Noname manuscript No.
(will be inserted by the editor)

Schema Matching Prediction with Applications to Data Source Discovery and Dynamic Ensembling

Tomer Sagi · Avigdor Gal

the date of receipt and acceptance should be inserted later

Abstract Web-scale data integration involves fully automated efforts which lack knowledge of the exact match between data descriptions. In this paper we introduce *schema matching prediction*, an assessment mechanism to support schema matchers in the absence of an *exact match*. Given attribute pair-wise similarity measures, a predictor predicts the success of a matcher in identifying correct correspondences. We present a comprehensive framework in which predictors can be defined, designed and evaluated. We formally define schema matching evaluation and schema matching prediction using similarity spaces and discuss a set of four desirable properties of predictors, namely correlation, robustness, tunability, and generalization. We present a method for constructing predictors, supporting generalization and introduce prediction models as means of tuning prediction towards various quality measures. We define the empirical properties of correlation and robustness and provide concrete measures for their evaluation. over a set of predictors at different granularity levels. To illustrate the usefulness of schema matching prediction, we propose a method for ranking the relevance of deep Web sources with respect to given user needs. We show how predictors can assist in the design of schema matching systems. Finally, we show how prediction can support dynamic weight setting of matchers in an ensemble, thus improving upon current state-of-the-art weight setting methods. These claims are evaluated in an extensive empirical evaluation.

Technion – Israel Institute of Technology
E-mail: {stomers7@tx,avigal@ie}.technion.ac.il

1 Introduction

The use of Web data structural features was shown to be beneficial in the context of information seeking on the Web [28]. In some cases, information annotation in the form of, *e.g.*, ontologies, can assist in overcoming the ambiguity inherent to natural languages. In other cases, retrieval of information depends on querying, rather than browsing the Web. A prominent example of information seeking via querying is that of Web forms, commonly used as portals to deep Web information

Whenever structural features are present, information seeking on the Web is performed by using various matching techniques over these features, ranging from schema matching to ontology matching techniques. Generally speaking, schema and ontology matching provide correspondences between concepts describing the meaning of data (*e.g.*, attributes in database schemata and fields in Web forms) in various heterogeneous, distributed data sources. Schema matching is conceived to be one of the basic operations required by the process of data and schema integration [4], and thus has great effect on its outcomes.

In its origin, schema matching was conceived to be a preliminary process to schema mapping to serve applications such as query rewriting and database integration. A basic assumption within this field is that schema matching provides a set of correspondences to be then validated by some human expert before mapping expressions are generated.

Over the years, schema matching research has expanded and specialized to answer research and application questions in a variety of domains. Smith et. al. [42] observe that for applications that involve large enterprises, such as locating relevant concepts, there is no need for mapping expressions to be created, nor is it

possible to provide a set of definite correspondences. Even more important, exact matches, the outcome of a manual validation, are rare, hard to come by and generate. A similar setting exists in searching over structured data on the Web. Users can benefit from guidance towards relevant data sources, without the need of a mapping expression and definite correspondences. The vast amount of data out there also makes the task of manual integration of data all but impossible. The dataspace vision [15] also extends the traditional role of schema matching in data integration. Dataspaces can assist in reducing upfront costs for setting up a data integration system by gradually specifying schema mappings through interaction with end users in a pay-as-you-go fashion. In addition, the dataspace vision requires tools for the bootstrapping of service provision, profiling, discovering data sources, *etc.*

In the absence of an exact match, schema matchers perform a "best effort" matching without any indication of the prospective success of their efforts. In this work we introduce *schema matching prediction*, an assessment mechanism to support schema matchers in the absence of an exact match. Predictors foretell the success of a matcher in identifying correct correspondences by analyzing the matcher's pair-wise similarity scores.

Prediction carries its own merit. Presenting the user with a quality assessment of a match is a first step towards decision making under uncertainty. Prediction can also serve in directing the flow of data source discovery. For example, it can serve as a decision factor on whether accessing data through one Web form is preferred over the other. Also, prediction puts schema matchers in perspective, explaining why some matchers work while others do not and directing the improvement of existing matchers. Finally, prediction can serve schema and ontology matching tasks such as matcher self-configuration, matcher selection,user involvement optimization, *etc.*

Predicting the quality of the outcome of a schema matching process is a new task in schema matching literature, a task that has been mentioned as side remarks in some papers but never, to the best of our knowledge, treated in a thorough scientific manner. Towards this end, we offer a comprehensive examination of schema matching prediction on the conceptual, architectural, and empirical levels. On the conceptual level we present *similarity spaces*, a general model for evaluation of matching outcome and place prediction within it. On the architectural level we present predictor design-level properties and methods for constructing predictors with different desired properties. We continue the architectural discussion by showing how the *similarity spaces* model can be exploited to design predictors. We demonstrate these principles by presenting a set of predictors and by introducing the concept of prediction models. On the empirical level we present methods for evaluating predictors and discuss the desired empirical properties of predictors.

To illustrate the applicability of the proposed approach to assessing the relevance of deep Web information we introduce three applications of prediction. The first application is in data source discovery, motivated by the need to identify Web data sources that have a high fit with a user's data needs [39]. The user specifies her data needs as a set of concepts, which can be thought of as small schemata. Predictors serve as ranking measures to provide the user with a manageable set of candidate Web forms to consider. In the second application presented, weights of schema matchers in an ensemble are dynamically set as follows. By predicting the quality of each match result, predictors enable the matching system to change ensemble matcher weights at run-time. Dynamic weight setting allows automatic tuning of the matching task, thus allowing for fully automated processes that are more apt for Web-scale integration tasks than semi-automated iterated efforts. The third application demonstrates how entry-level predictors support designers of schema matching systems when designing second line matchers.

This comprehensive examination of prediction usage scenarios is supported by an extensive empirical evaluation. Our evaluation shows that carefully constructed predictors can be a powerful tool in the hands of schema and ontology matching researchers, practitioners, and Web users. In particular, we show that the predictors we propose in this work are correlated well with the quality measures they aim to predict (*e.g.*, precision and recall). We also show that fine-grain predictors, based on single attribute evaluation, correlate better with the quality outcome than schema level predictors. Relevant Web sources are shown to be accurately discovered using predictors and attribute-level predictors are shown to perform well in dynamic ensemble construction.

In this work we make the following contributions:

- At the conceptual level we introduce an evaluation model of matching result quality in the absence of an exact match.
- We present a general method for designing predictors using similarity subspaces and give examples of predictors that can be generated using this method.
- We propose a method for tuning predictor models to predict different quality measures.
- We offer a novel method for ranking schemata based on relevance to user needs, using predictors.

– We provide an extensive empirical evaluation, showing the benefits of our proposed approach towards prediction.

The rest of the paper is organized as follows. Section 8 discusses related work and Section 2 provides background on schema matching followed by the evaluation model in Section 3. A thorough exploration of predictors and predictor models is provided in Section 4. In Section 5 we introduce three use-cases for schema matching prediction. Section 6 provides a description of the empirical evaluation method followed by the evaluation results in Section 7 and a concluding discussion in Section 9.

(a) A Similarity Matrix

$S_0 \longrightarrow$ / $\downarrow S_1$	1 cardNum	2 city	3 arrivalDay	4 checkIn Time
1 clientNum	0.84	0.32	0.32	0.30
2 city	0.29	1.00	0.33	0.30
3 checkInDay	0.34	0.33	0.35	0.64

(b) A Binary Similarity Matrix

$S_0 \longrightarrow$ / $\downarrow S_1$	1 cardNum	2 city	3 arrivalDay	4 checkIn Time
1 clientNum	1	0	0	0
2 city	0	1	0	0
3 checkInDay	0	0	0	1

Table 1: Similarity Matrix Examples

2 Preliminaries

The following schema matching model is based on [16]. Let schema $S = \{a_1, a_2, \ldots, a_n\}$ be a finite set of attributes. Attributes can be both simple and compound, and compound attributes should not necessarily be disjoint. For any schemata pair $\{S, S'\}$, let $\mathcal{S} = S \times S'$ be the set of all possible attribute correspondences between S and S'. Let $M(S, S')$ be an $n \times n'$ *similarity matrix* over $\mathcal{S}$, where $M_{i,j}$ (typically a real number in $[0, 1]$) represents a degree of similarity between the i-th attribute of S and the j-th attribute of S'. $M(S, S')$ is a *binary* similarity matrix if for all $1 \leq i \leq n$ and $1 \leq j \leq n'$, $M_{i,j} \in \{0, 1\}$. A (possibly binary) similarity matrix is the output of the matching process.

Example 1 Table 1 provides examples of two similarity matrices between two simplified schemas, one (S_0) with four attributes and the other (S_1) with three. The similarity matrix in Table 1(a) is the outcome of a matching process, using some matcher. The similarity matrix in Table 1(b) is a binary similarity matrix, generated as the outcome of a decision-maker matcher [16], also known as *alignments as solutions* [41]. Such a matcher applies a heuristic of choosing those attribute pairs in which matcher confidence is above 0.5. It is worth noting that the matcher prefers matching CheckInDay of S_1 with CheckInTime on S_0 over arrivalDay. This may occur, for example, whenever a string matcher is used, giving more emphasis to the checkin component of the attribute name.

Matching schemas is often a stepped process in which different algorithms, rules, and constraints are applied. Several classifications of schema matching steps have been proposed over the years (see *e.g.*, [24, 8]) and this work focuses on three of the steps. First, we separate matchers that are applied directly to the problem and matchers that are applied to the outcome of other matchers, the former class of matchers is called *first-line matchers (1LM)* and the latter class is termed *second-line matchers (2LM)* [16]. In addition, we focus on ensemble-based methods, combining the results of various matchers to generate a combined match. We dub such methods *ensemble matchers*.

Putting the three matching tasks in the context of our model, 1LMs receive two schemata and return a similarity matrix, 2LMs receive a similarity matrix and return a (possibly binary) similarity matrix, and ensemble matchers receive a schema pair and a set of similarity matrices as input and return a (possibly binary) similarity matrix representing the combined result of these matchers. An important aspect of ensemble matchers is the matcher combination method in an ensemble. It can be determined *statically*, prior to the matching task. Alternatively, it can be *dynamically* generated, using the features of the schemata and similarity matrices at hand.

3 Evaluation Model

We start this section with the definition of similarity spaces in Section 3.1. In Section 3.2 we formally define schema matching evaluation using similarity spaces.

3.1 Similarity Spaces

We propose a vector space representation of schema matching outcome named *similarity spaces*.[1] Vector spaces support the functional analysis of evaluation methods. Functional analysis is the study of vector spaces utilizing several branches of applied mathematics [25]. Most of the principal results in functional

[1] The proposed term of a *similarity space* should not be confused with the one proposed by Zobel and Moffat [47] in the context of document vector spaces.

analysis are expressed as abstractions of intuitive geometric properties of three dimensional space. However, the same mathematical formulae hold in n-dimensional space ($n > 3$) as well. For convenience, we maintain matrix notation when referring to a dimension, marking a dimension as an (i, j) coordinate.

Definition 1 Given schemata S and S', a *similarity space* $\mathcal{V}_{\mathcal{S}}(S, S') = [0, 1]^{|\mathcal{S}|}$ is an $|\mathcal{S}|$-dimension vector space such that each dimension (i, j) in $\mathcal{V}_{\mathcal{S}}(S, S')$ corresponds to the attribute pair (a_i, a_j) in $\mathcal{S}$.

Whenever the referenced schemata S and S' are clear from the context, we use $\mathcal{V}_{\mathcal{S}}$ or $\mathcal{V}$ as a shorthand notation of $\mathcal{V}_{\mathcal{S}}(S, S')$.

Each entry in a matrix M over $\mathcal{S} = S \times S'$ is represented as an element of a similarity vector in $\mathcal{V}$. Therefore, a similarity vector represents the similarity of $|\mathcal{S}|$ **pairs** of attributes.

3.1.1 Similarity Subspaces

Evaluating the similarity of a schema as a whole may be too coarse for some matching tasks, as we show in Section 5. Therefore, we would like to be able to make statements over subspaces of the similarity space. For example, we may be interested in results for some attribute $s \in S$ only. For each dimension (i, j) in $\mathcal{V}$ we denote by $v^{i,j}$ the vector $(0, 0, ..., 1, ..., 0)$, the vector with all 0 values except the (i, j) element, assigned with a 1 value. We use these vectors to define similarity subspaces as follows.

Definition 2 $\boldsymbol{V} \subseteq \mathcal{V}_{\mathcal{S}}(S, S')$ is a *subspace* of $\mathcal{V}_{\mathcal{S}}(S, S')$ if $\boldsymbol{V}$ is generated by a set of vectors $v^{i,j}$ such that (i, j) is in $\mathcal{V}_{\mathcal{S}}(S, S')$.

Definition 3 Given a similarity matrix M over $\mathcal{S} = S \times S'$ and a subspace $\boldsymbol{V} \subseteq \mathcal{V}_{\mathcal{S}}(S, S')$, a *similarity vector* $\mathbf{v}(M)$ from the subspace $\boldsymbol{V}$ is the vector:

$$\mathbf{v}(M) = \sum_{(i,j) \text{ dimension in } \mathcal{V}_{\mathcal{S}}(S,S')} M_{i,j} v^{i,j} \quad (1)$$

The dimensionality of similarity subspaces of $\mathcal{V}$ ranges from 1 to $|\mathcal{S}|$. A similarity vector of dimension 1 represents a single matrix entry and one with dimension $|\mathcal{S}|$ is defined over the entire vector space, representing the whole similarity matrix. Any dimension in between represents part of the similarity matrix. For example, $\boldsymbol{V_i} = [0, 1]^{|S'|}$ is a subspace of a similarity space $\mathcal{V}$, representing the similarity of a single attribute a_i from schema S with each of the attributes of S'. We also denote by $\boldsymbol{V_{ij}} = [0, 1]^{|S|+|S'|}$ the subspace of a similarity space $\mathcal{V}$, representing the similarity of two attributes $a_i \in S$ and $a_j \in S'$ with all other attributes.

(a) Similarity Matrix

	3 arrivalDay	4 CheckInTime
3 checkInDay	0.35	0.64

(b) Binary Similarity Matrix

	3 arrivalDay	4 CheckInTime
3 checkInDay	0	1

(c) Exact Match Matrix

	3 arrivalDay	4 CheckInTime
3 checkInDay	1	1

Table 2: Partial Similarity Matrix Examples

Example 2 Consider Example 1. Let $\mathcal{V}$ be the vector space representation of the schema pair in Table 1. We define a 2-dimensional subspace $\boldsymbol{V} \subset \mathcal{V}$ over a single attribute from S_1, {checkInDay} and two attributes from S_2, {arrivalDay, CheckInTime}. Tables 2(a) and 2(b) show the relevant part of the similarity matrices in tables 1(a) and 1(b), respectively. Table 2(c) illustrates the relevant part of an exact match for the matching of checkInDay with arrivalDay and CheckInTime, as a binary similarity matrix. It is worth noting that while the decision maker chose to match checkInDay only with arrivalDay, checkInDay is in fact a combination of both arrivalDay and CheckInTime.

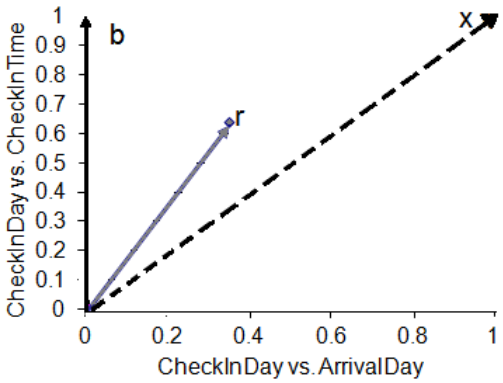


Fig. 1: Visualizing Similarity Vectors

We can now visualize the 2-dimensional similarity vectors of the three matrices in Figure 1. Each similarity vector is represented as a single point. Using the L_2 (Euclidean) norm for defining vector length, we can visualize the vectors using straight lines from the origin to the vector coordinates, as illustrated in Figure 1.

3.2 Schema Matching Evaluation

Schema matching evaluation is a task aiming at assessing the quality of a matching result.

Definition 4 Let $\mathcal{V}_{\mathcal{S}}$ be a similarity space over $\mathcal{S} = S \times S'$ and let $\boldsymbol{V} \subset \mathcal{V}_{\mathcal{S}}$ be a sub-space of $\mathcal{V}_{\mathcal{S}}$. A *schema matching evaluation method* is a function

$$g : \boldsymbol{V} \times ... \times \boldsymbol{V} \rightarrow \Re \tag{2}$$

g receives a set of similarity vectors representing schema matching results and evaluates them to return a single real value.

Informed? \ K	0	1	2	> 2
Yes	X	X		
No	pr	ed	dic	tors

Table 3: Evaluator classification

We now present a classification of different evaluators according to the properties of their input, also illustrated in Table 3. The first dimension involves the availability of an exact match in the input.

Informed evaluatiors are applied on vectors with respect to a given *exact match.*

Uninformed evaluators are applied on vectors without any knowledge given on correct attribute correspondences for the given schema pair.

The second dimension of classification is based upon the cardinality of the input vector set. We identify four classes of evaluators in this dimension:

Single Input ($K = 1$) evaluators $g : \boldsymbol{V} \rightarrow \Re$ evaluate a single similarity vector.

Double Input ($K = 2$) evaluators $g : \boldsymbol{V} \times \boldsymbol{V} \rightarrow \Re$ compare between two vectors of the same sub-space.

K-Input ($K > 2$) evaluators $g : \boldsymbol{V} \times ... \times \boldsymbol{V} \rightarrow \Re$ are the general extension to the double input functions receiving as input $k > 2$ vectors.

Schema Input ($K = 0$) evaluators $g : S_1 \times S_2 \rightarrow \Re$ are a special case that evaluate the schema matching task without a matching result but with only the original schemata intended to be matched. We note that for $K = 0$ the definition of an evaluator does not comply with that in Definition 4. The usefulness of such evaluators may seem questionable at first as they do not evaluate matching results. However, as Tu and Yu [44] have found, they can be used to predict matcher performance.

An overwhelming majority of the work to date on schema matching evaluation can be categorized into the {Informed,$K = 2$} category, comparing two similarity vectors one being a match result and the other being an exact match, representing a "correct" match, typically provided by an expert. Whenever an exact match is part of the input to g, the evaluation is performed with respect to it. For example, let $\{\mathbf{v}, \mathbf{v}^e\}$ be a pair of similarity vectors over $\mathcal{V}_{\mathcal{S}}$. $\mathbf{v}$ is a binary vector, representing the outcome of a decision maker schema matcher. $\mathbf{v}^e$ is a binary vector, representing the exact match. The well-known precision evaluation method follows this definition and can therefore be represented as follows:

$$g^{PR}(\mathbf{v}, \mathbf{v}^e) = \frac{\mathbf{v} \cdot \mathbf{v}^e}{\|\mathbf{v}\|} \tag{3}$$

where $\|\cdot\|$ represents the L_1 (Taxicab) norm. Recall is similarly defined as follows:

$$g^{RE}(\mathbf{v}, \mathbf{v}^e) = \frac{\mathbf{v} \cdot \mathbf{v}^e}{\|\mathbf{v}^e\|} \tag{4}$$

Precision and Recall serve as a basis for additional measures such as F-Measure, Overall and Accuracy, which also fall into this category. Sagi and Gal propose an extension of this basic class to non-binary vector comparison as well [38].

Other entries in this table are promising as well and may serve a wide variety of research and application domains. A few works only explored the uninformed category so far, *e.g.*, [44,30]. In this work, we introduce an uninformed K-Input evaluator, which evaluates a group of results and show its usage in data source discovery. In particular, we focus on the case of {Uninformed,$k = 1$} case.

4 Predictors

Predictors are a special class of schema matching evaluation methods, which evaluate one or more similarity vectors in order to assess the quality of the matching outcome without any knowledge on the correct correspondences (*i.e.*, exact match). Using the evaluator classification (see Table 3), predictors are uninformed evaluators.

In this section we first discuss the desired properties of predictors (Section 4.1), followed by a description of two predictor types (Section 4.2), namely internalizers and idealizers. Generalization is discussed in more details in Section 4.3. Finally, we introduce prediction models in Section 4.4 to support tunability. To illustrate our approach, we give examples of predictors throughout. Predictors were selected with the aim of better illustrating the novel aspects of this work.

4.1 Predictor Properties

We divide our characterization of predictors into two parts: structural properties that can be examined at design time (Section 4.1.1) and empirical properties that should be evaluated by experimentation (Section 4.1.2).

4.1.1 Structural Properties

We define two structural properties, namely generalization and tunability, as follows.

Generalization: Predictors may be applied to tasks with different requirements of granularity, from predicting match quality for a single attribute pair, to match quality of a schema pair, to match quality of multiple matchers, and everything in between, as illustrated in Figure 2. Predictors should be based upon principles that are applicable at several levels and can be specialized to several levels of granularity. In Section 4.3 we demonstrate how generalization enables the increase of prediction effectiveness and provide several predictors on different levels of granularity.

Tunability: Predictors may predict different qualities, putting more emphasis, for example, on precision or on recall. We should be able to tune predictors towards the desired quality in a specific scenario. In Section 4.4 we introduce prediction models as a method for providing tunability.

4.1.2 Empirical Properties

We define two empirical properties, namely correlation and robustness, as follows.

Correlation: By definition, correlation is measured between two variables. Predictors should correlate well with the quality of the matcher's decision making and in our case, we measure correlation between a predictor's value and a quality measure (such as precision and recall). An accepted measure for the correlation of continuous variables is R^2 also known as *Pearson Squared*. It's result is often interpreted as the proportion of variability in a dependent variable explained by the predicting variables [43]. In our context, predictors are the predicting variables and quality measures are the dependent variables. Useful predictors are strongly correlated[2] with the quality measures they are designed to predict, thus allowing designers to use their predicted values as a proxy for the unknown quality of the result. Correlation among predictor values (intercorrelation) indicate redundancy, leading to a selection process among predictors.

[2] According to Cohen [7], correlation values over .01 represent a small effect, over .09 a medium effect and over .25 a large effect.

Robustness: To ensure a predictor's robustness, its performance should be statistically significant when tested over a substantial number of schema pairs and stable over varying datasets and schema matchers. We use the following three tools to ensure ensure robustness:

• *Statistical significance* is the probability that the observed phenomenon (in this case the predictor being sound) emanates from random noise. Our measure for statistical significance is the one-tailed standard t-test, measuring the hypothesis that the observed result was generated by random noise. A low significance value (usually a value below 0.01) is preferred since the probability that the result observed was caused by random noise is low.

• *Sample Size:* A special attention should be given to the number of schema pairs on which a predictor is evaluated. Miles and Shevlin [**?**] provide scientists with some useful rules of thumb when calculating the minimal sample size by the expected size of the effect one wishes to measure. When applied to schema matching prediction, detecting a medium sized effect (0.3 to 0.6 correlation) on a single predictor requires 60 samples while testing six predictors require over 100 schema pairs. In the only evaluation of a schema matching predictor published to date [30], to the best of our knowledge, correlation was examined for about 35 pairs from a single dataset, which is lower than recommended by Miles and Shevlin. In our experiments (detailed in Section 7.1) the number of pairs is between 810 and 7708.

• In the field of statistics, *interfering variables* are features specific to the experiment and sampled population that may interfere with the outcome, making it an artifact of the experiment with little implication on more general settings [43]. In the context of schema matching, since predictors are based on constraints or properties that may hold only in specific datasets and schema pairs, we wish to ensure the observed prediction is independent of these features. To ensure robustness in this case one should control dataset and sample features by performing multiple regression experiments, each time controlling for prior variables such as *source dataset* and *matching system*. If the variance explained by the predictor is substantial over the variance explained by dataset and matching system priors, we may conclude that the predictor is robust.

4.2 Predictor Types

We now present and demonstrate the two predictor types, namely internalizers and idealizers.

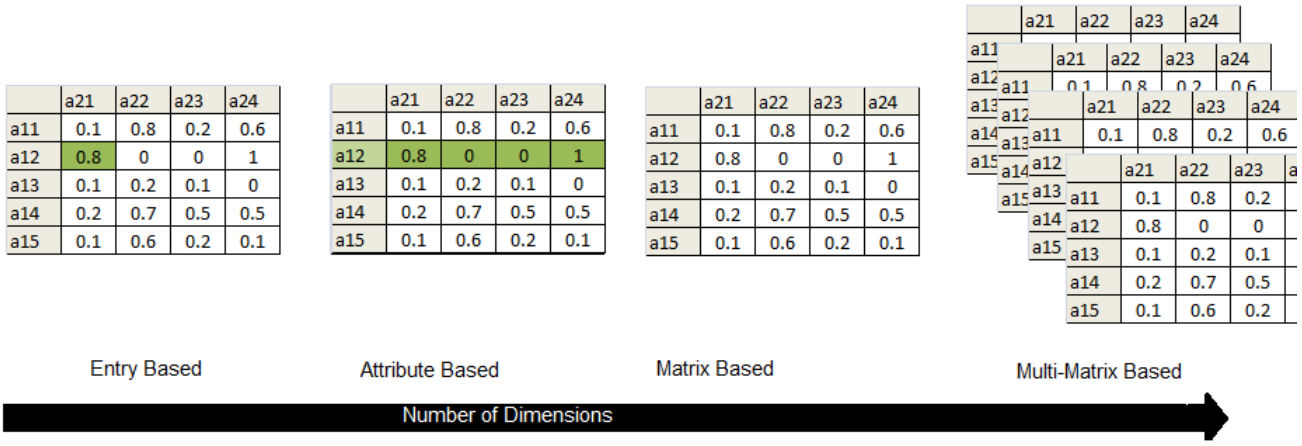


Fig. 2: Visualizing granularity

4.2.1 Internalizers

Internalizers are devised by empirical analysis of high-quality result vectors and the identification of some shared internal structure or measurable property. For example, consider the *dominants* property [16] (first used by [45] and also dubbed later as *harmony* [30]). Given a schema pair $\{S, S'\}$ and a similarity matrix $M(S, S')$, a pair of attributes $\{a_i, a_j\}$ is *dominant* if the similarity measure $M_{i,j}$ satisfies the following dominance property:

$$\forall a_k \in S' \;\; M_{i,k} \leq M_{i,j} \wedge \forall a_k \in S \;\; M_{k,j} \leq M_{i,j} \tag{5}$$

Empirical analysis (Section 7) shows that similarity vectors with a high ratio of dominant pairs often yield high quality results. We therefore define a matrix predictor *DOM* that measures the ratio of dominant values with respect to the smallest schema size. Additional internalizers are described in Section 4.3.

4.2.2 Idealizers

Idealizers assume the existence of some principle that *high quality* results adhere to and predict performance based on comparing the supplied result and an *ideal* vector, adhering to this principle. For example, one may hypothesize that an ideal match vector is a vector over $\{0,1\}^{|S \times S'|}$, containing only binary values. Therefore, one similarity vector is evaluated to be better then another if it is "closer" to some binary similarity vector. For such an idealizer in our case-study example, the vector $(0.9, 0.1)$ is preferred to $(0.5, 0.5)$. $(0.5, 0.5)$ is in equal distance to any binary similarity vector, which is further than the distance between $(0.9, 0.1)$ and $(1, 0)$. Application of this methodology requires:

Ideal Vector: principles that result vectors should adhere to, formalized as constraint functions.
Comparison: a consistent method for comparing the result vectors to ideal vectors.
Selection: a method to select the ideal vector for comparison.

We now provide formal definitions for these three requirements. We begin by adapting the definition of a constraint function from [16] to support similarity subspaces.

Definition 1 *Given a subspace $\boldsymbol{V} \subseteq \mathcal{V}_{\mathcal{S}}(S, S')$, a* constraint function *over $\boldsymbol{V}$ is a boolean function:*

$$\Gamma : \boldsymbol{V} \rightarrow \{0, 1\} \tag{6}$$

□

Γ partitions $\boldsymbol{V}$ into valid (ideal) and invalid (non-ideal) vectors so that ideal vectors satisfy Γ. We denote by $\boldsymbol{V}^i$ the set of all ideal vectors in subspace $\boldsymbol{V}$. We find the use of constraint functions in prediction to be complimentary to previous approaches. Rather than using the constraint function to modify the matcher outcome we use it to predict the current result success. While the former carries disadvantages such as reduced quality outcome, the latter is useful in characterizing the ability of a matcher to successfully handle a given instance of the matching problem.

For vector comparison, we require a match comparison function defined as follows.

Definition 2 *Let $\boldsymbol{V} \subseteq \mathcal{V}_{\mathcal{S}}(S, S')$ be a subspace and let $\mathbf{v}^i \in \boldsymbol{V}^i$ be an ideal similarity vector of a matrix M^i ($\mathbf{v}^i = \mathbf{v}(M^i)$). A* match comparison function $\iota(M, M^i)$ *is a function possessing the following properties:*

1. $\iota(M, M^i) \geq 0$ *(non-negativity)*

2. $\iota\left(M, M^i\right) = 1 \iff \mathbf{v}\left(M\right) = \mathbf{v}\left(M^i\right)$ *(reflexivity)*
3. $\iota\left(M, M^i\right) = \iota\left(M^i, M\right)$ *(symmetry)* □

For the role of $\iota\left(M, M^i\right)$ we suggest the well-known cosine similarity function, which measures the cosine of the angle between two vectors $\mathbf{v}\left(M\right)$ and $\mathbf{v}\left(M^i\right)$ (summation is performed over all dimensions of $\boldsymbol{V}$):

$$\begin{aligned}\iota\left(M, M^i\right) &= \frac{\mathbf{v}\left(M\right) \cdot \mathbf{v}\left(M^i\right)}{\| \mathbf{v}\left(M\right) \| \cdot \| \mathbf{v}\left(M^i\right) \|} \\ &= \frac{\sum M_{i,j} \times M^i_{i,j}}{\sqrt{\sum \left(M_{i,j}\right)^2 \times \sum \left(M^i_{i,j}\right)^2}}\end{aligned} \tag{7}$$

Given a similarity matrix M, ι induces a natural partial order among ideal vectors with respect to M. Let $\boldsymbol{V} \subseteq \mathcal{V}_{\mathcal{S}}\left(S, S'\right)$ be a subspace, M a similarity matrix over over $\mathcal{S} = S \times S'$, and $\{\mathbf{v}^i, \mathbf{v}^{i'}\} \subseteq \boldsymbol{V}^i$ two ideal vectors of matrices M^i and $M^{i'}$, respectively. Then $\mathbf{v}^i \preceq \mathbf{v}^{i'}$ iff $\iota\left(M, M^i\right) \leq \iota\left(M, M^{i'}\right)$. We can therefore say that $\mathbf{v}^i$ is *minimal* for a given M iff

$$\forall \mathbf{v}^{i'} \in \boldsymbol{V}^i : \mathbf{v}^i \preceq \mathbf{v}^{i'} \tag{8}$$

With such a partial order, we can now formally define an idealizer.

Definition 3 *Let M be a similarity matrix and M^i be a matrix such that:*

- $\mathbf{v}^i = \mathbf{v}\left(M^i\right)$,
- $\mathbf{v}^i \in \boldsymbol{V}^i$, *and*
- $\mathbf{v}^i$ *is minimal for* M.

An idealizer g^i *is defined to be*

$$g^i = \iota\left(M, M^i\right) \tag{9}$$

□

Idealizers presented in this paper share their comparison (Eq. 7) and selection method (using ideal vector minimality), and differ only by their constraint (Γ) functions.

4.3 Generalization

Using the similarity space we can now generate predictors at different levels of granularity, satisfying the generalization characteristic. Granularity of predictors is defined by the subspace over which they provide prediction. In this work we focus on with $K = 1$ predictors. We briefly discusses the special case of $K > 1$ as part of our future work.

Given schemata S and S', *Matrix predictors* predict some quality over a matrix, using vectors of size $|\mathcal{S}|$ (recall that $\mathcal{S} = S \times S'$). *Attribute predictors* evaluate a vector defined over a single attribute, using vectors of size $|S'|$, and *entry predictors* evaluate a single matrix entry. In the rest of this section we define predictors of three granularity levels (matrix, attribute, and entry). In Section 7 we evaluate the effectiveness of the proposed predictors, assessing them in typical usage scenarios against state-of-the-art schema matching techniques as well as their empirical properties as defined in Section 4.1.

4.3.1 Matrix Predictors

The *Binary Matrix Converter* predictor (BMC) is a matrix idealizer, where ideal vectors are all binary vectors. Effectively, the selected ideal vector is created by rounding each dimension of a similarity vector value to the nearest binary value. Such an idealizer is based on earlier works (*e.g.*, [16], Ch. 4.2) showing that non-binary similarity matrices are, in fact, binary matrices masked by some random noise. This predictor prefers schema matching vectors with elements that are either close to 0 or to 1, indicating lower noise and is expected to yield high precision when compared with the exact match, yet may suffer on recall. We note that BMC is naïve, allowing trivial schema matchers, which always return simple $\bar{0}$ or $\bar{1}$ vectors, to score high. A variation of the naïve approach presented above, dubbed the *Binary Matrix Max* predictor (BMM), constrains the number of 1 values in the ideal similarity matrix to be $k = \max\left(|S|, |S'|\right)$. Such a constraint creates a weak link between the size of schemata and the required matches in the idealized vector. In the absence of additional constraints, some attributes may not be matched while others may be matched multiple times. We expect this predictor to support higher recall than the naïve binary matrix converter.

The next predictor tightens the relationship between the schemata at hand and an ideal vector. The *Larger Matched Matrix* predictor (LMM) requires matching **all** attributes from the larger schema. The ideal vector is thus constrained to have at least 1-value entry for each attribute of the larger schema. Let S be the larger schema, then the ideal (binary) vector of LMM must adhere to the following constraint:

$$\sum_{j=1}^{m} v_{i,j} \geq 1 \ \forall A_i \in S \tag{10}$$

In addition to the three idealizers described above we evaluate the following four internalizers:

- *Max* is an internalizer for which the maximum values for each row and column are summed and divided

by the total number of rows and columns. Formally,

$$p_{Max} = \frac{\sum_{i=1}^{n} \max_j (v_i j) + \sum_{j=1}^{m} \max_i (v_i j)}{n+m} \quad (11)$$

– *AvgM* is a simple average of all similarity values in non-zero entries. Formally,

$$p_{AvgM} = \frac{\sum_{i,j|v_{ij}>0} v_{ij}}{\sum_{i,j|v_{ij}>0} 1} \quad (12)$$

– *STDEV* is the standard deviation of the sample, given by:

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(v_i - \mu)} \quad (13)$$

where N is the number of non-zero vector entries and μ is the sample average.

– *DOM* is the dominants predictor described in detail in Section 4.2.1.

4.3.2 Attribute Predictors

Attribute predictors evaluate vectors over a subspace corresponding to a single attribute. We define two attribute predictors that are attribute-level generalizations of the matrix predictors presented above:

– With *Binary M To N Attribute predictor* (BNA), the closest binary vector is created by rounding each element to the closest number in $\{0, 1\}$, the same way BMC works for the whole matrix. We observe that this predictor allows $n : m$ correspondences.

– *One-to-One Attribute Predictor* (1T1) constraints ideal binary vectors to have exactly one element equal to 1. This vector chooses randomly an entry among those that equal $\max(v)$ to 1 and the other entries are set to 0. Thus *1T1* applies a similar constraint as LMM.

In addition to the two idealizers described above we evaluate the following three internalizers:

– *MaxA* is the maximum value of the attribute vector entries:

$$\forall a_i \in S : p_{MaxA_i} = \max_j (v_i j) \wedge \forall a_j \in S' : p_{MaxA} = \max_i (v_i j) \quad (14)$$

– *AvgA* is the average confidence in non-zero attribute vector entries.

$$\forall a_i \in S : p_{AvgA_i} = \frac{\sum_{i,j|v_{ij}>0} v_{ij}}{\sum_{i,j|v_{ij}>0} 1} \quad (15)$$

– *STDEV* is the standard deviation of the sample using Eq. 13.

– *CoVar* is the coefficient of variation, given by dividing standard deviation (σ) by the sample mean (μ) ($\frac{\mu}{\sigma}$).

4.3.3 Entry Predictors

Entry predictors attempt to predict the value of a specific entry, yet surrounding entries can assist in assessing this entry in comparison with its neighbors. The idealizers we evaluate in this work are based on the *dominants* property. For a given attribute pair $\{a_i, a_j\}$, dominance serves as our ideal vector. The following three variations are predictors, based on the similarity in the $\boldsymbol{V_{ij}}$ subspace. Given a schema pair $\{S, S'\}$ and a similarity matrix $M(S, S')$, the *Dominant Binary* predictor (DBN) compares $\boldsymbol{v}(M) \in \boldsymbol{V_{ij}}$ with the vector $v^{i,j}$ (recall that $v^{i,j}$ is a binary vector in which the only element with a value of 1 is the (i, j) element). The *Dominant Set Max* predictor (DSM) compares $\boldsymbol{v}(M)$ with a modified vector where $v_{i,j} = max(\boldsymbol{v})$ and all other elements are left unchanged. Finally, the *Dominant Lower All* predictor (DLA) compares $\boldsymbol{v}(M)$ to a modified vector where all elements larger than $v_{i,j}$ are lowered to equal $v_{i,j}$.

Entry internalizers examine the entry value with respect to some vector properties. As with idealizers, we will use the vector composed of the row and column of the entry at hand. To avoid issues of differing scale between matchers and datasets, we use the following two normalization methods to evaluate the value of an entry w.r.t the surrounding vector:

– *Normalized Value (NV)*: Let $\max(\mathbf{v})$ and $\min(\mathbf{v})$ be the maximum and minimum values of a given vector $\mathbf{v}$ then the normalized value of each entry v_i is given by:

$$\frac{v_i - \min(\mathbf{v})}{\max(\mathbf{v}) - \min(\mathbf{v})} \quad (16)$$

– *Ranked Value (RV)*: Let r_i be the descending rank of element v_i in a given vector. For example, given a vector $(0.1, 0.6, 0.4, 0, 4)$, the corresponding ranks would be $(4, 1, 2, 2)$. Let $R = \max_i (r_i)$. The ranked value of r_i is defined as

$$\frac{r_i - 1}{R - 1} \quad (17)$$

To complete our list of entry predictors we look at a seemingly trivial, but intriguing predictor: *Val*. The *Val* entry predictor simply takes the confidence value assigned by the matcher as a prediction for the quality of the entry. This predictor may actually represent

state-of-the-art in entry predictors as matching systems today inherently assume a correlation between the assigned confidence value and the actual quality (see, for example, the monotonicity principle [**?**]). We evaluate this assumption together with the correlation of other predictors in Section 7.1.

4.4 Tunability

A single predictor may generalize well, however, its tunability is limited. Therefore, we suggest the use of multiple predictors to complement each other in various scenarios. To this effect we now define *prediction models*.

Definition 4 *A* prediction model *is a set of pairs* $\{(g_1,\omega_1),...,(g_k,\omega_k)\}$ *where for each* $1 \leq i \leq k$, g_i *is a predictor and* $\omega_i \in \mathbb{R}$ *is the weight of* g_i *in the model.*□

Loosely correlated predictors can be composed into a model. The weights of its participating predictors should be tuned so that their combined prediction correlates well with the desired quality that we wish to predict. To construct prediction models from individual predictors, you may use the standard statistical method of *stepwise (multi-valued) regression* [13], in which predicting variables (in our case predictors) are added to the model if they improve model correlation with the dependent variable (in our case the quality measure) and removed from the model if they do not contribute to correlation. Each predictor's contribution to the explained variance is measured by the β values of the constructed prediction models. A high absolute β value is interpreted to indicate that a high portion of the variance in the quality measure is explained by this component. It is worth noting that a predictor may not be significant on its own, yet when combined into the prediction model it becomes significant and contributes substantially to model correlation. When performing step-wise regression one should use the *Adjusted* R^2 measure which is adjusted to account for the minuscule increase in R^2 caused by simply adding additional predictors to the model.

5 Putting predictors into action

Having introduced predictors, their formal model, classification, and several examples of each type, we now show, through three different examples, the benefit potential in their usage. Section 5.1 introduces an analysis of schema matchers using entry predictors. We show how attribute predictors can be used to improve matching effectiveness in Section 5.2. Finally, in Section 5.3 we show a case study of deep Web source ranking using multiple predictors.

5.1 Analyziung Second-line Matchers

Entry predictors provide a unique value for each similarity vector element (or similarity matrix entry). For well correlated predictors, this value can be used as an indication of the quality of this entry. We observe that the plethora of second-line matchers designed to-date operate on these indications. Those matchers which are designed as decision makers are required to analyze an entry and decide whether to include it in the final result as a correct correspondence. Previously, when evaluating a new second-line matcher, designers had to make-do with compiling experiment results and comparing precision and recall results of the proposed matcher against other matchers. Little insight was available into the mechanism in which the decision maker chooses matches.

To illustrate the benefit of using predictors to gain insights into second-line matching, consider the *Dominants* property, described in Section 4.2.1. The dominants property was previously used as a decision maker (*e.g.*, [45,16]) by simply selecting those entries that demonstrate this property for the final result. To better understand the dominants property we examined the distribution of the NV entry predictor values (see Section 4.3.3) for true matches, *i.e.*, where the exact match vector element value is 1.0, versus non-matches. Recall that the NV predictor gives the normed value of an entry with respect to its surrounding vector (row and column of the entry). Thus, for a fully dominant entry the normed value is 1.

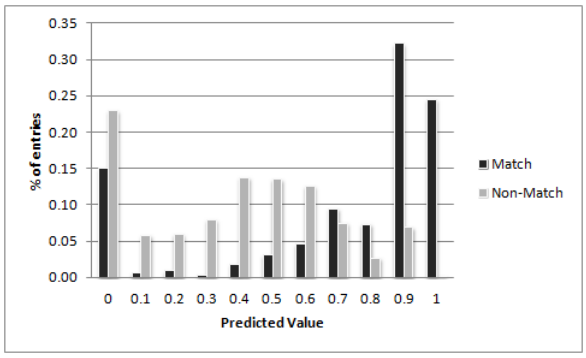


Fig. 3: Entry Predictor Distribution on Matches

Figure 3 presents the distribution of true matches and non-matches with respect to the predicted values for 32,000 entries from eight similarity vectors. For a

complete description of our empirical setup, see Section 7. Since the dominants property requires a value to fully dominate its row and column entries, it identifies correctly only about 25% of the true matches (and none of the non-matches). Moreover, Figure 3 highlights the observation that high values (indicating high, although not absolute, dominance) also indicate correct correspondences. Therefore, the use of dominants in its strictest form works well towards precision but does poorly on recall. A more judicial use of the dominants property could be to select those pairs with dominance of over 0.9, accounting for almost 60% of the correct matches. Thus, our analysis of entry predictors leads to better strategies in second-line matching, namely: choosing pairs with 90% dominance rather than 100% dominance.

5.2 Improving Match Effectiveness

Schema 1	Schema 2
First Name:	* First Name:
Last Name:	* Last Name:
Gender:	* Gender:
What are your favourite Hobbies...	List the causes...
Requested Password:	* Password:
Password Confirmation:	* Retype Password:
Yes	I agree to the terms
REGISTER	Sign Up Now

Table 4: Exact Match Example

Table 4 lists some terms from an exact match between two schemata, extracted from Web forms in the *forums* domain. For the majority of cases, a simple term matcher suffices to identify the correct correspondences with relative ease. Therefore, such a matcher is expected to score high with matrix predictors. However, for attribute pairs such as (Yes, I Agree to the terms) and (REGISTER, Sign Up Now), a term matcher would have no insight while a data type matcher would correctly identify that both attributes in the first pair are check mark controls and in the second pair command buttons.

Table 5 illustrates the outcome of comparing the REGISTER attribute with all other attributes in the second schema for both a term and data type matchers. In such scenarios, it would be best to use attribute predictors, leading to more emphasis on the Data Type matcher vector for these two attribute pairs.

Attribute 1	Attribute 2	Term	Data-type
REGISTER	* First Name:	0	0
REGISTER	* Last Name:	0	0
REGISTER	* Gender:	0.1	0
REGISTER	List the causes...	0.01	0
REGISTER	* Password:	0	0
REGISTER	* Retype Password:	0.01	0
REGISTER	I agree to...	0.01	0
REGISTER	Sign Up Now	0	1

Table 5: Correspondence vectors

Section 7.1 provides an empirical support to the example shown above, demonstrating the superiority of attribute level prediction over matrix level prediction.

5.3 Data-source Lookup

To illustrate the use of multiple vectors in predictors we show a method where several prediction methods are combined to improve performance. The strategy, named *mult*, seeks to exploit the notion of unique domain expressions as an indicator for relevance. In the field of Information Retrieval, it is a common practice to place higher emphasis on those terms that are unique to a document with respect to the document corpus using measures such as *TF/IDF* in which the weight of a term is reduced if the term is abundant in other documents and thus rare terms have a higher impact in document comparisons. Applying a similar approach to data-source lookup requires quantification of the relative rarity of attributes in the schema corpus. We do so by using attribute predictors.

In the proposed method, prediction values are combined using a linear transformation of the similarity space, such that dimensions of attributes covered by many schemas are de-emphasized. This process is followed by applying a matrix prediction using the new dimensions to rank the candidate schemata by descending predicted quality of their similarity vectors with the target schema.

Algorithm 1 provides details of the *mult* method. Similarity vectors are first generated by matching target schema S_0 with each of the candidate schemata, using a $Match$ operator (lines 3-5).

In lines 6-10, an attribute-schema prediction matrix is constructed by predicting the precision of each candidate schema S_j with respect to each attribute A_i in the target schema (using $APredictPrecision$). Prediction is done using an attribute predictor model tuned on precision. To illustrate this step, consider Table 1(a) of Example 1, taking S_0 to be the target schema and S_1 to be a candidate schema together with four other

Algorithm 1 *mult*

1: **Input:** S_0: target schema ; $(S_1, S_2, \ldots, S_m)$: candidate schemata
2: **Output:** $\left(S_{(1)}, S_{(2)}, \ldots, S_{(m)}\right)$: ranked list of candidate schemata
3: **for** $j = 1 \rightarrow m$ **do** ▷ Match schemata
4: $\quad \mathbf{v}_j \leftarrow Match\left(S_0, S_j\right)$
5: **end for**
6: **for** $i = 1 \rightarrow n$ **do** ▷ Attribute prediction
7: $\quad$ **for** $j = 1 \rightarrow m$ **do**
8: $\quad\quad p_{ij} \leftarrow APredictPrecision\left(A_i, \mathbf{v}_j^{A_i}\right)$
9: $\quad$ **end for**
10: **end for**
11: **for** $i = 1 \rightarrow n$ **do** ▷ Compute reduction factor
12: $\quad w_i \leftarrow \sum_{j=1}^{m} p_{ij}$
13: **end for**
14: **for** $i = 1 \rightarrow n$ **do** ▷ Normalize weights
15: $\quad w_i \leftarrow \frac{w_i - \min(\mathbf{w})}{\max(\mathbf{w}) - \min(\mathbf{w})}$
16: **end for**
17: **for** $j = 1 \rightarrow m$ **do** ▷ Transform and predict
18: $\quad$ **for** $i = 1 \rightarrow n$ **do**
19: $\quad\quad \bar{\mathbf{v}}_j^{A_i} \leftarrow \mathbf{v}_j^{A_i} \cdot w_i$
20: $\quad\quad pS_j \leftarrow MPredictPrecision\left(\bar{\mathbf{v}}_1, \ldots, \bar{\mathbf{v}}_m\right)$
21: $\quad$ **end for**
22: **end for**
23: **return** $\left(S_{(1)}, S_{(2)}, \ldots, S_{(m)}\right) \leftarrow Rank\left(\{S_1, \ldots, S_m\}, \mathbf{pS}\right)$

schemata (not shown here). Assume, for sake of simplicity, the use of *MaxA* as the attribute predictor (Eq. 14). Follows this step of the algorithm, each column in Table 1(a) is reduced to an entry in the S_1 column of Table 6.

Target attribute	S_1	S_2	S_3	S_4	S_5
cardNum	0.84	0.83	0.72	0.11	0.13
city	1.00	0.82	0.89	0.68	0.82
arrivalDay	0.35	0.14	0.15	0.73	0.21
checkInTime	0.64	0.23	0.34	0.46	0.13

Table 6: Attribute-schema prediction matrix example

Normalized dimensional reduction weights are generated in lines 11-16 by taking the L_1 (Taxicab) norm of each row as a reduction factor for the corresponding target attribute. For example, in Table 6 city weights are reduced to a great magnitude since all schemata cover it well, resulting in greater relevant importance to attributes such as arrivalDay and checkInTime, which are more representative of the hotel search domain. Therefore, the weight of city values will be reduced to zero while arrivalDay, with the lowest L_1 norm, will retain the weight of 1. Similarly, cardNum will be reduced by .397 while checkInTime remain relatively unchanged with a weight of .874. Table 7 represents the weighted version of Table 1(a) in which each entry is multiplied by the target term weight.

$S_0 \longrightarrow$ / $\downarrow S_1$	1 cardNum	2 city	3 arrivalDay	4 checkIn Time
1 clientNum	0.33	0.0	0.32	0.26
2 city	0.12	0.0	0.33	0.26
3 checkInDate	0.13	0.0	0.35	0.56

Table 7: Weighted Similarity Matrix

Finally, in lines 17 to 23, each similarity vector is transformed using the new weights and a matrix predictor model is used to rank the new vectors and return an ordered set of candidate schemas. The de-emphasis of city and cardNum makes S_1 and S_4 the highest ranked candidate schemas due to their strong coverage of arrivalDay and checkInTime.

The *mult* method is put into use in Section 7.4, where we experiment with it to rank Web-form schemata using various ranking strategies.

6 Empirical Evaluation Method

In this section we present our empirical evaluation method. The setup of our experiments and the measures used are detailed in Section 6.1. The evaluation methodology is discussed in Section 6.2. Section 7 is devoted to the experiment results and analysis.

6.1 Experimental setup

Evaluations were conducted on an Intel(R) Core(TM)2 Quad CPU Q8200 @ 2.33GHz. An experiment system was coded in Java, JDK version 1.6.0. JVM was initiated with 2.00GB heap-memory.

We used three datasets with very different characteristics to facilitate the evaluation of robustness. All datasets are available for download.[3] *Webform* is a dataset containing 247 schemata, automatically extracted from Web forms using the OntoBuilder extractor. Each schema was classified to one of 21 domains (Table 8). There are 60,762 possible schema pairs, out of which an exact match was defined manually for 149 pairs. Web forms are small, with 10-30 attributes each and an overwhelming majority of 1 : 1 correspondences.

The *P.O.* (Purchase order) dataset was introduced by Madhavan et. al. [26] and extended by our group. The dataset contains 12 XML documents describing

[3] https://bitbucket.org/tomers77/ontobuilder-research-environment/downloads/datasets.zip

DomainID	DomainName	Web-Forms
1	Aviation	13
2	Betting	12
3	Book Shops	6
4	Car Rental	14
5	Complaint Form	12
6	Cosmetics	4
7	Date Matching	24
8	Forums	16
9	Fan Clubs	12
10	Hotel Search	10
11	Hotels	30
12	Job Find	5
13	Magazine Subscription	19
14	Moving	2
15	News	7
16	Search Engines	5
17	Shoe Stores	12
18	Ticket Stores	2
19	Vacation Time-Sharing	2
20	Web-Mail	24
21	Finance	16

Table 8: Web-form domains

purchase orders extracted from various systems and matched into pairs. Schemas are medium scaled with 50-400 attributes each. Some attributes have a $1:n$ match.

The *Univ.* (University) dataset contains 15 university application forms from various US universities, collected as part of the NisB project[4] and converted to XSD. Schemas are medium scaled with 50-150 attributes each. 180 exact correspondences were identified manually overall. Cases of $n:m$ matching are frequent in this dataset.

We used a variety of matching algorithms from two matching systems, namely AMC and OntoBuilder, as follows. Auto Mapping Core (AMC) [35] is an SAP schema matching research prototype. Its matchers are designed to explore various features and dependencies that exist in business schemata. We used the following AMC matchers in the evaluations: NAME, TOKEN-PATH, DATA-TYPE, PATH and SIBLING. OntoBuilder is a research prototype, developed for matching schemata that represent deep Web data. The following OntoBuilder first-line matchers were used in the evaluation (for a detailed description, see [16]): Term, Value, Composition (Graph), Precedence, and Similarity Flooding (a re-implementation of the algorithm presented in [31]). The decision makers used for all matchers are Stable Marriage(SM),Threshold(t) which simply chooses all vector elements such that $v_{ij} \geq t$; and MaxDelta(δ) which takes for each row / column attributes which are within δ of the max value in that row / column.

[4] http://www.nisb-project.eu/

6.2 Evaluation Methodology

Evaluating evaluation measures poses a unique challenge. Instead of comparing the outcome of a matcher to some exact match (using precision and recall for example), predictors need to be evaluated on the quality of their, well, prediction. In order to evaluate the usefulness of our proposed predictors we require statistical measures which were explained in detail in Section **??**. Also, to ensure validity, evaluation is done using unbiased instance selection protocols as described herein.

6.2.1 Quality Measures

Whenever predictors are used to improve results of schema matching, they are evaluated using match quality measures. When matching two schemata we use precision (P) and recall (R) with the Harmonic F-Measure ($F1$) as a combined schema-pair measure. Extending such measures to varying levels of granularity requires careful analysis.

To evaluate an attribute predictor (defined in Section 4.3.2) we require a quality measure for a sub-space defined over a single similarity matrix row. Using the vector space representation of these measures (Eqs. 3 and 4) makes this extension trivial by limiting the result vector $\mathbf{v}$ and the exact match vector $\mathbf{v}^e$ to those dimensions corresponding with the required attribute.

Taking the same approach towards entry level predictors proves to be problematic. When reducing the vector $\mathbf{v}^e$ to a single dimension and using Eq. 4, vector-based Recall is reduced to one (1.0) if the entry is true and ∞ (division by 0) if the entry is false. Even if false entry prediction value is set to some arbitrary value, these results hamper regression methods by introducing non-smooth variables. To overcome this limitation we propose to evaluate entry predictors by classifying each entry into one of the four set-comparison options:

True Positive (TP) if both match result and exact match are true.
False Positive (FP) if match result is true and exact match is false.
True Negative (TN) If both match result and exact match are false.
False Negative (FN) If match result is false and exact match is true.

Distribution of prediction values in each of these categories can then be compared. We demonstrate both

visual comparison and statistical analysis of distribution parameters (mean and standard deviation) in Section 7.1.

(a) Macro vs. micro averaging, part I

Pair	$\mathbf{v}\cdot\mathbf{v}^e$	$\|\mathbf{v}\|$	$\|\mathbf{v}^e\|$	P	R
1	5	10	6	0.50	0.83
2	25	50	60	0.50	0.42
3	2	2	3	1.00	0.67
Average				0.67	0.64
Median				0.50	0.67
μAvg				0.52	0.46

(b) Macro vs. micro averaging, part II

Pair	$\mathbf{v}\cdot\mathbf{v}^e$	$\|\mathbf{v}\|$	$\|\mathbf{v}^e\|$	P	R
1	5	10	6	0.50	0.83
2	26	50	60	0.52	0.43
3	1	2	3	0.50	0.33
Average				0.51	0.53
Median				0.50	0.43
μAvg				0.52	0.46

Table 9: Macro vs. micro averaging

Aggregation of match quality results over multiple experiments should also be done with care. Using a simple average over P, R, and $F1$ is sensitive to outliers. Replacing average values with median values deal with outliers better but is still considered *macro-averaging*, giving an equal weight to each schema pair and thus rendering it sensitive to differences in pair properties. To demonstrate the drawbacks of macro-averaging, consider Table 9. Each of the two tables contains the outcome of a match experiment where a matcher configuration is run on three pairs. Each pair is represented by a row in the table. For each pair, P and R are calculated using their constituent vector components. Careful comparison of the results reveals that the performance of the two configurations is identical on the first pair and differs by one match on the second and third pair. Macro-averaging measures (average and median in this example) are highly skewed towards the smaller schema pair. This is highlighted by this example as the loss of one match in the smaller pair by the second configuration is heavily penalized both in terms of P and R by averaging and for R only when using median. This loss is not compensated by the gain of a match in the larger schema pair.

Micro-averaging measures give equal weight to each attribute pair. In Table 9, micro-averaged P and R are calculated over the sum of the vector components and the result is presented in the μAvg row. As far as the micro-averaged measures are concerned, the loss of one match in the third pair is exactly offset by the gain of another match in the second pair. However, micro-averaging is thus skewed towards larger schema pairs that dominate the result of a collection.

Given the biases in micro- and macro-averaging, in our ensemble experiments we chose to employ ranking as a method to report on the relative quality of each method with respect to all other methods in the experiment. Ranking prevents biases towards larger schemata to which macro-averaging is sensitive and still gives each pair an equal weight, unlike micro-averaging. Ranking is, therefore, preferred for comparing methods over diverse pairs from several datasets.

Data source discovery experiments require different quality measures. In these experiments, schemata are retrieved from a collection and judged for relevance. Though it may be argued that relevance is not a binary property, for the sake of simplicity we use binary relevance and can therefore employ standard relevance quality metrics, originating from the field of Information Retrieval. The first family of metrics is *Precision@K (P@K)* (with $K = 1, 5, 10$). P@K is used to judge a single experiment, in which a ranked list of candidate schemas is presented for a given target schema. For candidate schema S_i, let $I(S_i)$ be an indicator such that $I(S_i)$ equals 1 iff S_i is relevant and 0 otherwise, then:

$$P@K = \frac{\sum_i I(S_i)}{K} \tag{18}$$

While P@K is simple to calculate and understand, it is often criticized for being limited by the arbitrary selection of K. Furthermore, when comparing several experiments, an underlying assumption is that the proportion of relevant and irrelevant examples is maintained the same over all experiments. Since in our case, proportions vary (between 0.015 and 0.12), we report results using the *R-Precision (R-Pr)* measure as well. R-Pr judges an experiment by setting the parameter K in Eq. 18 to be the number of relevant samples in the collection. R-Pr values are often lower than P@K values as the number of relevant samples is often higher than typical K choices.

7 Evaluation Results

Following our characterization of desired predictor properties we evaluate the empirical properties (correlation and robustness) of the predictors and prediction models presented in Section 4 and the practical applications of prediction presented in Section 5.

In Section 7.1 we evaluate the correlations between the proposed matrix and attribute-level predictors and

various quality measures. All predictors were found to be significantly correlated with both precision and recall yet attribute level predictors were better correlated with their quality measures than matrix predictors. Correlation tests are accompanied by significance tests on various datasets and matching algorithms to evaluate robustness. Section 7.2 evaluates prediction models, our proposed solution to tunability, by evaluating their correlation with different qualities. Prediction models are shown to be superior to individual predictors both by virtue of their ability to be tuned towards different quality measures and by presenting improved correlation with these measures. The rest of the section evaluates the three usage examples for prediction detailed in Section 5: Entry predictors are shown to be useful in improving match selection of in Section 7.3. Relevant data sources are shown to be accurately ranked using attribute and matrix predictors in Section 7.4. Finally, in Section 7.5 attribute predictors are used for ensemble construction and the approach is shown to outperform the state-of-the-art technique that uses matrix predictors.

7.1 Correlation and Robustness

In this section we evaluate the correlation and robustness of matrix and attribute-level predictors. Matrix-level correlation experiments were conducted on 810 similarity vectors. 270 vectors were generated by applying nine first-line matchers from both AMC and OntoBuilder on 30 schema pairs randomly selected from the three datasets. On each vector, matrix predictors (described in Section 4.3.1) of which three are idealizers and four are internalizers were applied. Vectors were then selected using one of three second-line matchers (Threshold(0.5), MaxDelta(0) and MaxDelta(0.05)) generating binary vectors on which P and R were calculated.

Table 10 summarizes correlation results between matrix predictors and quality measures. With respect to P, all predictor correlations, excluding LMM and Max, were significant ($p < .001$), meaning that the correlation observed is with high probability a real phenomenon rather than random noise. Recall that, according to Cohen [7], correlation values over .01 represent a small effect, over .09 a medium effect and over .25 a large effect. Predictors best correlated with P are (in descending order): DOM, BMM, AvgM, BMC, and STDEV. The negative correlation between STDEV and P means that increased values of STDEV predict lower quality in terms of P. While correlation values are low, they are significant over a large and diverse dataset. We improve on these results using prediction models in the following section. As for R, all predictor correlations are found significant. Correlations are substantially higher, with STDEV, Max, and LMM presenting correlation values over .2 and DOM, AvgM, BMM, and BMC presenting values around .05. The fact that some predictors are well correlated with P, while others with R is encouraging in the context of prediction models, as it allows us to construct models that are tuned towards different quality measures.

Predictor	Measure	P	R
BMC	R^2	.039	.054
	sig.	.000	.000
BMM	R^2	.049	.065
	sig.	.000	.000
LMM	R^2	.000	.222
	sig.	.925	.000
STDEV	R^2	-.037	.322
	sig.	.000	.000
AvgM	R^2	.043	.080
	sig.	.000	.000
Max	R^2	-.002	.238
	sig.	.220	.000
DOM	R^2	.126	.083
	sig.	.000	.000

Table 10: Matrix predictor correlation (N=810)

To further understand the relationships among predictors, we performed inter-correlation analysis. Predictors that are highly correlated may be redundant, while low correlation between predictors may be used to identify predictors that base their prediction on different phenomena in the data.

Results are presented in Table 11. Recall that significance values of less than .01 are desirable, showing the correlation values are significant. The strong correlation between BMC, BMM, and LMM is to be expected as they are consecutive refinements of the same principles. The weak correlation between DOM and all other predictors is encouraging as it indicates that it may be combined successfully with other predictors to create an improved prediction model. In general, internalizers seem to be weakly correlated with idealizers, which is, again, encouraging. The large number of significant correlations (significance of less than 0.01) is an expected result and can be traced to their common correlation with the quality measures.

A similar protocol to the one described above was followed for attribute predictors. Table 12 summarizes correlation results for attribute predictors over 7708 attribute vectors from five schema pairs matched using four different matchers from OntoBuilder and AMC. All predictors were significant ($p < .05$). Results show that all predictors correlate well with the quality measures. $1T1$ is the best matcher to correlate with P while

Predictor	Measure	BMC	BMM	LMM	STDEV	AvgConf	Max	DOM
BMC	R^2	1	.692	.447	.137	.446	.107	.048
	sig.		.000	.000	.000	.000	.000	.000
BMM	R^2	.692	1	.656	.186	.606	.217	.119
	sig.	.000		.000	.000	.000	.000	.000
LMM	R^2	.447	.656	1	.400	.398	.452	.133
	sig.	.000	.000		.000	.000	.000	.000
STDEV	R^2	.137	.186	.400	1	.388	.537	.000
	sig.	.000	.000	.000		.000	.000	.621
AvgM	R^2	.446	.606	.398	.388	1	.288	.017
	sig.	.000	.000	.000	.000		.000	.000
Max	R^2	.107	.217	.452	.537	.288	1	.144
	sig.	.000	.000	.000	.000	.000		.000
DOM	R^2	.048	.119	.133	.000	.017	.144	1
	sig.	.000	.000	.000	.621	.000	.000	

Table 11: Matrix predictor inter-correlation (N=810)

Predictor	Result	P	R
1T1	R^2	.32	.14
BNA	R^2	.21	.45
AVG	R^2	-.08	.20
MAX	R^2	.20	.51
STDEV	R^2	.24	.34
CoVar	R^2	.15	-.04

Table 12: Attribute predictor correlation (N=7708)

Step	Added	Removed	Adj. R^2	f^2 (Effect Size)
1	DS		.038	0.04
2	STDEV		.342	0.46
3	DOM		.409	0.11
4	AvgM		.424	0.03
5	LMM		.434	0.02
6	BMM		.444	0.02
7		AvgM	.444	0.00
8	Max		.447	0.01

Table 13: R matrix-predictor model construction

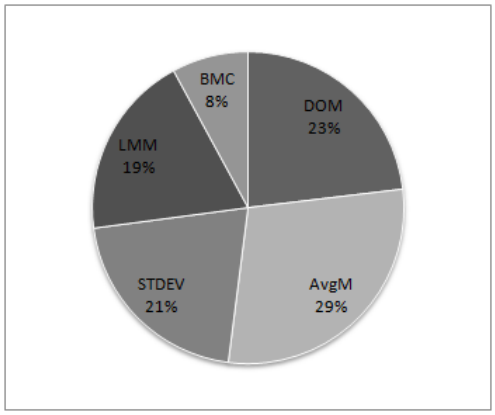
MAX is the best matcher to correlate with R. We note that AVG has negative correlation with P and positive correlation with R. Inter-correlation results (omitted for brevity) are similar and indicate that idealizers are distinctly different from internalizers.

7.2 Tunability using Prediction Models

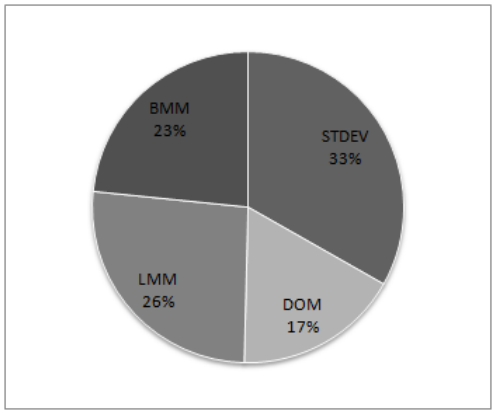
We now present an evaluation of the prediction model construction method, presented in Section 4.4. To measure robustness of predictors over dataset types, step-wise regression was initialized with a dummy variable indicating dataset type (-1 for small web forms, +1 for large XML-Schema datasets: purchase-order and university). We then added, one by one, predictors in a decreasing order of explained variance (R^2) and remove predictors rendered insignificant or non-contributing by the addition of a predictor. At each step, the model's *Adjusted* R^2 and significance are measured. Table 13 presents the R matrix-level prediction model evolution, step by step. The effect size of adding (or removing) each predictor is measured using *Cohen's* f^2 [7] which measures the incremental effect of adding a predictor in multiple regression. An accepted convention is that effect sizes of 0.02, 0.15, and 0.35 are considered small, medium, and large, respectively.

Examining results of the step-wise regression leads to several conclusions. The small effect size of the dataset (DS) dummy variable and the large effect size of the predictor entered directly after it, lead to the conclusion that the correlation of the prediction model is robust with respect to dataset type, since the contribution of the schema size to explaining the variability in the results is small (0.04). Similar results were obtained when controlling for matching system and during the construction of the P prediction model. The two major contributors to prediction model correlation are STDEV and DOM, which is expected due to the large individual correlations. The addition of BMM caused AvgM to become insignificant in the model, this is due to the substantial inter-correlation between these two variables. The minuscule effect size of adding Max allows us to settle for the model after step 7 as our final model.

The weight of each predictor within the model is illustrated in Figure 4. The composition of the constructed matrix-level prediction models for both P and R is given in Figure 4(a) and Figure 4(b), respectively. Though it may seem that STDEV has the same role in both models, it should be noted that in the precision model its weight is with a negative sign (increased STDEV values predict decreased P values and increased



(a) P Prediction Model



(b) R Prediction Model

Fig. 4: Matrix-Prediction Model Composition

R values). It is worth noting that the individual performance of each of the predictors is not the only measure when determining the prediction ability within a prediction model. Therefore, even though LMM has an insignificant individual correlation with P, it still contributes substantially to the P prediction model. This phenomenon indicates that LMM contributes to refine the predictions made by the other predictors, by separating those cases in which the combined score is the same but P still differs. Thus, using prediction models allows us to tune our prediction separately towards different quality measures.

Granularity	Measure	Model R^2	Best Predictor R^2
Matrix	R	0.44	0.32
Matrix	P	0.35	0.13
Attribute	R	0.67	0.51
Attribute	P	0.47	0.32

Table 14: Prediction Models R^2

Combining the various predictors using prediction models can also be used to boost prediction ability. Table 14 presents the correlation (labeled **Model** R^2) of the different granularity level prediction models showing a significant improvement over best individual predictor correlation (rightmost column). Here, as in the individual predictor correlations, it is obvious that attribute-level predictor models are better correlated with the predicted quality than matrix-level predictors and that R models are better correlated than P models. Figure 5 provides a visualization of prediction model correlation for the matrix-level R prediction model. It shows a clear correlation, where increased R prediction (X-axis) is accompanied by an increased R value (Y-axis). We can conclude that prediction models correlate well with the measure that they intend to predict. Their correlation is much higher than that of individual matchers and taking into account the large amount of schema-pairs used (810 and 7708 for matrix and attribute predictors respectively), represent a high degree of correlation.

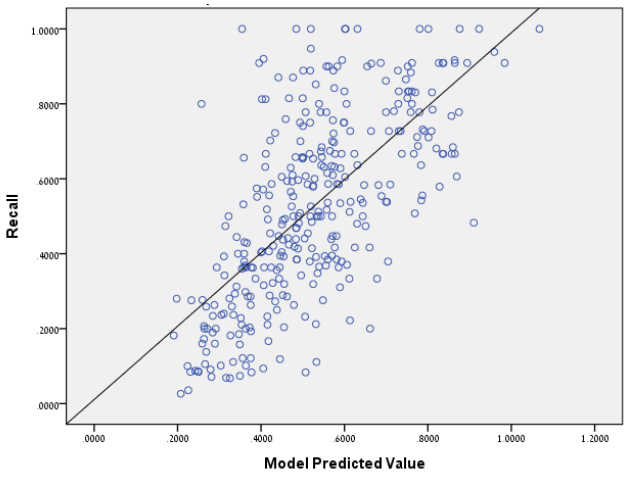


Fig. 5: R Model Correlation

7.3 Schema Matching Design Application: Entry Predictors

In this section, the usage of entry-level predictors as a basis for second-line matchers is evaluated. As explained in Section 5.1, the correlation between entry predictors and quality measures can be exploited to perform entry level tasks such as match selection.

Correlation of entry predictors is examined via a comparison of predictor value distribution over the four set-inclusion categories (True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN)).

A well correlated entry predictor should exhibit higher values for TP versus FP and for FN versus TN. The difference in distributions allows schema matching tasks to perform better in match selection. To examine correlation, four schema pairs (two from the *Web-Form* dataset and two from the *Purchase Order* dataset) where matched using four first-line matchers: Term and Precedence from the Ontobuilder matching system, and TOKENPATH and SIBLING from the AMC matching system. Each entry of the 16 vectors was given a prediction score using all entry predictors. Subsequently, each vector was fed into a decision-maker second-line matcher and the result compared to the exact match vector to allow each entry to be assigned to one of the four set-inclusion categories. The process described generated 106,700 examples for which the distribution of prediction values could be compared across set inclusion categories.

To illustrate the notion of distribution comparison, consider figures 6 and 7, comparing the distribution of three entry predictors described in Section 4.3.3, namely Normed Value (NV), Ranked Value (RV), and Value (Val) over two pairs of set-inclusion categories: TP versus FP in Figure 6 and TN versus FN in Figure 7.

The difference in distributions between categories can be exploited to improve matching results. For example, to improve P in the system examined here we can filter out positive results by removing those with a normed value of 0.9 or less. The gain in P should be substantial with only a minor loss of R. Similarly, improving R can be achieved by adding results with a value higher than 0.6 or a normed value of over 0.8. These distributions remain stable when adding and removing pairs from different datasets, suggesting that the behavior is inherent in the matchers used in this case. To evaluate which predictors present better correlation we use a two-tailed t-test between the means of TP and FP and between the means of TN and FN. Recall that a lower t-test significance value (sig.) indicates a lower probability that the difference in means emanates from random noise, and therefore represents a higher significance of the result.

Reviewing the results in Table 15(a), all predictors, excluding DSM, present significance over 99%. While DBN, DLA and Val have bigger differences in means, the larger standard deviation indicates a substantial overlap between the distributions. Following this line of thought makes NV a preferred candidate with a substantial part of the distribution non-overlapping. Results for the negative categories (Table 15(b)) are similar with DSM the only non-significant predictor and NV showing the best separation.

(a) Positive Categories

Predictor	Cat.	Mean	Std. Deviation	sig.
DBN	TP	.397	.403	.000
	FP	.147	.204	
DSM	TP	1.000	.001	.588
	FP	1.000	.001	
DLA	TP	.672	.389	.000
	FP	.872	.251	
NV	TP	.992	.033	.001
	FP	.974	.064	
RV	TP	.999	.005	.000
	FP	.994	.020	
Val	TP	.604	.143	.010
	FP	.554	.159	

(b) Negative Categories

Predictor	Cat.	Mean	Std. Deviation	sig.
DBN	TN	.039	.026	.000
	FN	.061	.069	
DSM	TN	.946	.129	.703
	FN	.943	.137	
DLA	TN	.979	.084	.002
	FN	.954	.123	
NV	TN	.393	.292	.000
	FN	.661	.370	
RV	TN	.571	.327	.000
	FN	.827	.257	
Val	TN	.297	.186	.000
	FN	.398	.248	

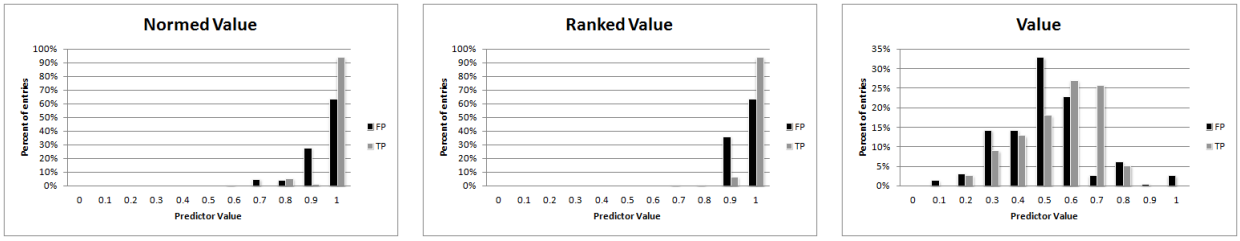
Table 15: Predictor Distribution Parameters

7.4 Deep Web Application: Data Source Discovery

In the following experiment we used prediction to assess the relevance of schemata that represent deep Web data sources. Recall that the *Web-Form* dataset contains 247 such schemata, separated into 21 domains (Table 8). Each experiment entailed choosing one such domain and for each schema in this domain (dubbed the target schema) perform schema matching with all other schemata and calculating various predictors. Predictor results where used to subsequently rank the list of 246 retrieved schemas by decreasing order of predicted relevance. Each (candidate) schema in the list was assigned a binary relevance score, based on whether the candidate schema was of the same domain as the target schema. Table 16 provides an example, where the top-10 predicted matches are assigned with a binary relevance (last column). P@1, P@5, P@10 and R-Pr were calculated on the results.

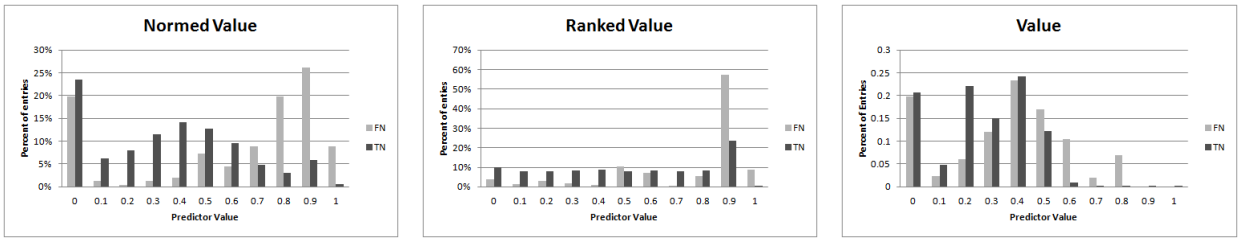
Evaluation compared the following prediction strategies:

– **R Matrix Prediction Model (*RMat*)**: Candidate schemata were ranked by a matrix prediction model. The model was constructed using step-wise regression targeting R. Regression was performed



(a) Normed Value (b) Ranked Value (c) Value

Fig. 6: Entry Predictor Distribution on Positive Entries



(a) Normed Value (b) Ranked Value (c) Value

Fig. 7: Entry Predictor Distribution on Negative Entries

Target	Target Domain	Candidate	Prediction Value	Candidate Domain	Relevant
regman.freeze.com	WebMail	mail01.mail.com	0.62	WebMail	1
regman.freeze.com	WebMail	www.postmaster.co.uk	0.60	WebMail	1
regman.freeze.com	WebMail	fanclub.wd40.com	0.57	FanClubs	0
regman.freeze.com	WebMail	login.myspace.com	0.56	Forums	0
regman.freeze.com	WebMail	registration.excite.com	0.54	WebMail	1
regman.freeze.com	WebMail	www.arabia.com	0.55	WebMail	1
regman.freeze.com	WebMail	www.bbking.com	0.54	FanClubs	0
regman.freeze.com	WebMail	www.linuxmail.com	0.54	WebMail	1
regman.freeze.com	WebMail	auth.ivillage.co.uk	0.54	WebMail	1
regman.freeze.com	WebMail	wp.eurobet.com	0.50	Betting	0

Table 16: Retrieval example with relevance judgement

over 149 web-form schema pairs for which an exact match is available (out of 60762 pairs participating in the experiment).

- **P Matrix Prediction Model (*PMat*)**: Candidate schemata were ranked by a prediction model, constructed in the same manner, but targeting P rather than R as in *RMat*.
- **Multi Predictor (*mult*)**: For each target schema, all candidates where matched with result vectors fed into the multi-predictor method as detailed in Algorithm 1.
- **Combined Predictor (*Comb*)**: A simple average of *PMat* and *mult* ranks was used, after scaling *mult* to match the value distribution of *PMat*.
- ***Pmat* Top30 Re-Ranked by *mult* (*RR*30)**: Candidate schemata were ranked by *PMat* and the top 30 results where consequently re-ranked by *mult*.

The strategies detailed above represent an initial foray into the task of data source ranking and are, by no means, an exhaustive exploration of all prediction methods or prediction score combination methods. We defer additional exploration to future work. Due to the sensitivity of $P@K$ to the amount of relevant documents, $P@K$ for a given K was calculated only for domains with at least $1 + 2K$ schemata.

Strategy	Pr@1	Pr@5	Pr@10	R-Pr
RMat	.113	.095	.125	.079
PMat	**.631**	.472	.457	.246
Mult	.577	.482	.442	.251
Comb	**.631**	**.518**	.500	**0.268**
RR30	.626	.511	**.538**	.260

Table 17: Performance of alternative prediction strategies

Table 17 presents the results of this set of experiments. Overall strategy results are micro-averaged (each target schema is equally weighted). While no strategy is dominant on all measures, *Comb* and *RR30*, which combine the P prediction model ($PMat$) with the multi-predictor ($mult$) outperform independent usage of either one. The striking failure of $RMat$ is probably due to the disparity between the quality measure on which it is tuned and the measures targeted in this application.

As a concluding remark we note that a closer examination of Table 16 reveals some of the limitations of the relevance assignment method that was employed. For example, consider *myspace.com*, assigned to the *Forums* domain, and therefore judged as irrelevant to *freeze.com*, which is assigned to the *Web-Mail* domain. At the time of the crawl and domain assignment, *freeze.com* was a Web-portal providing free email and targeted promotions to this email. Registration to freeze, as reflected by the extracted schema, contained details such as "Marital Status", which may seem irrelevant for a Web-mail service but is reasonable for targeted promotions. This and other similar form fields cause freeze to seem similar to a social network such as myspace. It is therefore acknowledged that actual relevance is uncertain to some extent. Therefore, performing multi-judge non-binary relevance judgments may prove to be more realistic. In the context of this observation, P results should be considered as conservative.

7.5 Schema Matching Application: Dynamic Ensemble Weighting

In this experiment we evaluate the performance of a prediction model based on attribute predictors to the dynamic weight setting of matcher ensembles. Recall that in Section 4.1 we desired predictors to be generalizable to different granularity levels. We exploit this property of well-designed predictors to provide predictions on single attributes, thus allowing the matching system to dynamically construct a different matcher ensemble for each attribute. We have compared the performance of 10 individual matchers with two methods for dynamic ensemble weight setting on 25 randomly selected schema pairs from the three datasets. Each ensemble is combined of the 10 matchers with weights being set dynamically using the specific weighing scheme. AWeighted uses a prediction model with attribute predictors. MWeighted uses DOM, a matrix predictor, to set weights for the different matchers. MWeighted represents the current state-of-the-art, as was presented by Mao et. al. [30].

Matcher	WebForm	P.O.	Univ.	Total
AWeighted	2.9	**2.3**	**1.8**	**2.4**
MWeighted	2.4	3.6	3.5	3.1
Term Match	**1.6**	5.3	3.0	3.3
AMC Name	2.9	3.7	3.3	3.3
AMC Token Path	4.1	4.0	2.5	3.7
AMC Path	2.7	4.8	6.0	4.2
Precedence	3.0	6.3	4.2	4.5
Graph	4.4	7.2	5.7	5.7
AMC Sibling	5.9	8.9	6.3	7.1
Value Match	6.5	10.7	8.2	8.4
AMC DataType	7.4	9.7	8.3	8.4
Similarity Flooding	8.0	10.8	8.2	9.0

Table 18: Average Rank of Matchers on R

Table 18 summarizes the results of the experiment. For each matcher, the average rank of its R result with respect to all other matchers on the same schema pair is calculated. Tie breaking was done using P. Best performer results are marked with boldface. AWeighted is ranked, on average, higher than MWeighted. For the *Web-form* dataset, the Term matcher proves to be very dominant and no combination of matchers can defeat it.

Matcher	WebForm	P.O.	Univ.	Total
AWeighted	9	**8**	**6**	**23**
MWeighted	**10**	7	5	22
AMC Name	8	7	4	19
Term Match	**10**	3	5	18
AMC Token Path	6	5	**6**	17
Precedence	9	2	4	15
AMC Path	9	4	1	14
Graph	5	0	0	5
AMC Sibling	2	0	1	3
AMC DataType	1	0	0	1
Value Match	0	0	0	0
Similarity Flooding	0	0	0	0

Table 19: # of Times Ranked in Top-3

Table 19 reports on the number of times a matcher was ranked in the top-3 results. AWeighted is best with 23 out of 25 cases ranked at the top-3 best performers. Combining the results in tables 18 and 19, we conclude that while individual matchers effectiveness varies over different datasets, prediction-based ensembles present superior performance by allowing the matching system to identify those matchers that perform better on a specific task. We thus show the *robustness* of our predic-

tors as they are able to predict performance on a wide variety of datasets and matching algorithms. In the absence of any other a-priori information, AWeighted, using attribute predictors and thus taking the best of each matcher, is preferred over MWeighted and any individual matcher.

8 Related work

Schema and ontology matching research has been going on for more than 25 years now. With its roots in database integration, schema matching research has since expanded to include deep web and XML integration (e.g. [21, **?**,**?**]) and is now widely recognized as a standalone research field (see surveys [1, 40, 37, 41] and books [14, 16, 2]). Over the years, a significant body of work has been devoted to the identification of matchers, aimed at providing correspondences that will be effective from the user's point of view. Examples of algorithmic tools include COMA [9], Cupid [27], OntoBuilder [19], DIXSE [**?**], Similarity Flooding [31], Clio [32], Glue [11], and others [3, 5, 34, 23, 46, 22]. These have come out of a variety of research communities, including database management, information retrieval, the information sciences, data semantics and the semantic Web, and others. Benchmarks, such as the OAEI,[5] were developed to assess the effectiveness of matchers.

Over the years, a realization has emerged that schema matchers are inherently uncertain. A matcher may consider several possible correspondences as probable, and when it needs to choose, it may choose wrong [16]. A prime reason for the uncertainty of the matching process is the enormous ambiguity and heterogeneity of data description concepts: It is unrealistic to expect a single matcher to identify the correct mapping for any possible concept in a set. Since 2003, work on the uncertainty in schema matching has picked up, along with research on uncertainty in other areas of data management [17, 29, 12, 18, 6].

Matcher performance prediction serves as an uncertainty management tool and is useful in scenarios requiring uncertainty management, including that of bootstrapping [39]. We are unaware of any thorough methodological treatment of prediction in schema matching. However, recent papers show the need for better tools for prediction and a deeper analysis of the qualities of good predictors. We next review some first attempts to utilize prediction in the schema matching process.

The application of a-priori evaluation of schema features to direct and influence the execution of schema matching was first suggested by Tu and Yu [44], which used schema features to select execution strategies. Similar work was done by Peukert et. al. [36], using matrix features, calculated on the matching outcome, and schema features to influence matcher selection.

Our proposed application of prediction to dynamically set ensemble weights should be viewed in the context of ensemble construction. Contemporary ensemble tools such as LSD [10] and SMB [20] perform off-line learning of matcher weights in an ensemble. Mao et. al. [30] presented a method (HADAPT) (also used by Ngo et. al. [33]) of setting ensemble weights based on a single measure dubbed *harmony* (also known in the literature as *dominants* [16]). HADAPT dynamically assigns weights based on the matching outcome of individual matchers and their success prediction. However, this approach lacks methodological analysis of prediction properties. For example, reliance on a single prediction measure that assumes a $1:1$ matching, as done by HADAPT, limits the usefulness of the approach. The use of *prediction models* (Section 4.4), which combine the input from several predictors, allows a more flexible treatment of a schema matching problem instance. Also, prediction models allows the prediction of different qualities for different needs.

Both Mao et. al. and Peukert et. al. calculate prediction over a complete similarity matrix. This introduces an implicit assumption into matcher result evaluation. By assigning a single score to the matcher result, HADAPT implicitly assumes that poor results on some of the schema attributes mandate poor results for all attributes. In this work we hypothesize that some matchers may work well on some parts of the schema and others may work well on other parts. This hypothesis is validated empirically both by superior correlation of attribute-level predictors over matrix-level predictors (Section 7.1) and by the success of dynamic ensemble weight setting based on attribute predictors (Section 7.5). Therefore, by presenting predictors that generalize to schema fragments (and in particular to individual attributes), we allow combining the best results of each matcher.

This is the first thorough analysis of prediction as a stand-alone technique in schema matching that we are aware of. We find that the closest work to date is the one of Mao et. al. which provide a limited analysis of *Harmony* (which we identify as a predictor) over a monolithic dataset and a small number of schema pairs. This manuscript provides a general framework in which predictors may be conceptualized (Section 3), designed and evaluated (Section 4). Our evaluation of the proposed predictors (including Harmony, named Dominants in this manuscript) is extensive over a variety of datasets

[5] http://oaei.ontologymatching.org/

and matching algorithms from disparate matching systems. We hope this work will serve as an impetus for future work to suggest new predictors and refine existing ones using the conceptual, architectural and empirical tools we have presented.

9 Conclusions

We have introduced a new schema matching task called prediction, a subset of the more general evaluation task, assessing the quality of matching outcome in the absence of an exact match. We have provided a formal model for prediction using similarity spaces, shown a set of desirable properties of predictors, and demonstrated how to create predictors that satisfy these properties based on the similarity matrix abstraction. We have also shown the usage of predictors in three use-cases: We have shown how entry predictors can be used to enhance performance of second-line matchers. We have shown how match effectiveness can be improved using attribute predictors to dynamically set ensemble weights and we proposed an effective methodology for ranking Web sources, as part of a discovery task, using a combination of attribute and matrix predictors.

The empirical analysis validates the following four claims: 1) generalization allows attribute predictors to perform better than state-of-the-art methods in dynamically setting weights for ensembles; 2) predictor models improve on individual predictors by allowing tuning towards specific matching qualities and increasing correlation with predicted quality measures; and 3) predictors perform well in ranking the relevance of Web forms. 4) entry predictors can be used to improve match selection of second-line matchers.

This work is just a first step in analyzing predictors and their properties. Much of this work can serve as a guideline for the future development of different predictors of all granularity levels. In terms of future work, we intend on identifying new predictors, especially $K > 1$ predictors that were not examined in this work. Using the categorization of evaluators, presented in Section 3.2, the cases of $K = 2$ and $K > 2$ use more than one similarity vector defined over the same similarity space $\mathcal{V}_{\mathcal{S}}$. Looking back at Figure 2, prediction based on multiple vectors is illustrated at the right-hand-side of the figure, where multiple similarity matrices of the same schema pair are used for the prediction. $K = 2$ provides the opportunity to predict the degree of similarity between similarity vectors or the amount of information gain one vector represents over the other. A collection of similarity vectors ($K > 2$) may be evaluated to predict cluster related properties such as silhouette.

An additional direction of future research involves the investigation of the choice of distance measures and selection methods in idealizers and extending the similarity space into multi-schema matching scenarios.

Acknowledgment

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 256955.

References

1. C. Batini, M. Lenzerini, and S. Navathe. A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, 18(4):323–364, December 1986.
2. Z. Bellahsene, A. Bonifati, and E. Rahm, editors. *Schema Matching and Mapping*. Springer, 2011.
3. S. Bergamaschi, S. Castano, M. Vincini, and D. Beneventano. Semantic integration of heterogeneous information sources. *Data & Knowledge Engineering*, 36(3):215–249, 2001.
4. P.A. Bernstein and S. Melnik. Meta data management. In *Proc. 20th Int. Conf. on Data Engineering*, 2004. Tutorial Presentation.
5. S. Castano, V. De Antonellis, and S. De Capitani di Vimercati. Global viewing of heterogeneous data sources. *IEEE Trans. Knowl. and Data Eng.*, 13(2):277–297, 2001.
6. R. Cheng, J. Gong, and D.W. Cheung. Managing uncertainty of XML schema matching. In *Proc. 26th Int. Conf. on Data Engineering*, pages 297–308, 2010.
7. J. Cohen. *Statistical power analysis for the behavioral sciences*. Lawrence Erlbaum, 1988.
8. H. Do, S. Melnik, and E. Rahm. Comparison of schema matching evaluations. In *Proceedings of the 2nd Int. Workshop on Web Databases (German Informatics Society), 2002.*, 2002.
9. H.H. Do and E. Rahm. COMA - a system for flexible combination of schema matching approaches. In *Proc. 28th Int. Conf. on Very Large Data Bases*, pages 610–621, 2002.
10. A. Doan, P. Domingos, and A.Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 509–520, 2001.
11. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *Proc. 11th Int. World Wide Web Conf.*, pages 662–673, 2002.
12. X. Dong, A. Halevy, and C. Yu. Data integration with uncertainty. *The VLDB Journal*, 18:469–500, 2009.
13. N. Draper and H. Smith. *Applied Regression Analysis*. John Wiley & Sons, Inc., New York, 2nd edition edition, 1981.
14. J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.
15. M.J. Franklin, A.Y. Halevy, and D. Maier. From databases to dataspaces: a new abstraction for information management. *SIGMOD Record*, 34(4):27–33, 2005.

16. A. Gal. *Uncertain Schema Matching*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.
17. A. Gal, A. Anaby-Tavor, A. Trombetta, and D. Montesi. A framework for modeling and evaluating automatic semantic reconciliation. *VLDB J.*, 14(1):50–67, 2005.
18. A. Gal, M.V. Martinez, G.I. Simari, and V.S. Subrahmanian. Aggregate query answering under uncertain schema mappings. In *Proc. 25th Int. Conf. on Data Engineering*, pages 940–951, 2009.
19. A. Gal, G. Modica, H.M. Jamil, and A. Eyal. Automatic ontology matching using application semantics. *AI Magazine*, 26(1):21–32, 2005.
20. A. Gal and T. Sagi. Tuning the ensemble selection process of schema matchers. *Information Systems*, 35(8):845–859, 2010.
21. B. He and K. Chen-Chuan Chang. Statistical schema matching across Web query interfaces. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 217–228, 2003.
22. J. Kang and J.F. Naughton. On schema matching with opaque column names and data values. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 205–216, San Diego, California, 2003. ACM Press.
23. M.L. Lee, L.H. Yang, W. Hsu, and X. Yang. XCLUST: Clustering XML schemas for effective integration. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, pages 292–299, McLean, Virginia, 2002. ACM Press.
24. Y. Lee, M. Sayyadian, A. Doan, and A. Rosenthal. eTuner: tuning schema matching software using synthetic scenarios. *VLDB J.*, 16(1):97–122, 2007.
25. D.G. Luenberger. *Optimization by vector space methods*. Wiley-Interscience, 1997.
26. J. Madhavan, P.A. Bernstein, A. Doan, and A. Halevy. Corpus-based schema matching. In *Proc. 21st Int. Conf. on Data Engineering*, pages 57–68, 2005.
27. J. Madhavan, P.A. Bernstein, and E. Rahm. Generic schema matching with Cupid. In *Proc. 27th Int. Conf. on Very Large Data Bases*, pages 49–58, 2001.
28. J. Madhavan, S. Jeffery, S. Cohen, X. Dong, D. Ko, C. Yu, and A. Halevy. Web-scale data integration: You can only afford to pay as you go. In *Proceedings of CIDR*, pages 342–350, 2007.
29. M. Magnani, N. Rizopoulos, P. McBrien, and D. Montesi. Schema integration based on uncertain semantic mappings. In *Proc. 24th Int. Conf. on Conceptual Modeling*, pages 31–46, 2005.
30. M. Mao, Y. Peng, and M. Spring. A harmony based adaptive ontology mapping approach. In *Proc. of Semantic Web and Web Services*, 2008.
31. S. Melnik, E. Rahm, and P.A. Bernstein. Rondo: A programming platform for generic model management. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 193–204, 2003.
32. R.J. Miller, M.A. Hernàndez, L.M. Haas, L.-L. Yan, C.T.H. Ho, R. Fagin, and L. Popa. The Clio project: Managing heterogeneity. *SIGMOD Record*, 30(1):78–83, 2001.
33. D.H. Ngo, Z. Bellahsene, et al. Yam++-a combination of graph matching and machine learning approach to ontology alignment task. *Journal of Web Semantics*, 16, 2012.
34. L. Palopoli, L.G. Terracina, and D. Ursino. Experiences with using DIKE, a system for supporting cooperative information systems and data warehoude design. *Information Systems*, 28(7):835–865, 2003.
35. E. Peukert, J. Eberius, and E. Rahm. AMC-a framework for modelling and comparing matching systems as matching processes. In *Proc. 27th Int. Conf. on Data Engineering*, pages 1304–1307. IEEE, 2011.
36. E. Peukert, J. Eberius, and E. Rahm. A self-configuring schema matching system. In *Proc. 28th Int. Conf. on Data Engineering*, 2012.
37. E. Rahm and P.A. Bernstein. A survey of approaches to automatic schema matching. *VLDB J.*, 10(4):334–350, 2001.
38. T. Sagi and A. Gal. Non-binary evaluation for schema matching. In *Proc. 31st Int. Conf. on Conceptual Modeling*, October 2012.
39. A.D. Sarma, X. Dong, and A.Y. Halevy. Bootstrapping pay-as-you-go data integration systems. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 861–874, 2008.
40. A. Sheth and J. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183–236, 1990.
41. P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. *Journal of Data Semantics*, 4:146 – 171, December 2005.
42. K.P. Smith, M. Morse, P. Mork, M.H. Li, A. Rosenthal, D. Allen, and L. Seligman. The role of schema matching in large enterprises. In *CIDR 2009, Fourth Biennial Conference on Innovative Data Systems Research*, Asilomar, CA, USA, January 2009.
43. R. G. D. Steel and J. H. Torrie. *Principles and Procedures of Statistics*. McGraw-Hill, New York, 1960.
44. K.W. Tu and Y. Yu. CMC: Combining multiple schema-matching strategies based on credibility prediction. In Lizhu Zhou, Beng Ooi, and Xiaofeng Meng, editors, *Database Systems for Advanced Applications*, volume 3453 of *Lecture Notes in Computer Science*, pages 995–995. Springer Berlin / Heidelberg, 2005.
45. J. Wang, J.R. Wen, F. Lochovsky, and W.Y. Ma. Instance-based schema matching for web databases by domain-specific query probing. In *Proc. 30th Int. Conf. on Very Large Data Bases*, pages 408–419. VLDB Endowment, 2004.
46. X. Yang, M.L. Lee, and T.W. Ling. Resolving structural conflicts in the integration of XML schemas: A semantic approach. In *Proceedings of the International Conference on Conceptual Modeling (ER)*, Chicago, Illinois, July 2003. Lecture Notes in Computer Science, Springer.
47. J. Zobel and A. Moffat. Exploring the similarity space. *SIGIR Forum*, 32:18–34, April 1998.