	PROV_COM_DOM	PROV_RES
TELEPHONE	TELEFONO	NUMERO_TELEFONO

186   Each attribute is cleaned, standardized and normalized through multiple transformations, turning
187 everything lower case, removing special symbols, punctuation, repeated spaces, non-alphabetic characters
188 and finally normalizing accented letters.

189 **Indexing**

190 Indexing is performed with *multiple blocking indexing*, combining the results obtained from 3 blocking
191 keys.

192   1. Blocking key 1: $\langle$SURNAME,NAME,B.DATE$\rangle$, using the Double Metaphone phonetic algorithm
193   for the key comparison;

194   2. Blocking key 2: $\langle$B.DATE,B.PLACE,B.PROVINCE$\rangle$, using the Double Soundex phonetic algo-
195   rithm for the key comparison;

196   3. Blocking key 3: $\langle$SURNAME,NAME,SEX,B.DATE, B.PLACE,B.PROVINCE$\rangle$ using for the key
197   comparison: the last 3 character suffix for name and surname; the first 4 character prefix for the
198   birth place and province; and the year and month for the birth date.

199 These blocking criteria were determined through experimental test and chosen to maximize the pair
200 completeness as much as possible by keeping the number of candidate record pairs generated low, to
201 reduce the execution time in view of the numerous tests to be performed.

202 **Comparing**

203 In the comparison phase, the similarity of each pair of records (A, B) is measured using a set $\mathcal{S}$ of
204 string-based similarity functions on the corresponding attributes (name of the first record with name of the
205 second record, surname of the first record with surname of the second record and so on). Each comparison
206 function has a normalized output in the interval $[0, 1]$. Where 0 indicates maximum dissimilarity and 1
207 indicates maximum similarity (please see Navarro (2001); Christen (2012)).

208   The set $\mathcal{S}$ of comparison functions used is listed below:

209   (a) Jaro-Winkler,   (d) Jaccard,   (g) Trigrams,
    (b) Levenshtein,   (e) Siresen-Dice,   (h) Exact.
    (c) Cosine,   (f) Bigrams,

$$\text{similarity}(A,B) = (0.833, 0.571, 0.288, 0.961, 0.666, 0.777, 0.774, 0.962,$$
$$1.000, 1.000, 1.000, 1.000, 0.333, 0.333, 0.000, 0.333,$$
$$1.000, 1.000, 1.000, 1.000, ..., 0.000, 0.000, 0.000, 0.000)$$
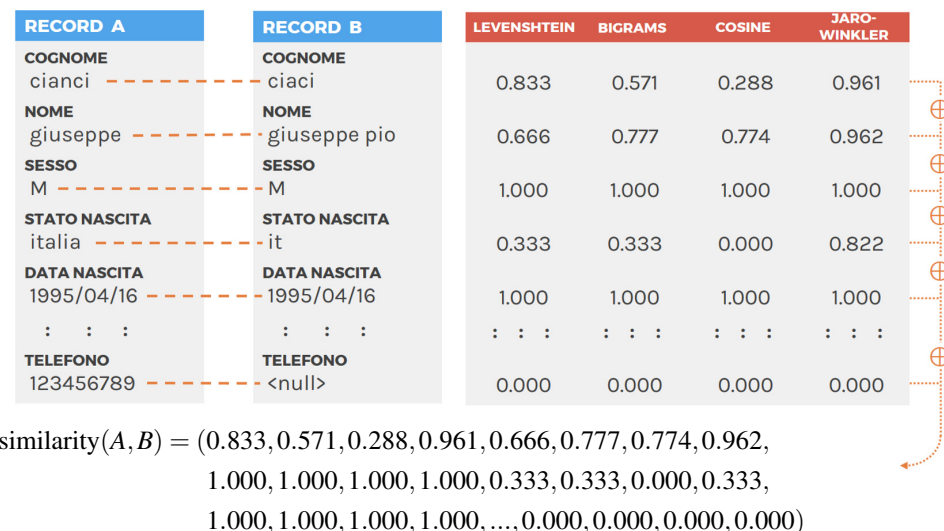
Figure 2. Example of the similarity vector obtained comparing two records using four similarity functions (please see Table 2 for attribute mapping).

Each one of the corresponding attributes pairs (a_i, b_i) is compared using all the function in the set and the resulting values are then chained in a similarity vector **s**.

$$\mathbf{s} = \text{sim}_\mathcal{S}(a_1, b_1) \oplus \text{sim}_\mathcal{S}(a_2, b_2) \oplus \text{sim}_\mathcal{S}(a_3, b_3) \oplus \dots$$

Figure 2 shows an example of this procedure. At the end, each pair of records will be associated with a similarity vector to be used for classification.

Classifying

The ANN used for the classification is shown in the Fig. 3, it's a fully connected MultiLayer Perceptron network (MLP from now on), with two hidden layers in a pyramidal structure, a *ReLu* activation function:

$$\text{ReLu}(x) = x^+ = \max(0, x)$$

and a *Softmax cross entropy* loss function:

$$\mathcal{L}(y, \hat{y}) = \sum_i H(y_i, \hat{y}_i) = \sum_i y_i \cdot \log \hat{y}_i$$

The optimal number of neurons and the size of the layers have been determined through iterative optimization on experimental tests. The final architecture is:

- *Input Layer*: a layer of as many neurons as the components of the similarity vectors to be classified;

- *Hidden Layer*: two layers to form a pyramidal structure, the first with 8 and the second with 4 neurons;

- *Output Layer*: a layer of two neurons, one for the class of the matches (M) and the other for the non-matches (NM).

For the initialization the `glorot_uniform_initializer` (aka Xavier uniform initializer) was used. With this random initalization of the MLP parameters no relevant changes in the performance were noted over different runs.

The network was trained using the optimizer Adam optimizer by applying a L1 regularization to avoid overfitting.
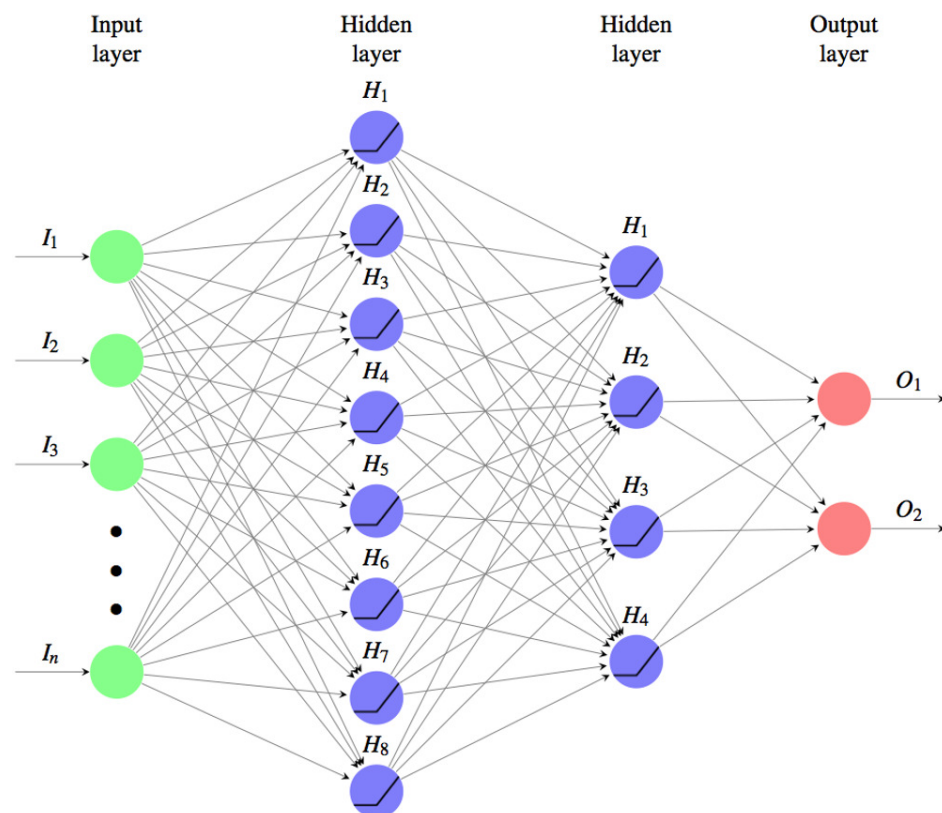
Figure 3. Used fully connected MLP architetture for the classifyng phase. Where the input layer as many neurons as the components of the similarity vectors to be classified, There are two hidden layers one of size 8 and the other of 4 and the output layer is composed of two units, one for the class of the matches (M) and the other for the non-matches (NM). The activation function used is the ReLu and the loss function is the Softmax cross entropy. The weights are initalized with Xavier uniform initializer and the training was performed using the Adam Optimization optimizer by applying a L1 regularization.

Training data-set generation

The training data-set, in the format $(feature, label)$ is generated based on the candidate record pairs identified in the indexing phase between databases A and B, where:

- *feature*: is the similarity vector obtained comparing the records of the pair;

- *label*: is true-match (M) or true-not match (NM), according to the gold standard.

The built training data-set contains 10876 samples, 1567 of which are labeled as true-match (M) and 9300 as true-non-match (NM).

Since non-matching record pairs are more than matching ones, the training data are moderately imbalanced (they are in the same order of magnitude). Over-sampling techniques like ADASYN He et al. (2008) and based on SMOTE, such as SMOTENC Bowyer et al. (2011) SMOTENN Batista et al. (2004), SMOTETomek Batista et al. (2003) have been tested, without any significant improvement, so oversampling was skipped from final experiments.

EXPERIMENTS

For each test the same set of attributes, pre-processing and indexing techniques have been used in order to focus on the comparing and classifying phases.

Fig. 4 shows the starting condition for the tests in order to have a reference on the number of true-match and of pairs identified by the indexing between the various coupled databases.
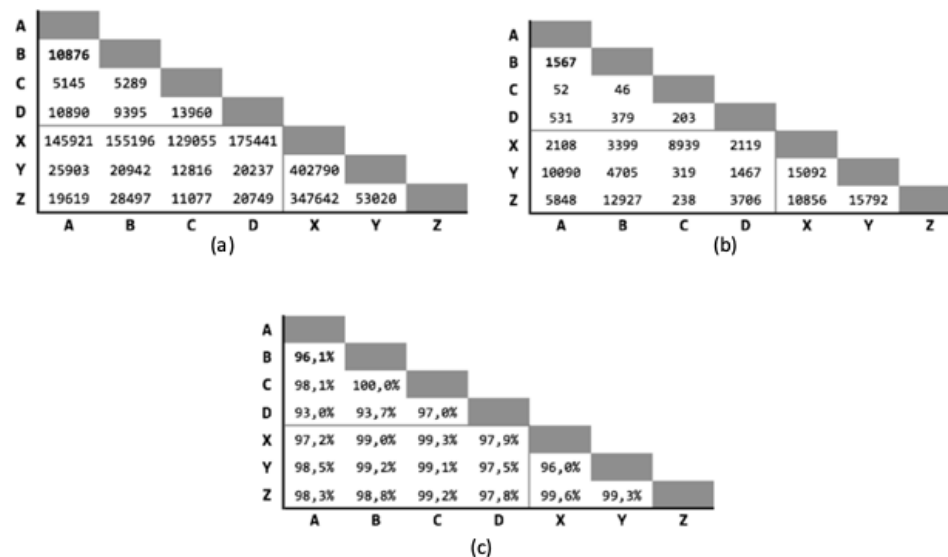
Figure 4. Tests starting conditions after the indexing phase, for all the possible pairs of databases. (a) Number of candidate record pairs to be classified, obtained from the indexing phase. (b) Number of True-Match record pair in the gold standard. (c) Pair completeness, i.e. percentage of true match retrieved in the indexing phase with respect to the gold standard.

Since there are 7 different databases available, grouped in municipalities (4) and banks (3), 21 executions will be performed for each test, pairing databases between groups and excluding deduplication (that is the match of a database against itself). Only the pair (A, B) is used as training-set.

The chosen figure of merits are *precision* and *recall*, preferred to plain accuracy due to the moderate imbalance, as explained by Christen and Goiser (2007). Their average and standard deviation over the 21 runs are reported in the following.

Exact Matching

In the very first test, RL has been performed with an *Exact Match* goal, where the candidate record pairs are classified as match only if all the respective fields are perfectly equal. This test has been carried out only to show how much two databases, while containing the same information, actually differ in the values of their attributes.
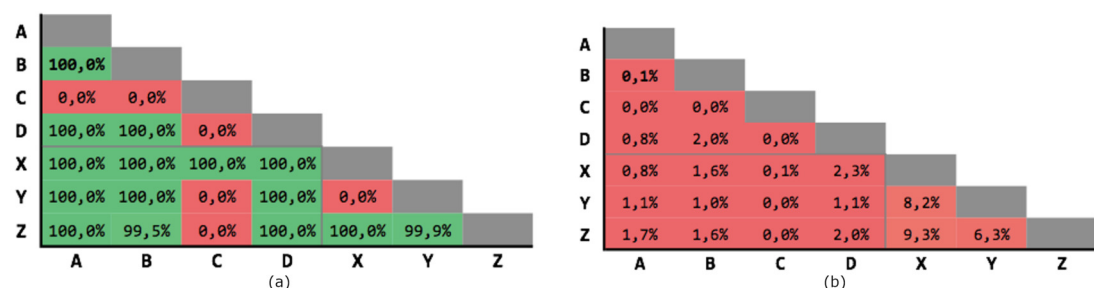
Figure 5. Exact Matching Results. (a) precision; (b) recall.

As expected (Fig. 5), the precision is extremely high, but the number of matches is extremely low, as shown by the recall. In fact, in some cases no pair have been identified, with a maximum of only 4 matches, by consequence of errors, noise, and random variations in the corresponding data.

Threshold based classification

In the next test an *optimal binary threshold classifier* was used. The threshold value was determined through a PR graph (precision-recall) generated on the training data-set. As threshold, the value having coordinates closest to the ideal point (1.0,1.0) was chosen (see Figg. A.1, A.2 and A.3 in Appendix).

Levenshtein threshold

in this test, the only similarity criterion was the Levenshtein normalized distance Levenshtein (1966) — one of the most widely used comparison metrics.

- *Comparing*: Levenshtein normalized distance.

- *Classifying*: binary threshold with $\theta = 0.7$ as optimal value, applied to the weighted sum of the similarity vector.

- *Weighting*: each attribute has the same importance and weight.
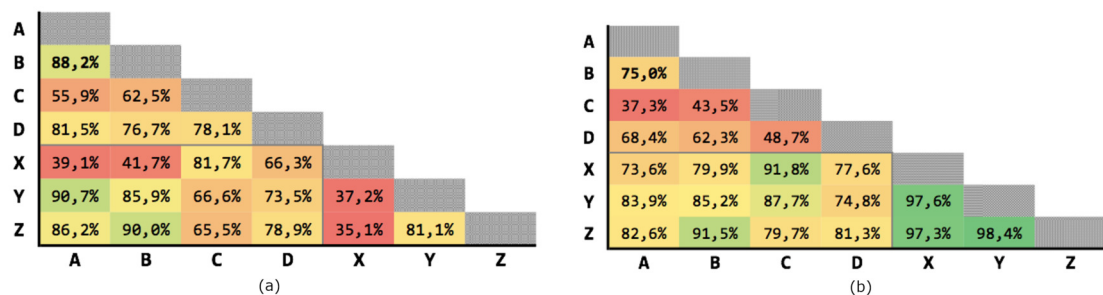
Figure 6. Results for Levenshtein normalized distance with threshold $\theta = 0.7$. (a) precision; (b) recall.

Fig. 6 shows the results obtained on the 21 executions. The average precision is 69,63% and the average recall is 77,04%, although with high variability.

Multiple criteria threshold

In this test, multiple comparison criteria for each attribute were used, with the underlying idea that different criteria measure different facets of the similarity between them. Recall actually improved.

- *Comparing*: each attribute is compared using the following metrics:

  (a) Jaro-Winkler,   (d) Jaccard,   (g) Trigrams,
  (b) Levenshtein,   (e) Siresen-Dice,   (h) Exact.
  (c) Cosine,   (f) Bigrams,

- *Classifying*: binary threshold with $\theta = 0.63$ as optimal value applied to the weighted sum of the similarity vector.

- *Weighting*: each attribute has the same importance and weight.

Fig. 7 shows the results of the 21 executions: the average precision is 71,93% and the average recall is 85,85%, although with high variability.

Weighted multiple criteria threshold

In this test, the previous results were improved by varying the importance of the various fields through an appropriate weighing. The weights, for each attribute, were automatically determined based on their distinct values distribution, and normalized in such a way that their sum is equal to 1.0 (see Fig. A.4 in Appendix).

Among the various possibilities of normalization (*linear*, *max*, *quadratic*, *exp* ...) that *logarithmic* seems to give the best results for both overall precision and recall maintaining low variance (see Fig. A.4 in Appendix).

	A	B	C	D	X	Y	Z
A							
B	91,5%						
C	55,6%	65,3%					
D	86,3%	84,4%	79,5%				
X	42,2%	44,2%	82,2%	67,4%			
Y	91,1%	87,8%	68,5%	76,2%	40,9%		
Z	88,4%	90,7%	65,6%	81,1%	39,2%	82,7%	

(a)

	A	B	C	D	X	Y	Z
A							
B	79,1%						
C	58,8%	69,6%					
D	74,9%	68,7%	69,0%				
X	87,1%	87,8%	98,6%	83,1%			
Y	91,1%	91,7%	97,2%	83,1%	99,6%		
Z	89,9%	94,7%	93,6%	86,1%	99,6%	99,8%	

(b)
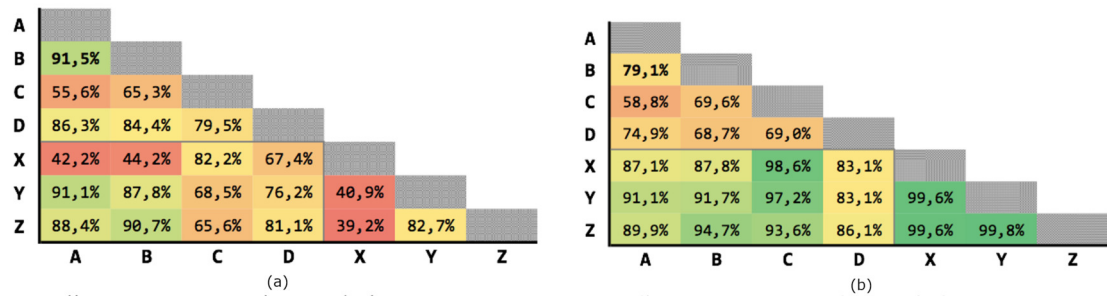
Figure 7. Results for multiple criteria similarity with threshold $\theta = 0.63$, unweighted. (a) precision; (b) recall.

- ***Comparing***: each attribute is compared using the following metrics:

  (a) Jaro-Winkler,   (d) Jaccard,   (g) Trigrams,
  (b) Levenshtein,   (e) Siresen-Dice,   (h) Exact.
  (c) Cosine,   (f) Bigrams,

- ***Classifying***: binary threshold with $\theta = 0.63$ as optimal value applied to the weighted sum of the similarity vector.

- ***Weighting***: the weights of the various attributes are estimated according to the distribution of their distinct values and normalized using a logarithmic function. The associated weight is directly proportional to the number of distinct values over the totals.

	A	B	C	D	X	Y	Z
A							
B	95,3%						
C	78,1%	83,6%					
D	93,1%	95,3%	86,2%				
X	87,4%	86,5%	89,9%	88,2%			
Y	95,1%	96,2%	84,9%	90,1%	89,0%		
Z	94,1%	94,2%	80,1%	87,9%	88,1%	93,8%	

(a)

	A	B	C	D	X	Y	Z
A							
B	97,7%						
C	98,0%	100,0%					
D	98,0%	74,1%	98,0%				
X	92,1%	91,0%	99,5%	85,5%			
Y	95,3%	95,3%	99,4%	88,3%	99,5%		
Z	94,7%	96,6%	97,0%	90,2%	99,5%	99,8%	

(b)
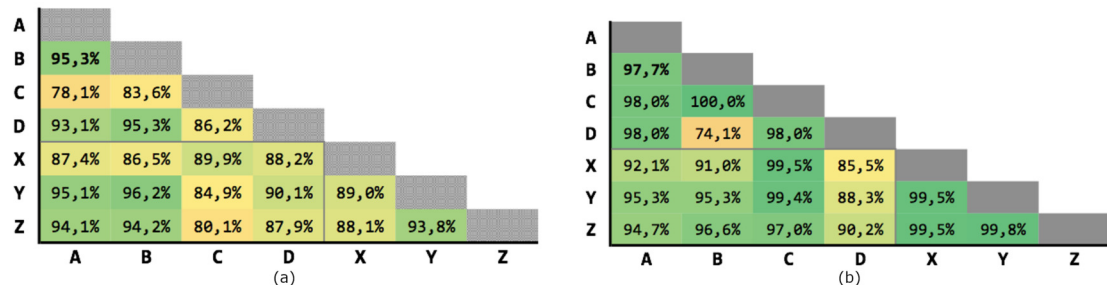
Figure 8. Results for multiple criteria similarity with threshold $\theta = 0.63$, weighted. (a) precision; (b) recall.

Fig. 8 shows the results of the 21 executions: the average precision is 89,37% and the average recall is 94,74%, with low variability.

MLP based classification
To allow a fair comparison, the tests using MLP classifier have followed the same schema of the the previous ones.

MLP Levenshtein
in this test, only the normalized Levenshtein distance was used, likewise the homonym test with the threshold classifier. As can be seen, the results clearly outperform all previous ones.

- ***Comparing***: Levenshtein normalized distance only.

- ***Classifying***: MLP based classifier.

Fig. 9 shows the results of the 21 executions: the average precision is 95,04% and the average recall is 97,71%, with very low variability.
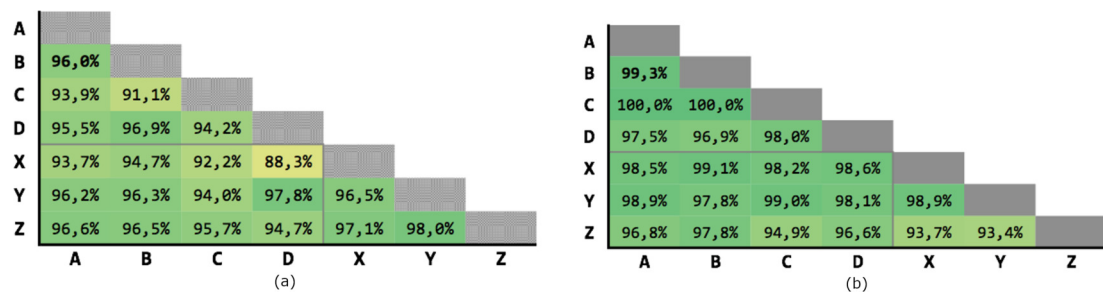
Figure 9. Results of the Levenshtein test with MLP classifier. (a) precision; (b) recall.

MLP with multiple criteria

In this test, multiple comparison criteria for each attributes have been used, likewise the homonym test with the threshold classifier. The results are almost perfect, especially for recall. In addition, the high precision allowed the manual control of the "false-positive" pairs, many of which are actually correct, but due to errors have a different fiscal code in the gold standard (see *Data sources*. Considering these fixes, precision reaches 100% in some cases.

- *Comparing*: Each attribute is compared using the following comparators:

  (a) Jaro-Winkler,   (d) Jaccard,   (g) Trigrams,
  (b) Levenshtein,   (e) Siresen-Dice,   (h) Exact.
  (c) Cosine,   (f) Bigrams,

- *Classifying*: MLP based classifier.
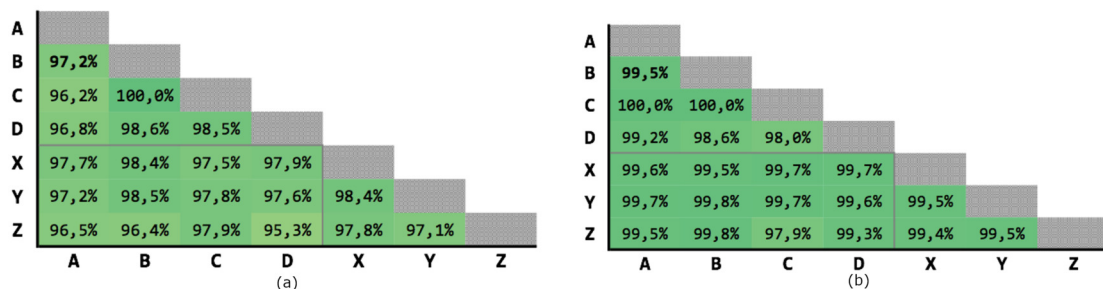
Figure 10. Results of test #2 for MLP classification. (a) precision; (b) recall.

Fig. 10 shows the results of the 21 executions: the average precision is 97,58% and the average recall is 99,39%, with very low variability.

Table 3. Experimental results Summary: Summarized results obtained through the mean values and standard deviation of the 21 executions for each test.

Classifier	Comparators	%Precision	%Recall
Threshold	Levenshtein	69,63% ± 17,76%	70,04% ± 16,66%
Threshold	All	71,93% ± 17,51%	85,85% ± 11,61%
Weighted Threshold	All	89,37% ± 4,98%	94,74% ± 6,12%
Multilayer Perceptron	Levenshtein	95,04% ± 2,28%	97,71% ± 1,78%
Multilayer Perceptron	All	97,58% ± 0,99%	99,39% ± 0,55%

CONCLUSIONS

First the various stages of the classic Record Linkage (RL) process have been presented, then a classifier based on multiple criteria and Neural Networks has been proposed in the *classification* stage of RL.

320 Specifically, the chaining of different similarity measures on different fields has been used as feature
321 vector for the subsequent classification of record pairs based on Multi-Layer Perceptron (MLP). The
322 proposed feature vector plus MLP classifier has been tested on several real-world datasets belonging
323 to geographically close banks and municipalities, scoring remarkably (please see Tab. 3) and clearly
324 outperforming the threshold-based methods. Neural Networks, even with a shallow architecture and few
325 nodes, proved to be effective classifiers and should be seriously considered for RL when even a modest
326 amount of labeled data is available.

ACKNOWLEDGEMENTS

328 The authors thank SADAS s.r.l.[1] for providing the necessary data and making their columnar database,
329 proprietary technologies and research laboratories available.

REFERENCES

331 Aiken, V. C. F., Dórea, J. R. R., Acedo, J. S., de Sousa, F. G., Dias, F. G., and de Magalhães Rosa, G. J.
332   (2019). Record linkage for farm-level data analytics: Comparison of deterministic, stochastic and
333   machine learning methods. *Computers and Electronics in Agriculture*, 163:104857.
334 Batini, C. and Scannapieco, M. (2006). *Data Quality: Concepts, Methodologies and Techniques*. Springer,
335   Berlin.
336 Batista, G. E. A. P. A., Bazzan, A. L. C., and Monard, M. C. (2003). Balancing training data for automated
337   annotation of keywords: a case study.
338 Batista, G. E. A. P. A., Prati, R. C., and Monard, M. C. (2004). A study of the behavior of several methods
339   for balancing machine learning training data. *SIGKDD Explor. Newsl.*, 6(1):20–29.
340 Bowyer, K. W., Chawla, N. V., Hall, L. O., and Kegelmeyer, W. P. (2011). SMOTE: synthetic minority
341   over-sampling technique. *CoRR*, abs/1106.1813.
342 Brizan, D. G. and Tansel, A. U. (2006). A. survey of entity resolution and record linkage methodologies.
343   *Communications of the IIMA*, 6(3):5.
344 Christen, P. (2011). A survey of indexing techniques for scalable record linkage and deduplication. *IEEE
345   transactions on knowledge and data engineering*, 24(9):1537–1555.
346 Christen, P. (2012). *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and
347   Duplicate Detection*. Springer Publishing Company, Incorporated.
348 Christen, P. and Goiser, K. (2007). Quality and complexity measures for data linkage and deduplication.
349   In *in Quality Measures in Data Mining, ser. Studies in Computational Intelligence*, pages 127–151.
350   Springer.
351 Di Cicco, V., Firmani, D., Koudas, N., Merialdo, P., and Srivastava, D. (2019). Interpreting deep
352   learning models for entity resolution: An experience report using lime. In *Proceedings of the Second
353   International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*, aiDM
354   '19, pages 8:1–8:4, New York, NY, USA. ACM.
355 Dong, X. L. and Rekatsinas, T. (2018). Data integration and machine learning: A natural synergy. In
356   *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pages
357   1645–1650, New York, NY, USA. ACM.
358 Ebraheem, M., Thirumuruganathan, S., Joty, S. R., Ouzzani, M., and Tang, N. (2018). Distributed
359   representations of tuples for entity resolution. *PVLDB*, 11:1454–1467.
360 Gottapu, R. D., Dagli, C., and Ali, B. (2016). Entity resolution using convolutional neural network.
361   *Procedia Computer Science*, 95:153 – 158. Complex Adaptive Systems Los Angeles, CA November
362   2-4, 2016.
363 Harron, K., Dibben, C., Boyd, J., Hjern, A., Azimaee, M., Barreto, M. L., and Goldstein, H. (2017).
364   Challenges in administrative data linkage for research. *Big data & society*, 4(2):2053951717745678.
365 He, H., Bai, Y., Garcia, E. A., and Li, S. (2008). Adasyn: Adaptive synthetic sampling approach
366   for imbalanced learning. In *IN: IEEE INTERNATIONAL JOINT CONFERENCE ON NEURAL
367   NETWORKS (IEEE WORLD CONGRESS ON COMPUTATIONAL INTELLIGENCE), IJCNN 2008*,
368   pages 1322–1328.

---

[1] website: https://www.sadasdb.com/en/

Hou, B., Chen, Q., Shen, J., Liu, X., Zhong, P., Wang, Y., Chen, Z., and Li, Z. (2019). Gradual machine learning for entity resolution. In *The World Wide Web Conference*, WWW '19, pages 3526–3530, New York, NY, USA. ACM.

Kasai, J., Qian, K., Gurajada, S., Li, Y., and Popa, L. (2019). Low-resource deep entity resolution with transfer and active learning. *CoRR*, abs/1906.08042.

Kooli, N., Allesiardo, R., and Pigneul, E. (2018). Deep learning based approach for entity resolution in databases. In *Intelligent Information and Database Systems*, pages 3–12, Cham. Springer International Publishing.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710. Doklady Akademii Nauk SSSR, V163 No4 845-848 1965.

Maratea, A. and Ferone, A. (2018). Deep neural networks and explainable machine learning. In *Fuzzy Logic and Applications - 12th International Workshop, WILF 2018, Genoa, Italy, September 6-7, 2018, Revised Selected Papers*, pages 253–256.

Mudgal, S., Li, H., Rekatsinas, T., Doan, A., Park, Y., Krishnan, G., Deep, R., Arcaute, E., and Raghavendra, V. (2018). Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pages 19–34, New York, NY, USA. ACM.

Navarro, G. (2001). A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1):31–88.

Siegert, Y., Jiang, X., Krieg, V., and Bartholomäus, S. (2016). Classification-based record linkage with pseudonymized data for epidemiological cancer registries. *IEEE Transactions on Multimedia*, 18(10):1929–1941.

Subitha, S. and Punitha, S. C. (2014). An effective method for matching patient records from multiple databases using neural network. *International Journal of Computer Applications*, 104(12):17–21.

Vatsalan, D., Christen, P., and Verykios, V. S. (2013). A taxonomy of privacy-preserving record linkage techniques. *Information Systems*, 38(6):946–969.

Wilson, D. R. (2011). Beyond probabilistic record linkage: Using neural networks and complex features to improve genealogical record linkage. In *The 2011 International Joint Conference on Neural Networks*, pages 9–14.

Zhang, L. Q. (2011). *Record Linkage Methodology and Applications*, pages 377–413. Springer New York, New York, NY.

Zhao, H. and Ram, S. (2005). Entity identification for heterogeneous database integration—a multiple classifier system approach and empirical evaluation. *Information Systems*, 30(2):119–132.

⁴⁰¹ **A APPENDIX**

⁴⁰² **PRECISION RECALL CURVE**

⁴⁰³ In this section are shown in Figg. A.1, A.2 and A.3, for each threshold-based test, the precision-recall
⁴⁰⁴ curve generated on the training-set database pair (A,B). These curves were used to determine the optimal
⁴⁰⁵ threshold value for classifiers.
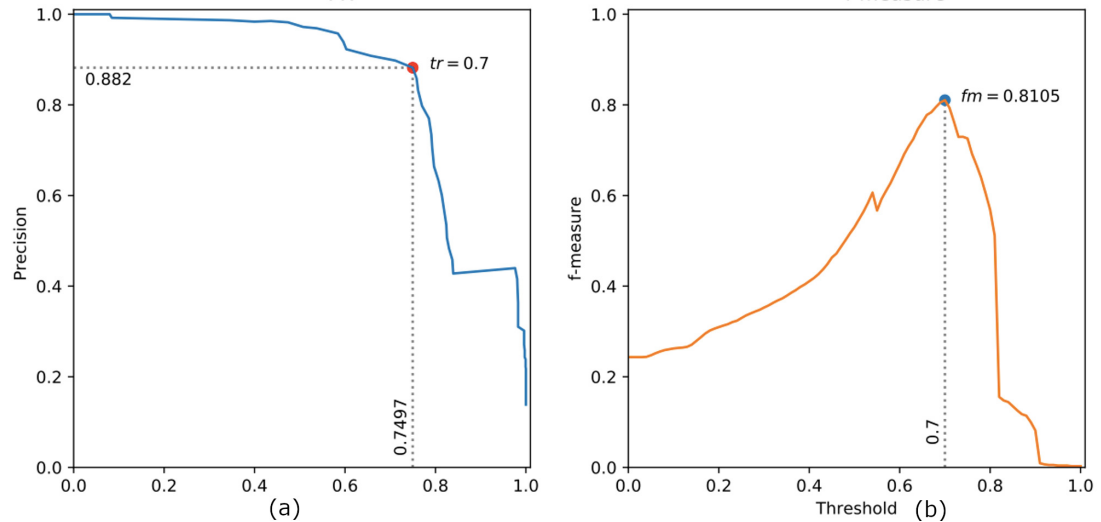
Figure A.1. Levenshtein threshold. (a) the precision-recall curve generated on the training-set in experiment #1; The optimal threshold is $tr = 0.7$. (b) the F1-score when the threshold changes.
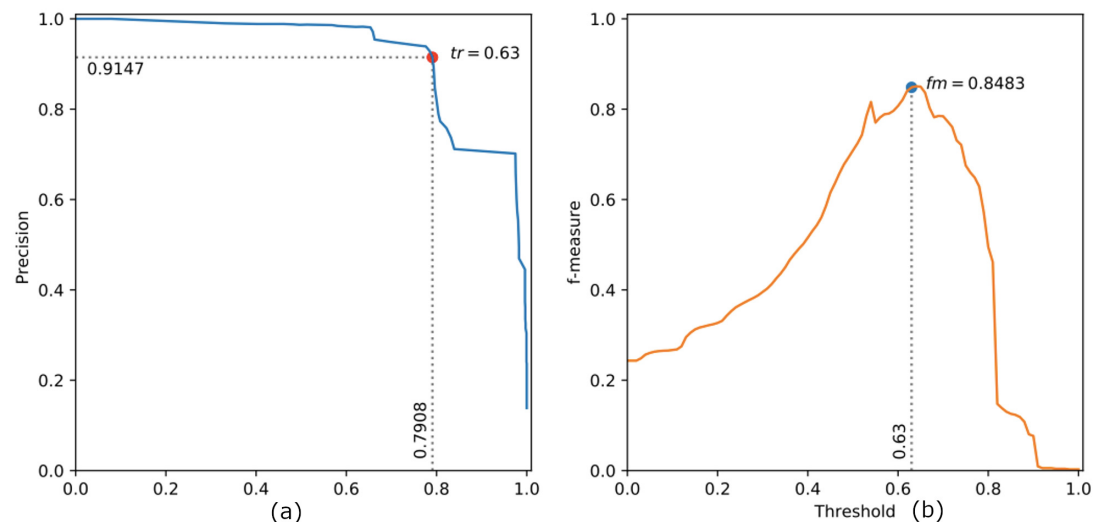
Figure A.2. Multiple criteria threshold. (a) the precision-recall curve generated on the training-set in experiment #2; the optimal threshold is $tr = 0.63$. (b) the F1-score when the threshold changes.

⁴⁰⁶ **WEIGHT VECTOR ESTIMATION**

In this section, the methodology used in experiment 3 for the weights estimation is described. The weight vector has as many components as there are attributes selected for the RL each of which is calculated as:

$$w_i = \frac{\log\|\operatorname{supp}\mathbf{a}_i\|}{\log\|\mathbf{a}_i\|} \cdot \frac{\log\|\operatorname{supp}\mathbf{b}_i\|}{\log\|\mathbf{b}_i\|} \tag{1}$$
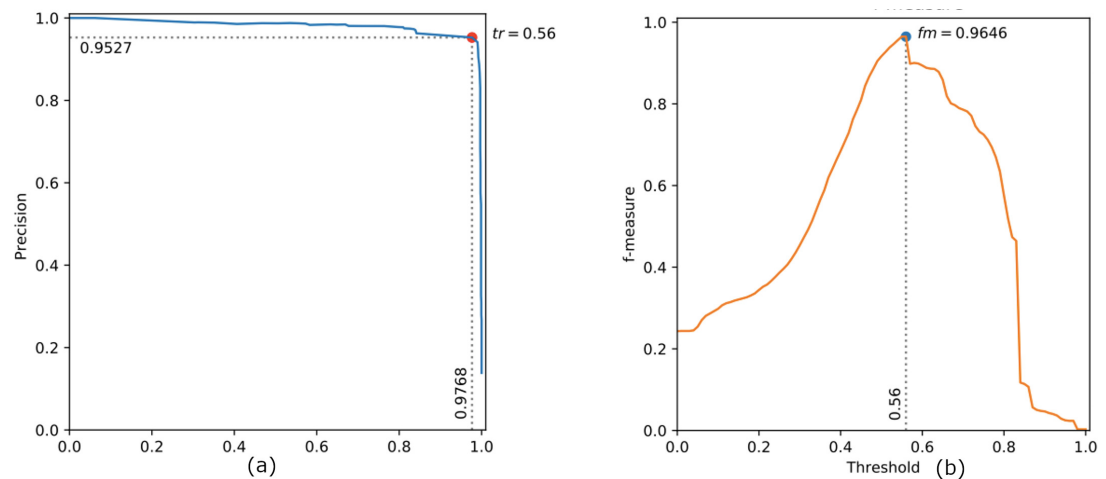
Figure A.3. Weighted multiple criteria threshold. (a) the precision-recall curve generated on the training-set in experiment #3; the optimal threshold is $tr = 0.56$. (b) the F1-score when the threshold changes.

407 where $\mathbf{a}_i$ and $\mathbf{b}_i$ are the multi-set values of the corresponding i-th attribute of the first and second table
408 respectively. The notation supp$\cdot$ indicate the support, i.e. the set of unique items in multi-set and $\|\cdot\|$
409 denotes the cardinality.

410   This measure takes into account both the number of distinct values and the size difference of the two
411 tables.

WEIGHT VECTOR NORMALIZATION
412

413 In this section are shown, the vector normalization technique and then, for each normalization base
414 function, in Fig. A.4, the precision-recall curve generated on the training-set database pair (A,B). These
415 curves were used to select the best type of normalization.

Normalization
416

417 To alter the contribution of the various attributes during the classification while maintaining unchanged
418 the weighted sum of the components of the similarity vector, a normalization of the weight vector is
419 necessary.

Given a weight vector $\mathbf{w} = (w_1, w_2, \ldots, w_n)$ and same *base* function f a normalized vector $\hat{\mathbf{w}} = (\hat{w}_1, \hat{w}_2, \ldots, \hat{w}_n)$ can be obtained simply by applying element-wise the equation:

$$\hat{w}_i = \frac{f(w_i)}{\sum_{j=1}^{n} f(w_j)} \tag{2}$$

420 i.e. applying the function f to each element of the input vector w_i and normalizing these values by
421 dividing by the sum of all these values; this normalization ensures that the sum of the components of the
422 output vector $\hat{\mathbf{w}}$ is 1.

The tested base function f are listed below:

$$
\begin{array}{llll}
\text{linear:} & f(x) = x & \text{quadratic:} & f(x) = x^2 \\
\text{cubic:} & f(x) = x^3 & \text{logaritmic:} & f(x) = \log(1+x) \\
\text{softmax:} & f(x) = e^x & \text{inverse:} & f(x) = x^{-1}
\end{array}
$$

423

Precision-recall curves

Results for both overall precision and recall among the various possibilities of normalization: *linear*, *max*, *quadratic*, *exp ... logarithmic*.
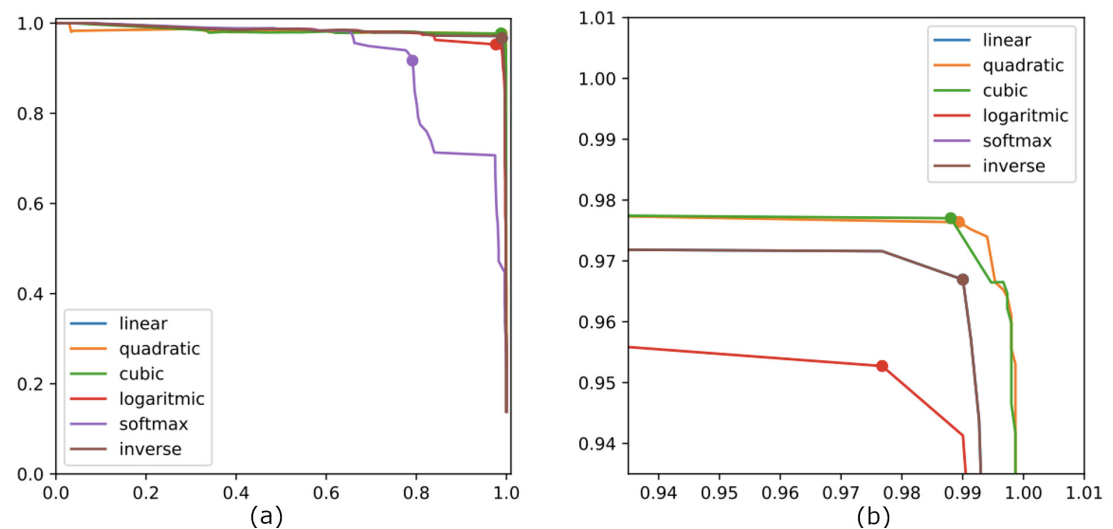
Figure A.4. (a) the precision-recall curves generated on the training-set in experiment #3 for each of the normalization techniques. (b) a zoomed view of the same curves. The relative winner, in terms of precision and recall, was the quadratic normalization, but results of higher quality and lower variance were obtained using a logarithmic function.