Schema Normalization for Improving Schema Matching *

Serena Sorrentino[2], Sonia Bergamaschi[1], Maciej Gawinecki[2], Laura Po[1]

[1] DII - University of Modena and Reggio Emilia, Italy
[2] ICT Doctorate School - University of Modena and Reggio Emilia, Italy
name.surname@unimore.it

Abstract. Schema matching is the problem of finding relationships among concepts across heterogeneous data sources (heterogeneous in format and in structure). Starting from the "hidden meaning" associated to schema labels (i.e. class/attribute names) it is possible to discover relationships among the elements of different schemata. Lexical annotation (i.e. annotation w.r.t. a thesaurus/lexical resource) helps in associating a "meaning" to schema labels. However, accuracy of semi-automatic lexical annotation methods on real-world schemata suffers from the abundance of non-dictionary words such as compound nouns and word abbreviations. In this work, we address this problem by proposing a method to perform schema labels *normalization* which increases the number of comparable labels. Unlike other solutions, the method semi-automatically expands abbreviations and annotates compound terms, with a minimal manual effort. We empirically prove that our normalization method helps in the identification of similarities among schema elements of different data sources, thus improving schema matching accuracy.

1 Introduction

Schema matching is a critical step in many applications such as: data integration, data warehousing, E-business, semantic query processing, peer data management and semantic web applications [14]. In this work, we focus on schema matching in the context of data integration [2], where the goal is the creation of mappings between heterogeneous data sources (heterogeneous in format and in structure). Mappings are obtained by a schema matching system by using a set of semantic matches (e.g. location = area) between different schemata. A powerful mean to discover matches is the understanding of the "meaning" behind the names denoting schemata elements, i.e. labels in the following [17]. In this context, lexical annotation, i.e. the explicit association of the "meaning" (synset/sense in WordNet (WN) terminology [8]) to a label w.r.t. a thesaurus (WN in our case) is a key tool.

* **Acknowledgements:** This work was partially supported by MUR FIRB Network Peer for Business project (http://www.dbgroup.unimo.it/nep4b) and by the IST FP6 STREP project 2006 STASIS (http://www.dbgroup.unimo.it/stasis).

The strength of a thesaurus, like WN, is the presence of a wide network of semantic relationships among words meanings, thus providing a corresponding inferred semantic network of lexical relationships among the labels of different schemata. Its weakness, is that it does not cover, with the same detail, different domains of knowledge and that many domain dependent terms, as *non-dictionary words*, may not be present in it. Non-dictionary words include compound nouns (CNs), abbreviations etc.

The result of automatic lexical annotation techniques is strongly affected by the presence of these non-dictionary words in schemata. For this reason, a method to expand abbreviations and to semantically "interpret" CNs is required. In the following, we will refer to this method as schema labels *normalization*. Schema labels normalization helps in the identification of similarities between labels coming from different data sources, thus improving schema mapping accuracy.

A manual process of label normalization is laborious, time consuming and itself prone to errors. Starting from our previous works on semi-automatic lexical annotation of structured and semi-structured data sources [3], we propose a semi-automatic method for the normalization of schema labels able to expand abbreviations and to annotate CNs w.r.t. WN.

Our method is implemented in the MOMIS (Mediator envirOnment for Multiple Information Sources) system [4, 2]. However, it may be applied in general in the context of schema mapping discovery, ontology merging and data integration system. Moreover, it might be effective for reverse engineering tasks, when we need to abstract an entity relationship schema for a legacy database.

The rest of the paper is organized as follows. In Section 2, we define the problem in the context of schema matching; in Sections 3, 4 and 5 we describe our method with reference to classification of labels for normalization, abbreviations expansion and CNs interpretation, respectively. Section 6 describes related works; in Section 7 we demonstrate the effectiveness of the method with extensive experiments on real-world data sets; finally Section 8 is devoted to conclusion and future work.

2 Problem definition

Elements names represent an important source for assessing similarity between schema elements. This can be done semantically by comparing their meanings.
Definition 1 *Lexical annotation of a schema label is the explicit assignment of its meaning w.r.t. a thesaurus.*

Starting from the lexical annotation of schema labels we can derive lexical relationships among them on the basis of the semantic relationships defined in WN among their meanings.
Definition 2 *A compound noun (CN) is a word composed of more than one word called CN constituents. It is used to denote a concept, and can be interpreted by exploiting the meanings of its constituents.*
Definition 3 *An abbreviation is a shortened form of a word or phrase, that*

consists of one or more letters taken from the word or phrase.
Definition 4 *Let S and T be two heterogeneous schemata, and $E_S = \{s_1, ..., s_n\}$ and $E_T = \{t_1, ..., t_k\}$, respectively, the set of labels of S and T. A lexical relationship is defined as the triple $< s_i, t_j, R >$ where $s_i \in E_S$, $t_j \in E_T$ and R specifies a lexical relationship between s_i and t_j. The lexical relationships are:*

- *SYN: (Synonym-of), defined between two labels that are synonymous (it corresponds to a WN synonym relationship);*
- *BT: (Broader Term), defined between two labels where the first is more general then the second (the opposite of BT is NT, Narrower Term) (it corresponds to a WN hypernym/hyponym relationship);*
- *RT: (Related Term) defined between two labels that are related in a meronymy hierarchy (it corresponds to a WN meronym relationship).*

Figure 1 shows two schemata to be integrated, containing many labels with non-dictionary CNs (e.g. "CustomerName"), acronyms (e.g. "PO") and word abbreviations (e.g. "QTY"). These labels cannot be directly annotated, because they do not have an entry in WN. Schema label normalization (also called *linguistic normalization* in [14]) is the reduction of the form of each label to some standardized form that can be easily recognized. In our case, with labels normalization we intend the process of abbreviations expansion, and CNs interpretation.
Definition 5 *The interpretation of a CN is the task of determining the semantic relationships holding among the constituents of a CN.*
Definition 6 *Abbreviation expansion is the task of finding a relevant expansion (long form) for a given abbreviation (short form).*

Schema labels normalization improves the schema matching process by reducing the number of discovered *false positive/false negative relationships.*
Definition 7 *Let $< s_i, t_j, R >$ be a lexical relationship. Then it is a false positive relationship, if the concept denoted by the label s_i is not related by R to the concept denoted by the label t_j.*

For example, let us consider the two schema labels "CustomerName" and "CLIENTADDRESS", respectively in the source "PurchaseOrder" and "PO" (Figure 1). If we annotate separately the terms "Customer" and "Name", and "CLIENT" and "ADDRESS", then we would discover a SYN relationship between them, because the terms "Customer" and "CLIENT" share the same WN meaning. In this way, a false positive relationship is discovered because these two CNs represent "semantically distant" schema elements.
Definition 8 *Let $< s_i, t_j, R >$ be a lexical relationship. R is a false negative relationship if the concept denoted by the label s_i is related by R to the concept denoted by the label t_j, but the schema matching process does not return this relationship.*

Let us consider two corresponding schema labels: "amount" of the "PurchaseOrder" source and "QTY" (abbreviation for "quantity") of the "PO" source (Figure 1). Without abbreviation expansion we cannot discover that there exists a SYN relationship between the elements "amount" and "QTY".

Fig. 1. Graph representation of two schemata with elements containing abbreviations and CNs: (a) relational database schema, (b) XML schema.

3 Classifying schema labels for normalization

The schema labels normalization process consists of three phases: (1) classification for normalization, (2) abbreviation expansion and (3) CNs interpretation. In this section we focus on the first phase.

Classification for normalization consists of the following three steps: (1) selecting whole labels that need to be normalized, (2) tokenizing selected labels into separate words, and (3) identifying abbreviations among isolated words. To select labels that need to be normalized, we propose the following classification heuristic:

Definition 9 *A label has to be normalized, if (a) it occurs on the list of standard schema abbreviations or (b) neither it nor its stem has an entry in a dictionary.*

In this way CNs which have an entry in WN (e.g. "company name") will be treated as single words, while for CNs that do not have an entry in WN (non-dictionary CNs) we apply our CNs interpretation method. Additionally, the list of standard schema abbreviations is employed here to reduce the number of false negatives caused by *legitimate* English words, that have been used for abbreviations in the schema context, e.g. "id", the prevalent abbreviation in analyzed schemata, is a dictionary word in WN.

We perform tokenization by using one of the pre-existing approaches [9]: *simple* – based on camel case and punctuation, and *greedy* – handling also multi-word names without clearly defined word boundaries, e.g. 'WHSECODE'. The latter iteratively looks for the biggest prefixing/suffixing dictionary words and user-defined abbreviations in non-dictionary words.

For instance, let us assume we are classifying "PODelivery" label. This is not a dictionary word nor a standard schema abbreviation, thus classified for normalization. The tokenization splits it into into: "PO" and "Delivery" words, where the first is identified as an abbreviation.

4 Automatic abbreviations expansion

Automatic abbreviation expansion of already identified abbreviations requires the execution of the following steps: (1) searching for potential long forms for

the given short form; and (2) selecting the most appropriate long form from the set of potential long form candidates.

A schema can contain both *standard* and *ad hoc* abbreviations. Standard abbreviations either (a) denote important and repeating domain concepts (*domain standard abbreviations*), e.g. "ISBN" (International Standard Book Number) or (b) are standard suffix/prefix words used to describe how a value of a given schema element is represented (*standard schema abbreviations*), e.g. "Ind" (Indicator). On the contrary, ad hoc abbreviations are mainly created to save space, from phrases that would not be abbreviated in a normal context [22, 11].

To observe how different types of abbreviations can be handled automatically we analyzed short forms and their corresponding long forms in several open-source schemata . Based on our manual inspection, we found two sources relevant for finding possible long form candidates for ad hoc abbreviations: (a) *context* (`C`) of short form occurrence, as it is common practice to an attribute name with a short form of a class name, for instance "recentchanges" table contains "rc_user" and "rc_params"; (b) a *complementary schema* (`CS`) that we integrate with inspected schema; e.g. a short form "uom" in the XML schema (Figure 1b) can be expanded with long form "unit Of Measure" from the relational database schema (Figure 1a). Moreover, we found online abbreviation dictionary (`OD`) very useful for expanding domain standard abbreviations. Finally, as the list of standard schema abbreviations is bound we were able to discover a list of possible expansions for all of them and define as a user-defined dictionary (`UD`).

4.1 Proposed algorithm for abbreviation expansion

To handle different types of abbreviations the algorithm uses four aforementioned sources of long forms. However, the syntax of a short form itself does not provide any mean for distinguishing between ad hoc and standard abbreviations and thus we are not able to choose in advance the relevant source for expansion of a given short form. Nevertheless, we can consider the context and complementary schema as the most relevant sources in general, because they closely reflect the intention of a schema designer.

For each identified abbreviation the algorithm inquires all four sources for long form candidates, scores candidates according to the relevance of the source, combines scores of repeating long forms and chooses the top-scored one. The whole process is shown in Figure 2.

Combining expansion sources. Technically, for each identified short form sf the algorithm creates a list of long form candidates: $(< lf_i; sc(lf_i) >)_i$ obtained from all the sources where $sc(lf_i) \in [0, 1]$. The algorithm selects the top-scored long form candidate from the list. If the list is empty, then the original short form is preserved. The score of lf_i ($sc(lf_i)$) is computed by combining scores from the single sources:

$$sc(lf_i) = \alpha_{UD} \cdot sc_{UD}(lf_i) + \alpha_{CS} \cdot sc_{CS}(lf_i) + \alpha_C \cdot sc_C(lf_i) + \alpha_{OD} \cdot sc_{OD}(lf_i)$$

where $\alpha_{UD} + \alpha_{CS} + \alpha_C + \alpha_{OD} = 1$ are weights of sources relevance.

INPUT: sf – short form occurrence, **OUTPUT:** lf – long form for sf
compute the list $L_{UD} := (< lf_{UD}, 1 >)$, where lf_{UD} is a matching long form in UD
compute the list $L_{CS} := (< lf_{CS}, 1 >)$, where lf_{CS} is a matching long form in CS
compute the list $L_C := (< lf_C, 1 >)$, where lf_C is a matching long form in C of sf
compute the list $L_{OD} := (< lf_{OD,i}, sc_{OD}(lf_{OD,i}) >)_i$, where $lf_{OD,i}$ is
  a matching long form in OD
$L = L_{UD} \cup L_{CS} \cup L_C \cup L_{OD}$ // combine long forms scores
$lf := \arg\max_{lf_i \in L} sc(lf_i)$

Fig. 2. Procedure for selecting a long form for the given short form.

Obtaining expansions from sources. For a user-defined dictionary, a context and a complementary schema sources the score of lf_i is 1, if lf_i is found in the given source or 0 – otherwise. To define a *context* let us suppose sf_i be a short form identified in a label l. The label l is either: (a) an attribute of a class c or (b) a class belonging to schemata s. Then the context of sf_i is the class c or schema s. The context is retrieved for possible long form candidates using the four abbreviation patterns (practically, regular expressions created from characters of a short form) proposed in [7]. The *labels* in the schema complementary to the schema in which sf appears are retrieved for matching long form candidates using the same abbreviation patterns as in the context. Only the first matching candidate is considered. For instance, when expanding the "PO" abbreviation in "PODelivery" element the algorithm receives the following expansions from the particular sources: (a) from online dictionary: "Purchase Order", "Parents Of", (b) from context: "Purchase Order", (c) from complementary schema: "Purchase Order". The context of "PODelivery" is in this case the name of its schema, while "PO" is a complementary schema. Next, the algorithm merges lists of proposed candidates into a single one: "Purchase Order", "Parents Of".

Scoring expansions from an online dictionary. The online dictionary may suggest more then one long form for a given short form. For this purpose we propose disambiguation technique based on two factors: (a) the number of domains a given long form shares with both schemata and (b) its popularity in these domains. The intuition is that only those expansions are relevant, that are the most popular in the domains described by both schemata. We assume information about the domain of a long form and its popularity is given by the online dictionary.

Practically, we may define score of a long form candidate — $sc_{OD}(lf_i)$ — as follows:

$$sc_{OD}(lf_i) = \sum_{d \in CD(lf_i, schemata)} \frac{p(lf_i, d)}{P_{schema}},$$

$$P_{schema} = \sum_j \sum_{d \in CD(lf_j, schemata)} p(lf_j, d),$$

$$CD(lf_i, schemata) = D(lf_i) \cap D(schemata)$$

where $D(schemata)$ is a list of prevalent WN Domains[3] associated with schemata to integrate [3]. If there is no shared domain for any long form candidate, then score is computed as a general popularity of a long form candidate. Computation of $CD(lf_i, schemata)$ — the intersection of prevalent domains and domains associated with long form lf_i — involves the mapping between the categorization system of an online abbreviation dictionary and WN Domains classification. The mapping has been created by obtaining automatically all corresponding domains for words in names of categories, and then, manually, by analyzing sample abbreviations in questionable mappings.

There can be more than one online dictionary entry describing the same long form lf_i, but in different domains. Therefore, the entry can be modeled as a combination of a long form lf_i and a domain $d_{i,k} \in D(lf_i)$ in which it appears with the associated popularity. Formally, we define the t-th dictionary entry in the following form: $< e_t, p(e_t) >$, where $e_t =< lf_i; d_{i,k}) >$ and $d_{i,k} \in D(lf_i)$ is the k-th domain in the set of domains ($D(lf_i)$), in which the long form lf_i appears. The popularity $p(e_t)$ is not explicitly reported by the considered dictionary but can be easily estimated from the order of descending popularity in respect to which entries are returned by the dictionary. Thus we are able to calculate $p(e_t)$ using the following induction: $p(e_{t+1}) = p(e_t)/\kappa$, $p(e_1) = 1.0$, where $\kappa > 1$ is an experimentally defined factor[4].

For example, commerce, sociology and metrology are the prevalent domains for the schemata in Figure 1. Among three entries (with given categories) returned by the dictionary for "PO" — "Purchase Order" (Accounting), "Parents Of" (Law), "Purchase Order" (Military) — only the first one matters, because its category is mapped to commerce WN Domain — one of the schemata domains.

5 Compound noun interpretation

In order to perform semi-automatic CNs annotation, a method for their interpretation need to be devised. In the natural language disambiguation literature different CNs classifications have been proposed [21, 19]. In this work we use the classification introduced in [21], where CNs are classified in four distinct categories: *endocentric*, *exocentric*, *copulative* and *appositional* and we consider only endocentric CNs.

Definition 10 *An Endocentric CN consists of a head (i.e. the categorical part that contains the basic meaning of the whole CN) and modifiers, which restrict this meaning. A CN exhibits a modifier-head structure with a sequence of nouns composed of a* head *noun and one or more modifiers where the head noun occurs always after the modifiers.*

The constituents of endocentric compounds are noun-noun or adjective-noun, where the adjective derives from a noun (e.g. "dark room", where the adjective "dark" derives from the noun "darkness"). Our restriction on endocentric CNs is motivated by the following observations: (1) the vast majority of CNs of schemata

[3] http://wndomains.itc.it/wordnetdomains.html

[4] In experiments we successfully use $\kappa := 1.2$

Fig. 3. The CNs interpretation process.

fall in endocentric category; (2) endocentric CNs are the most common type of CNs in English; (3) exocentric and copulative CNs, which are represented by a unique word, are often present in a dictionary; (4) appositional CNs are not very common in English and less likely used as elements of a schema. We consider endocentric CNs composed of only two constituents, because CNs consisting of more than two words need to be constructed recursively by *bracketing* them into pairs of words and then interpreting each pair.

Our method can be summed up into four main phases: (1) CN constituents disambiguation; (2) redundant constituents identification; (3) CN interpretation via semantic relationships; (4) creation of a new WN meaning for a CN.

Phase 1. CN constituents disambiguation
In this phase the correct WN synset of each constituent is chosen in two steps:

1. *Compound Noun syntactic analysis*: this step performs the syntactic analysis of CN constituents, in order to identify the syntactic category of its head and modifier. If the CN does not fall under the endocentric syntactic structure, then it is ignored;
2. *Disambiguating head and modifier*: this step is part of the general lexical disambiguation problem. By applying our CWSD (Combined Word Sense Disambiguation) algorithm [3], each word is automatically mapped onto its corresponding WN 2.0 synsets.

As shown in Figure 3-a, for example, for the schema element "DeliveryCompany" we obtain the two constituents annotated with the correspondent WN meanings (i.e. "$Company_{\#1}$" and "$Delivery_{\#1}$").

Phase 2. Redundant constituents identification and pruning
During this phase we control whether a CN constituent is a *redundant word*. Redundant words are the words that do not contribute new information as their semantic contribution can be derived from the schema or from the lexical resource. For example, the typical situation in a schema is when the name of a class is a part of its attribute name (see for instance the "SHIPMENTADDRESS" attribute of the "SHIPMENT" class in Figure 1-b). As a result, the constituent class name is not considered, because the relationship holding among a class and its attributes can be derived from the schema.

Phase 3. CN interpretation via semantic relationships
This phase concerns selecting from a set of predefined relationships the one that in the best way captures the semantic relation between the meanings of a head and a modifier. The problem of devising a set of semantic relationships to be considered for the CNs interpretation has been widely discussed in the natural language disambiguation literature [12]. In [19] Levi defines a set of nine possible semantic relationships to interpret CNs: CAUSE ("flu virus"), HAVE ("college town"), MAKE ("honey bee"), USE ("water wheel"), BE ("chocolate bar"), IN ("mountain lodge"), FOR ("headache pills"), FROM ("bacon grease") and ABOUT ("adventure story"). On the contrary, Finin in [16] claims an unlimited number of semantic relationships. We choose the Levi semantic relationships set, as it is the best choice in the simplified context of a data integration scenario. According to [15], our method is based on the following assumption:
Definition 11 *The semantic relationship between a head and its modifier of a CN is derived from the one holding between their top level WN nouns in the WN nouns hierarchy.*

The WN nouns hierarchy has been proven to be very useful in the CNs interpretation task [12]. The top level concepts of the WN hierarchy are the 25 *unique beginners* (e.g. act, animal, artifact etc.) for WN English nouns defined by Miller in [8]. These unique beginners were selected after considering all the possible adjective-noun or noun-noun combinations that could be expected to occur and are suitable to interpret noun-noun or adjective-noun CNs as in our case.

For each possible couple of the unique beginners we manually associate the relationship from the Levi's set that best describes the meaning of this couple. For example, for the unique beginner pair "group and act" we choose the Levi's relationship MAKE (e.g. "group MAKE act"), that can be expressed as: a group performs an act. In this way, as shown in Figure 3b, we are able to interpret the label "DeliveryCompany" with the MAKE relationship, because "Company" is an hyponym of "group" and "Delivery" is an hyponym of "act".

Our method requires an initial human intervention to associate to each pair of unique beginners the right relationship. However, it may be considered acceptable, when compared with the much greater effort required for other approaches based on pre-tagged corpus where the number of CNs to be anno-

tated is much higher [12, 18]. Moreover, the method is independent from the domain under consideration and can be applied to any thesaurus providing a wide network of hyponym/hypernym relationships between defined meanings.

Phase 4. Creation of a new WN meaning for a CN
During this phase, we create a new WN meaning for the given CN. We distinguish the following two steps:

1. *Gloss definition*: during this step we create the *gloss* to be associated with a CN, starting from the relationship associated to it and exploiting the glosses of the CN constituents. Figure 3-c shows an example of this phase. The glosses of the constituents "Company" and "Delivery" are joined together according to the relationship MAKE.
2. *Inclusion of the new CN meaning in WN*: the insertion of a new CN meaning into the WN hierarchy implies the definition of its relationships with the other WN meanings. As the concept denoted by a CN is a subset of the concept denoted by the head, we assume that a CN inherits most of its semantics from its head [21]. Starting from this consideration, we can infer that the CN is related, in the WN hierarchy, to its head by an hyponym relationship. Moreover, we represent the CN semantics related to its modifier by inserting a generic relationship RT (*Related term*), corresponding to WN relationships as *member meronym*, *part meronym* etc. However, the insertion of these two relationships is not sufficient, it is necessary to discover also the relationships of the new inserted meaning w.r.t. the other WN meanings. For this purpose, we use the WNEditor tool to create/manage the new meaning and to set relationships between it and the WN ones [4]. The WNEditor automatically retrieves a list of candidate WN meanings sharing similarities with the new meaning. Then, the user is asked to explicitly declare the type of relationship (hyponymy, meronymy etc.) to relate the new meaning to another, if any. Figure 3-d shows an example of this step.

6 Related work

The problem of linguistic normalization has received much attention in different areas such as: machine translation, information extraction, information retrieval.

Many abbreviation expansion techniques are based on the observation that in documents the short forms and their long forms usually occur together in patterns. Selecting the most relevant long form is made w.r.t. different factors such as: *inverted frequency* [13], *document scope* [7] or *syntactic similarity* [11].

Many works in the literature for interpreting CNs involve costly pre-tagged corpus and heavy manual intervention [12, 18]. These approaches are based on a statistic co-occurrence of a relationship r between two words on corpus that contain different CNs manually labeled with the right semantic relationship. According to [15], we claim that the cost of acquiring knowledge from manually tagged corpus for different domains may overshadow the benefit of interpreting the CNs.

Number of	Labels	Non-dictionary words	CNs	Abbreviations
Schema 1	117	66	33	62
Schema 2	51	28	28	24

Table 1. Characteristics of test schemata.

Surprisingly, current schema integration systems either do not consider the problem of abbreviation expansion at all or solve it in non-scalable way by inclusion of a simple user-defined abbreviation dictionary [20, 1]. Lack of scalability comes from the fact that: (a) the vocabulary evolves over the time and it is necessary to maintain the table of abbreviations and (b) the same abbreviations can have different expansions depending on the domain. Moreover, this approach still requires an intervention of a schema/domain expert.

Similarly, in the context of data integration and schema mapping only a few papers address the problem of CNs interpretation. In [23] a preliminary CNs comparison for ontology mapping is proposed. This approach suffers from two main problems: first, it starts from the assumption that the ontology entities are accompanied with comments that contain words expressing the relationship between the constituents of a CN; second, it is based on a set of rules manually created. The well know CUPID algorithm [20], during the schema labels normalization phase, considers abbreviations, punctuation, etc. but not CNs. Generally, schema and ontology matching tools employing *syntactical matching* techniques do not interpret nor normalize CNs but they treat words in CNs separately [6].

7 Experimental results

We implemented our method for schema labels normalization in the MOMIS system [2]. Schema labels normalization is performed during the lexical annotation phase: during this phase each schema element of a local source is semi-automatically annotated by the CWSD algorithm. We also used *Abbreviations.com* online abbreviation dictionary. We tested the performance of our method over the two relational schemata of the well known Amalgam integration benchmark for bibliographic data [10]. Table 1 summarizes the test schemata features that are particularly suitable for the test.

Our evaluation goals were: (1) measuring the performance of our method, (2) checking whether our method improves the *lexical annotation* process and finally (3) estimating the effect of schema labels normalization on the *lexical relationships discovery* process.

7.1 Evaluating schema labels normalization method

The normalization process consists of: classification, abbreviations expansion and CNs interpretation. Since the errors of each step can be cumulated in the phases following, we evaluated performance of each step separately (using correct manually prepared input) and then as a whole.

	Precision	Recall
Total labels normalization (after GT/Ispell)	0.84	0.74

Table 2. Result of evaluation of schema labels normalization method.

Classification. We consider a label correctly classified for normalization if w.r.t. manual classification the label has been correctly tokenized and all abbreviations and CNs in the label have been identified. We evaluated classification method in two variants depending on the tokenization method used: (1) `ST`: simple and (2) `GT/Ispell`: greedy with Ispell English words list[5] as a dictionary (see Section 3 for details). The `ST` reaches nearly the same correctness (0.92) as `GT/Ispell` (0.93), because the schemata contain relatively few labels with unclearly undefined word boundaries (e.g. "bktitle").

Abbreviations expansions. W.r.t. manually classified and tokenized schemata labels the algorithm *expanded abbreviations* correctly in 90% of identified abbreviations. There are two reasons for errors: (a) lack of correct expansions in the external sources (context, documentation, online dictionary); and (b) partial matching of multi-words abbreviations, e.g. there is no correct matching in any source for "RID", but "ID" can be found in user-defined dictionary, while "R", standing for "record", in the element context.

CNs interpretation has been evaluated in terms of recall (the number of correct interpretation divided by the total number of CNs) and precision (the number of correct interpretations divided by the total number of interpreted CNs). During the evaluation process, a CN has been considered *correctly interpreted* if the Levi's relationship manually selected was the same as the one returned by our method. The CNs interpretation method obtains good result both for precision (0.86) and recall (0.75). However, the recall value is affected by the presence in the schemata of CNs such as "ManualPublished" or "ArticlePublished" that our method is not able to interpret as these terms are not endocentric CNs.

Table 2 shows the result of the whole schema labels normalization process by using our automatic classification, abbreviation expansion and semi-automatic CNs interpretation together.

7.2 Evaluating the lexical annotation process

The annotation results have been evaluated in terms of recall (the number of correct annotations divided by the total number of schema labels) and precision (the number of correct annotations divided by the total number of annotations). Table 3 shows the result of lexical annotation performed by CWSD without/with our normalization method. Without schema normalization, CWSD obtains a very low recall value, because a lot of CNs and abbreviations are present in the schemata. The application of our method permits to increase the recall while preserving the high precision.

[5] Ispell is a popular tool for spelling errors correction: http://wordlist.sourceforge.net/.

	Precision	Recall
CWSD	0.81	0.35
CWSD + Labels Normalization	0.83	0.78

Table 3. Comparison of lexical annotation (CWSD) without/with normalization.

	Precision	Recall	F-Measure
Lexical rel. discovered	0.58	0.33	0.42
Lexical rel. discovered + Normalization	0.90	0.75	0.82

Table 4. Comparison of lexical relationships discovered without/with normalization.

7.3 Evaluating the discovered lexical relations

To evaluate the quality of the lexical relationship discovered, we use the match quality measure defined in [5]. In particular, we compare the manually determined lexical relationships (MR) with the relationships returned by our semi-automatic method (AR). We determine: the true positives, i.e. correctly identified relationships (B), as well as the false positives (C) and the false negatives (A). Based on the cardinalities of these sets, the following quality measure are computed:

- *Precision*= $\frac{|B|}{|B|+|C|}$ reflects the reliability of the relationships predictions;
- *Recall*= $\frac{|B|}{|A|+|B|}$ specified the share of real relationships that is found;
- *F-Measure*= $2 * \frac{Precision * Recall}{Precision + Recall}$ – a combined measure of precision and recall.

Table 4 shows the result of the lexical relationships discovery process without/with normalization. In the first row we show the discovered lexical relationships without abbreviation expansion and considering the constituents of a CNs as single words with an associated meaning. Without schema labels normalization we discover few lexical relationships with low precision due the presence of a lot of false positive relationships. Instead, with our method we are able to improve recall and precision significantly.

8 Conclusion & future work

In this paper we presented a method for the semi-automatic normalization of schema elements labeled with abbreviations and CNs in a data integration environment. The experimental results have shown the effectiveness of our method, which significantly improves the result of the automatic lexical annotation process, and as a consequence, improves the quality of the discovered inter-schema lexical relationships. We demonstrated that, due to the frequency and productivity of non-dictionary words, a data integration system, during the lexical annotation phase, cannot ignore CNs and abbreviations without compromising recall. Future work will be devoted to investigate on the role of the set of semantic relationships chosen for the CNs interpretation process.

References

1. D. Aumueller, H. H. Do, S. Massmann, and E. Rahm. Schema and ontology matching with COMA++. In *SIGMOD'05*, pages 906–908, 2005.
2. S. Bergamaschi, S. Castano, and M. Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Record*, 28(1):54–59, 1999.
3. S. Bergamaschi, L. Po, and S. Sorrentino. Automatic annotation for mapping discovery in data integration systems. In *SEBD 2008*, pages 334–341, 2008.
4. D.Beneventano, S. Bergamaschi, F. Guerra, and M. Vincini. Synthesizing an integrated ontology. *IEEE Internet Computing*, 7(5):42–51, 2003.
5. H. H. Do, S. Melnik, and E. Rahm. Comparison of schema matching evaluations. In *Web, Web-Services, and Database Systems*, pages 221–237, 2002.
6. B. T. Le et al. On ontology matching problems - for building a corporate semantic web in a multi-communities organization. In *ICEIS (4)*, pages 236–243, 2004.
7. E. Hill et al. AMAP: automatically mining abbreviation expansions in programs to enhance software maintenance tools. In *MSR '08*, 2008.
8. George A. Miller et al. Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244, 1990.
9. H. Feild et al. An Empirical Comparison of Techniques for Extracting Concept Abbreviations from Identifiers. In *SEA'06*, November 2006.
10. R. J. Miller et al. The Amalgam Schema and Data Integration Test Suite. www.cs.toronto.edu/ miller/amalgam, 2001.
11. R. Uthurusamy et al. Extracting knowledge from diagnostic databases. *IEEE Expert: Intelligent Systems and Their Applications*, 8(6):27–38, 1993.
12. V. Nastase et al. Learning noun-modifier semantic relations with corpus-based and wordnet-based features. In *AAAI*, 2006.
13. W. Wong et al. Integrated scoring for spelling error correction, abbreviation expansion and case restoration in dirty text. In *AusDM '06*, pages 83–89, 2006.
14. J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.
15. J. Fan, K. Barker, and B. W. Porter. The knowledge required to interpret noun compounds. In *IJCAI*, pages 1483–1485, 2003.
16. T. W. Finin. The semantic interpretation of nominal compounds. In *AAAI*, pages 310–312, 1980.
17. F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-match: an algorithm and an implementation of semantic matching. In *Semantic Interoperability and Integration*, 2005.
18. M. Lapata. The disambiguation of nominalizations. *Computational Linguistics*, 28(3):357–388, 2002.
19. Judith N. Levi. *The Syntax and Semantics of Complex Nominals*. Academic Press, New York, 1978.
20. J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In *VLDB*, pages 49–58, 2001.
21. I. Plag. *Word-Formation in English*. Cambridge Textbooks in Linguistics. Cambridge University Press, New York, 2003.
22. L. Ratinov and E. Gudes. Abbreviation Expansion in Schema Matching and Web Integration. In *WI '04*, pages 485–489, 2004.
23. X. Su and J. A. Gulla. Semantic enrichment for ontology mapping. In *NLDB*, pages 217–228, 2004.